Privacy-enhanced Network Monitoring

Nils Ulltveit-Moe

Privacy-enhanced Network Monitoring

Doctoral Dissertation for the Degree *Philosophiae Doctor (PhD)* at
the Faculty of Engineering and Science, Specialisation in
Information and Communication Technology

University of Agder
Faculty of Engineering and Science
2014

Doctoral Dissertation at the University of Agder 83
ISSN: 1504-9272
ISBN: 978-82-7117-762-1

Printed by the Printing Office, University of Agder
Kristiansand

Acknowledgements

I would first like to thank my supervisor, Vladimir Oleshchuk, for encouraging me to pursue a PhD, and also for excellent supervision. Ole-Christoffer Granmo has been helpful in his role as second supervisor, especially when it comes to reviewing and discussing material related to learning systems and privacy metrics. Also thanks to Telenor for providing the PhD grant and Geir Køien who has been helpful in following up on contract work and administrative procedures with Telenor.

Also thanks to the faculty management and research administration, in particular faculty director Magne Aasheim Knudsen and Søren Kragholm, for supporting the PRECYSE project and Antonio Marqués and Santiago Caçeres at ETRA in Spain for taking UIA on board the project. Without the practical use cases, requirements and scenarios developed during the PRECYSE project, the proposed solution would not have been conceivable. Also a great thanks to everybody in the PRECYSE team - you are a great team to work with!

At last, but not least: Thanks to my wife Lynn and parents for supporting me and my son Oskar for cheering me up and giving me daily amounts of play and exercise.

This dissertation is supported by:

Telenor Research & Innovation under the contract DR-2009-1;

The PRECYSE project, funded by the European Commission under FP7 with contract number FP7-SEC-2012-1-285181 (www.precyse.eu);

University of Agder.

Oppsummering

Denne doktorgradsavhandlingen undersøker to nødvendige virkemidler som trengs for å bygge personvernforbedrede overvåkningssystemer for datanettverk: en teknologi for håndheving av personvern- eller konfidensialitetspolicyer; samt metrikker som måler lekkasje av privat eller konfidensiell informasjon for å verifisere og forbedre disse policyene. Håndhevingsmekanismen er basert på finmasket tilgangskontroll og reverserbar anonymisering av XML data for å begrense eller kontrollere tilgangen til sensitiv informasjon fra overvåkningssystemene.

Metrikkene kan brukes til å støtte en kontinuerlig forbedringsprosess, både for å kvantifisere lekkasjer av privat eller konfidensiell informasjon, og for å lokalisere hvor disse er samt foreslå forbedringstiltak. De planlagte tiltakene kan deretter bli gjennomført, enten ved å benytte en reverserbar anonymiseringspolicy eller ved å fjerne årsaken til informasjonslekkasjene. Personvernmetrikkene kan deretter verifisere at tiltakene virker som planlagt. Vesentlige avvik fra forventet lekkasje kan deretter brukes til å sette i gang nye forbedringstiltak. De viktigste resultatene i avhandlingen er:

- en personvernlekkasjemetrikk basert på standardavviket av entropien til gitte data (for eksempel IDS alarmer), som måler hvor mye sensitiv informasjon som lekker ut og hvor disse lekkasjene er;
- en tjenesteformidler (proxy) som tilbyr policybasert reverserbar anonymisering av informasjon i XML-baserte webtjenester. Denne løsningen støtter også flernivå sikkerhet, slik at bare autoriserte interessehavere kan få tilgang til den sensitive informasjonen;
- En metodikk som kombinerer personvernlekkasjemetrikkene med reverserbar anonymisering for å støtte en kontinuerlig forbedringsprosess med redusert lekkasje av privat eller konfidensiell informasjon over tid.

Dette kan brukes til å forbedre håndteringen av personsensitiv eller konfidensiell informasjon i tilfeller der administrerte sikkerhetstjenester har blitt tjenesteutsatt til partnere man kun har delvis tillit til, for eksempel for tjenesteutsatt sikkerhetsovervåking av helseinstitusjoner eller kritiske infrastrukturer. Løsningen er basert på relevante standarder som sørger for kompatibilitet med eksisterende innbruddsdeteksjonssystemer og alarmdatabaser.

Summary

This PhD dissertation investigates two necessary means that are required for building privacy-enhanced network monitoring systems: a policy-based privacy or confidentiality enforcement technology; and metrics measuring leakage of private or confidential information to verify and improve these policies. The privacy enforcement mechanism is based on fine-grained access control and reversible anonymisation of XML data to limit or control access to sensitive information from the monitoring systems.

The metrics can be used to support a continuous improvement process, by quantifying leakages of private or confidential information, locating where they are, and proposing how these leakages can be mitigated. The planned actions can be enforced by applying a reversible anonymisation policy, or by removing the source of the information leakages. The metrics can subsequently verify that the planned privacy enforcement scheme works as intended. Any significant deviations from the expected information leakage can be used to trigger further improvement actions. The most significant results from the dissertation are:

- a privacy leakage metric based on the entropy standard deviation of given data (for example IDS alarms), which measures how much sensitive information that is leaking and where these leakages occur;
- a proxy offering policy-based reversible anonymisation of information in XML-based web services. The solution supports multi-level security, so that only authorised stakeholders can get access to sensitive information;
- a methodology which combines privacy metrics with the reversible anonymisation scheme to support a continuous improvement process with reduced leakage of private or confidential information over time.

This can be used to improve management of private or confidential information where managed security services have been outsourced to semi-trusted parties, for example for outsourced managed security services monitoring health institutions or critical infrastructures. The solution is based on relevant standards to ensure backwards compatibility with existing intrusion detection systems and alarm databases.

Contents

List of Figures

List of Tables

Abbreviations

API	Application Programming Interface
APT	Advanced Persistent Threat
CERT	Computer Emergency Response Team
CSIRT	Computer Security Incident Response Team
DDoS	Distributed Denial of Service attack
DOM	Document Object Model
DoS	Denial of Service
DPI	Deep Packet Inspection
EM	Expectation Maximisation
EPAL	Enterprise Privacy Authorisation Language
ESB	Enterprise Service Bus
HR	Human Resources
IDMEF	Intrusion Detection Message Exchange Format
IDS	Intrusion Detection System
IDXP	Intrusion Detection eXchange Protocol
IETF	Internet Engineering Task Force
IODEF	Incident Object Description Exchange Format
IPFIX	IP Flow Information Export

IPS	Intrusion Prevention System
ISM	Information Security Management
JVM	Java Virtual Machine
LRU	Least Recently Used
MML	Minimum Message Length
MSS	Managed Security Service
OASIS	Organization for the Advancement of Structured Information Standards
OVAL	Open Vulnerability and Assessment Language
P3P	Platform for Privacy Preferences
PAP	Policy Administration Point
PbD	Privacy by Design
PDCA	Plan Do Check Act
PDF	Probability Density Function
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PII	Personally Identifiable Information
PIP	Policy Information Point
PLC	Programmable Logic Controller
PPS	Probability Proportional to Size sampling
RBAC	Role-Based Access Control
RID	Real-time Inter-network Defence protocol
SAML	Security Assertion Markup Language

SIEM Security Incident and Event Management system

SOA Service Oriented Architecture

SOAP Simple Object Access Protocol

SOC Security Operations Centre

STIX Structured Threat Information Expression

TAXII Trusted Automated eXchange of Indicator Information

Teleological The philosophical doctrine that final causes, design, and purpose exists in nature.

TM Time Machine

VPN Virtual Private Network

VTD Virtual Token Descriptor

XACML eXtensible Access Control Markup Language

XML eXtensible Markup Language

Part I

Introduction and Background

Part I is the introduction to the dissertation which amongst others contains the problem description and an overview over the dissertation. The next chapter contains an analysis of ethical, economic and technical issues and aspects of managed security services and proposes a set of requirements based on a set of use cases for privacy-enhanced network monitoring systems.

Chapter 1

Introduction

1.1 Background and motivation

This dissertation aims at investigating how privacy-enhanced network monitoring can be implemented, with a particular focus on privacy enhanced intrusion detection systems. The main research challenge is handling the delicate tradeoff between efficient monitoring of computer networks for signs of malicious activities, and at the same time maintain socially and legally acceptable solutions for handling data privacy and confidentiality. The objective is to avoid an unnecessary amount of sensitive information flows towards semi-trusted third party organisations or company internal employees performing network monitoring. The dissertation suggests several approaches to reduce the impact managed security services have on privacy. One important factor is being able to measure the privacy leakage of IDS rules. If it is possible to measure which IDS rules that are good or bad from a privacy perspective, then the Managed Security Service (MSS) provider will be able to tune the IDS rule set in order to reduce investigation of sensitive information according to the *need-to-know* principle.

Current Intrusion Detection Systems (IDS) do in general not support enforcement of privacy policies, which means that there is a significant chance of human error or that individuals are going beyond their call of duty when it comes to monitoring sensitive material. Another risk that needs to be taken seriously, is the risk of radicalisation of trusted personnel. This means that it is not sufficient to use trust and risk of legal liabilities, like confidentiality agreements, as the only means to ensure proper handling of private or confidential information. Sensitive information may then be revealed either for own profit or for political reasons. A confidentiality agreement is therefore a necessary but not sufficient requirement to avoid leakage of private or confidential information. This is the main motivation for doing research on technological measures which can improve enforcement of privacy policies in IDS alarms.

1.2 Problem Statement

The dissertation focuses on reducing privacy leakage in IDS alarms, primarily from network-based and signature-based IDS, although some of the methods and and techniques also may work to detect leakage of sensitive information in alarms from host-based IDS, firewall or Anti-Virus detection software. The dissertation investigates two main topics in this respect:

- Privacy leakage metrics that can detect where leakage of private or confidential information in IDS alarms may occur;
- Enforcement of privacy policies that in combination with the privacy metrics can reduce the possible privacy leakages in IDS alarms.

It is furthermore assumed that the network monitoring using IDS is outsourced to a Managed Security Service (MSS) provider. Details on privacy related problems for data forensic interfaces, like time machines and similar techniques [88], is considered outside the scope of the dissertation[1].

1.3 Adversary Model

By privacy leakage we here mean leakage of information that according to a privacy impact assessment has been found to be problematic, either because it

[1] The privacy enforcement method and privacy leakage metrics proposed in the dissertation are however general and can also be applied for data forensic interfaces and other monitoring techniques. The details of how this can be implemented is however left as future work.

reveals Personally Identifiable Information or because it reveals information that is confidential.

The dissertation aims at detecting and mitigating both accidental and deliberate privacy attacks assuming an outsourced service. There are two privacy adversaries that are considered:

1. Insider attacks from the service provider or other trusted parties: The privacy metrics and privacy preserving methods should be made incentive compatible[2], so that an insider adversary cannot shirk or cheat from a privacy perspective by doing technical adaptations to improve the privacy rating. The privacy enforcement methods should furthermore only provide access to private or confidential information according to a strict interpretation of need, and access to such information should be transparent, accountable and non-reputable according to the Privacy by Design criteria [25].

2. External attackers are assumed to have Dolev-Yao intruder capabilities [39]. This means that they can overhear, intercept and generate any message, and are only limited by the cryptographic methods used. The objective for external attackers may be to abuse the service, for example by triggering anonymisation for a security attack to go undetected.

1.4 Objectives and Scope

1.4.1 Scope of Dissertation

This dissertation is written as a book (monograph), and contains mostly material that either is published or that has been submitted to journals. I am the main contributor of all papers included in the dissertation. The tables below show a detailed survey of which material that has been published, modified or is unpublished during the PhD study.

[2] Incentive compatibility – a characteristic of mechanisms whereby each agent knows that his best strategy is to follow the rules, no matter what the other agents will do [79].

Part I Introduction and Background

Chapter	Reason
Chapter 1 Introduction *(unpublished)*	Dissertation internal
Chapter 2 A Roadmap Towards Improving Managed Security Services from a Privacy Perspective is based on *Ulltveit-Moe N. A Roadmap Towards Improving Managed Security Services from a Privacy Perspective* (submitted).	Contains the foundations and requirements for privacy metrics presented in chapter 6. The main change from the submitted version, is that section 2.12 is changed from focusing on technical solutions to system use cases and requirements. This is to maintain causality between the chapters of the dissertation, since the original paper described some of the solutions that will be developed in later chapters.

Part II Privacy Enforcement for Intrusion Detection Systems

Paper	Reason
Chapter 3 Two Tiered Privacy Enhanced IDS Architecture is based on *Ulltveit-Moe, N. and Oleshchuk, V., "Two tiered privacy enhanced intrusion detection system architecture", in Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2009. IDAACS 2009. IEEE (2009), pp. 8-14.*	Introduces an architecture for privacy-enhanced IDS. No significant changes from published version.

Chapter 4 Decision-cache Based XACML Authorisation and Anonymisation for XML Documents is based on *Nils Ulltveit-Moe N., Vladimir A. Oleshchuk V.: Decision-cache based XACML authorisation and anonymisation for XML documents. Computer Standards & Interfaces 34(6): 527-534 (2012)*	This chapter lays the foundations for the anonymisation scheme used in the dissertation. The introduction is cut down to avoid redundancy. The notation is modified to be consistent between Chapter 4 and Chapter 5. The default block marker used when anonymising is changed to ’X’, since the space character will not work for the reversible scheme. Added linear regression of performance data to improve figure. No other significant changes from published version.
Chapter 5 is based on *Ulltveit-Moe N. and Oleshchuk V. “*A Novel Policy-driven Reversible Anonymisation Scheme for XML-based Services*” (submitted).*	Extends the solution in the previous chapter to support reversible anonymisation. The chapter is an extended version of the journal paper which shows XACML implementation details.

Part III Privacy Leakage Detection and Avoidance

Chapter	Reason/changes
Chapter 6 Measuring Privacy Leakage for IDS Rules is based on *Ulltveit-Moe, N. and Oleshchuk, V. “Measuring Privacy Leakage for IDS Rules” (submitted).*	This contains the privacy metric definition. No significant changes from submitted version.

Chapter 7 Metrics-supported Privacy Enforcement is based on *Ulltveit-Moe N. and Oleshchuk V. "Metrics-supported Privacy Enforcement" (submitted).*	This paper connects the privacy metrics with privacy enforcement solutions. The chapter is a cut-down version of the journal paper to avoid overlap with chapter 6 and the adversary model and also to focus less on big data analytics. It does in addition contain a figure illustrating the Plan Do Check Act process that did not fit into the paper.

Part IV The Way Ahead

Paper	Reason
Chapter 8 contains conclusion and future research directions. Some of the information in this Chapter is based on *Ulltveit-Moe N., Gjøsæter T., Assev S., Køien G. M. and Oleshchuk V. Privacy Handling for Critical Information Infrastructures, in the proceedings of the Industrial Informatics (INDIN) 2013 11th IEEE conference, (IEEE 2013), pp 688-694.*	Concludes the dissertation and outlines future work.

Appendix A describes the technical XACML solution for supporting metrics-based privacy enforcement.	Not published, since it heavily relies on the information in Chapter 7. The appendix probably has limited research value, but may be useful for future development or standardisation.

Not Included in the Dissertation

Paper	Reason
Ulltveit-Moe, N. and Oleshchuk, V., "PRIvacy LEakage Methodology (PRILE) for IDS Rules", in Bezzi, M. and Duquenoy, P. and Fischer-Hübner, S. and Hansen, M. and Zhang, G., ed., Privacy and Identity Management for Life vol. 320, (Springer Boston, 2010), pp. 213-225.	This was the first attempt at defining a privacy metric. This metric was useful during initial data analysis to understand the problem domain, and it may be useful for analysing individual IDS rules in some cases. The metric is however flawed by normalising data relative to the information passing through each IDS rule, which causes the metric to not be incentive compatible also not aggregatable. The metric is also not practical, since it relies on manual data classification. This metric is superseded by the metric in Chapter 6.

Ulltveit-Moe, N. and Oleshchuk, V., "A Composite Privacy Leakage Indicator", Wireless Personal Communications 61 (2011), pp. 511–526.	This paper is based on the flawed metric in the previous paper, and is therefore not included. It is superseded by the metric in Chapter 6.
Nils Ulltveit-Moe and Vladimir Oleshchuk, "Privacy Violation Classification of Snort Ruleset", in 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing (Pisa, Italy , 2010), pp. 654–658.	This paper is not included, since it does not contain any substantial new knowledge needed for the development of privacy metrics.
Ulltveit-Moe, Nils and Oleshchuk, Vladimir A. and Køien, Geir M., "Location-Aware Mobile Intrusion Detection with Enhanced Privacy in a 5G Context", Wireless Personal Communications 57, 3 (2010), pp. 317–338.	Not included, since location-based IDS is considered outside the core objectives of the dissertation.
Ulltveit-Moe, Nils and Oleshchuk, Vladimir, "Mobile Security with Location-Aware Role-Based Access Control", in Prasad, Ramjee et al., Security and Privacy in Mobile Information and Communication Systems vol. 94, (Springer Berlin Heidelberg, 2012), pp. 172–183.	Not included, since location-based IDS is considered outside the core objectives of the dissertation.

Ulltveit-Moe, Nils and Oleshchuk, Vladimir Mobile Internet Security Enforcing Mobile Security with Location-aware RBAC, Wiley Security and Communication Networks (In press).	Not included, since location-based IDS is considered outside the core objectives of the dissertation. Extended journal version of paper above.

1.4.2 Privacy Objectives

A privacy-enhanced intrusion detection system should as far as possible fulfill a set of basic privacy objectives, like for example the seven foundational Privacy by Design (PbD) principles [25]:

1. Proactive not reactive - the system should aim at preventing privacy-invasive events or risks before they happen.

2. Privacy as the default - ensure that personal data automatically is protected.

3. Privacy embedded into the design - not bolted on as an add-on.

4. Positive sum - integrate privacy enhancing technologies that both support the privacy and security objectives creating a win-win situation.

5. Life cycle protection - ensure that sensitive data is protected from the data is created and until it can be securely destroyed.

6. Visibility/transparency - ensure that operations on privacy sensitive information is traceable.

7. Respect for users - keep the interests of the individual uppermost - offer appropriate notice and empowering user-friendly options.

The dissertation will aim at fulfilling as many as possible of these objectives. It must however be noted that an intrusion detection system normally is not aware of who a given user is. It is not a system that is used by the user, and where the user decides to store potentially sensitive user preferences. This means that storing user profiles, and allowing users to selectively retrieve information about them, by default will not be supported, since the IDS typically has no idea about the connection between users and data. This means that it for example does not make sense to require informed consent before intrusion detection can be applied for a given user. However, data networks that are being monitored for signs of intrusions should state this clearly, so that users using services on these networks are aware of this. Given these underlying restrictions of intrusion detection systems, the objective is to fulfill the privacy by design objectives, or at least outline how this can be done.

1.4.3 Technological Objectives

The technological objective of the dissertation is to design privacy-friendly technologies that can be useful for reducing the privacy invasiveness of network monitoring technologies like intrusion detection systems. Ian Goldberg has proposed four basic requirements that privacy enhancing techniques should adhere to, in order to be useful in practice [3]:

Usability the proposed methods/technologies must be easy to use for the Managed Security Service providers.

Deployability the proposed methods/technologies must be easy to deploy, also in existing networks/infrastructures, and must be possible to integrate into existing network monitoring technologies.

Effectiveness the proposed methods/technologies must have a potential to provide a significant and quantifiable reduction in privacy leakage for the intrusion detection systems involved.

Robustness the proposed methods must be robust, for example against misuse, in order to avoid that the organisations can improve the privacy leakage measurements only by performing technical adaptations (for example by changing the IDS rule set without any significant change in what is being monitored).

These requirements come in addition to the Privacy by Design criteria, in order to ensure that methods and techniques for enhancing privacy are easy to operationalise. The proposed scheme should for example be adaptable,

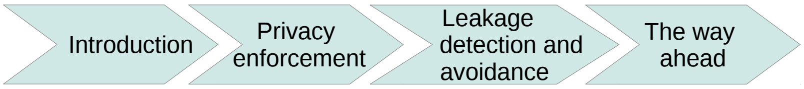


Figure 1.5.1: Structure of the dissertation.

so that anonymisation can be implemented based on a risk assessment and operative needs.

There are also use cases where security requirements demand that certain stakeholders, for example security analysts, have access to necessary sensitive information to do attack analysis. Privacy can however be protected and controlled also in these cases, by limiting who can access this sensitive information according to what is needed from an operational perspective, as well as by ensuring transparency on who can and who have accessed private or confidential information. In the end, it will be an operative decision by the MSS provider together with a data controller or a quality certification organisation to agree on defining an enforceable policy for this.

1.5 How the Dissertation is Organised

The dissertation is organised into four main parts, as illustrated in Figure 1.5.1: *Part I introduction and background, Part II Privacy Enforcement for Intrusion Detection Systems, Part III Privacy Leakage Detection and Avoidance and Part IV The Way Ahead.* The dissertation aims at keeping the notation consistent within each part and chapter, and also aims to keep shared notation consistent between parts and chapters. The dissertation does however in general operate with a part-local and chapter-local scope for most notation. This strategy is used to avoid running out of symbols, since parts II and III of the dissertation rely on an extensive notation related to cryptography and information theory respectively.

Part I Introduction and Background

Part I gives a gentle introduction to the problem and discusses privacy requirements. Chapter 1 *Introduction* describes the problem to be solved and lists objectives and scope of the dissertation, as well as defining the adversary model. Chapter 2 *A Roadmap Towards Improving Managed Security Services from a Privacy Perspective* does an analysis of ethical, economic and technical aspects of managed security services and proposes a set of requirements based on some use cases for privacy-enhanced network monitoring systems.

Part II Privacy Enforcement for Intrusion Detection Systems

Part II discusses technical controls that can be used to enforce privacy-enhanced IDS. Chapter 3 *Two Tiered Privacy Enhanced IDS Architecture* presents an architecture for privacy enhanced IDS, where the operation is split into an outsourced first-line service which operates in privacy preserving mode, and a second-line service that does attack analysis and has authorisation and security clearance to violate privacy and confidentiality if needed. It furthermore suggests how coarse grained anonymisation of sensitive information in IDS alerts can be performed by using an eXtensible Access Control Markup Language (XACML) policy.

Chapter 4 *Decision-cache Based XACML Authorisation and Anonymisation for XML Documents* extends the coarse-grained anonymisation solution in the previous chapter to a fine-grained solution for authorisation and anonymisation of sensitive information in XML messages. This is used for anonymisation of IDS alarms based on the Intrusion Detection Message Exchange Format (IDMEF) format. It furthermore uses an XACML decision cache to improve performance.

Chapter 5 *Reversible Anonymisation for Intrusion Detection Systems* extends the Decision-Cache based anonymiser to support reversible anonymisation of sensitive information in IDS alarms with support for multi-level security. The privacy policy scheme is furthermore extended to support anonymity by default, key sharing and secure data destruction.

Part III Privacy Leakage Detection and Avoidance

Part III analyses privacy metrics, and proposes how these can be integrated with the privacy enforcement scheme discussed in Part II.

Chapter 6 *Measuring Privacy Leakage for IDS Rules* proposes a theoretical model of privacy leakage in IDS rules based on quantitative information flow analysis. The proposed metric is based on the standard deviation of entropy, with a length correction to avoid being incentive incompatible. The metric is verified both in simulations as well as using real IDS alarms. The chapter also outlines how the metric can be generalised to other IDS techniques than signature-based IDS.

Chapter 7 *Metrics-based Privacy Enforcement for IDS Alarms* combines the privacy enforcement mechanism in Part II with the privacy leakage metric in Chapter 6. It describes a methodology for privacy leakage detection and avoidance which allows a continuous improvement process from a privacy and confidentiality perspective. It shows how the privacy metrics in combination with the privacy enforcement techniques can be used to implement metrics-supported privacy enforcement policies, and also discusses possible vulnerabilities of such a scheme. The chapter furthermore proposes a detailed entropy map indicating where (in which octets) of the elements or attributes of an IDS alarm privacy leakages occur.

Part IV The Way Ahead

Chapter 8 *Discussion, Conclusion and Future Work* concludes the dissertation. First, it discusses to what degree the proposed solution supports the Privacy by Design objectives in Section 1.4.2. Then the possible impact that the proposed solution may have is discussed, including to what degree it fulfills the the technological objectives in Section 1.4.3. This leads up to the general conclusion, which amongst others concludes on how well the proposed solutions supports the problem statement. The last section outlines future work and discusses possible starting points for further research.

Chapter 2

A Roadmap Towards Improving Managed Security Services from a Privacy Perspective

Increased ethical awareness and improved guidelines, methodologies and tools are needed for handling private or confidential information for outsourced managed security services. The current situation is that some techniques, for example intrusion detection systems, may be too privacy invasive. At the same time, investigation of transnational on-line crime is impeded by a proliferation of attacks, lack of electronic evidence and legal hindrances across national borders. This means that new strategies are needed that facilitate increased cooperation between organisations on attack detection, at the same time as private and confidential information must be respected. This chapter proposes a roadmap, requirements and use cases for how these privacy issues can be controlled using a continuous improvement process. Organisations performing monitoring of computer networks for signs of attacks can for example be expected to benefit from using quantifiable privacy and security metrics as part of a service level agreement. It is furthermore analysed whether automatic blocking of malicious traffic is better than surveillance of such activities. Last, the chapter discusses how incentive compatible contractual means can be used to reduce the moral hazard both from a privacy and security perspective for outsourced managed security services.

2.1 Introduction

Globalised organised cyber-crime is a serious and rapidly increasing problem in today's society. It can ultimately threaten the IT-infrastructure of countries [9]. In addition, Advanced Persistent Threats (APTs), like the Stuxnet and Duku worms targeting critical infrastructures are being developed by governmental agencies as part of a cyber warfare strategy [34]. The conviction rates are low for cyber crime, because it is hard to get evidence that can be traced back to the offender. Another reason is that it is difficult and expensive to investigate crime involving several countries with different legislation [75].

Various attack detection techniques, like Intrusion Detection Systems (IDS), spamfilters or anti-virus are being used to detect, investigate and prevent cyber-crime both in the private and public sector. It is legal to perform monitoring of computer networks and hosts using such potentially invasive technologies in most European countries, as long as the *purpose* with the monitoring is to detect cyber-attacks. There are for example explicit exceptions for measures related to detecting cyber-attacks in the EC communications directive [46]. This may nevertheless be problematic from a privacy or confidentiality perspective, because the *effect* of such monitoring is largely unknown. This means that better methods are needed to protect private or confidential information, at the same time as better techniques are required for ensuring transparency on use of such information. This can for example be in the form of secure logging of access to private or confidential information.

This chapter discusses ethical problems related to Managed Security Services (MSS) - network security services that are outsourced to a service provider [70]. The main focus in this chapter is on attack detection techniques like Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS), with a particular focus on privacy. Based on this ethical discourse, a roadmap is proposed on how to improve handling of private and confidential information for such systems.

This chapter is organised as follows: The next section gives an introduction to what intrusion detection and prevention systems are. The effect of outsourcing security monitoring is discussed in section 2.3. Section 2.4 gives a definition of privacy, and section 2.5 discusses how privacy can be improved for IDS from a high-level perspective. Section 2.6 discusses sources of information leakages in intrusion detection and prevention systems. Section 2.7 discusses network monitoring from an ethical and human rights perspective, and section 2.8 discusses how the arms race between adversaries and security interests impact these monitoring technologies. Section 2.9 discusses the effects of privacy and security interests normally competing for the same funding, and section 2.10 discusses privacy valuation. Section 2.11 discusses advantages and disadvantages with automatic attack prevention (IPS) compared to attack detection (IDS) from an ethical perspective. Section 2.12 proposes a roadmap towards improved privacy for managed security services based on ethical principles, technical privacy enforcement mechanisms, privacy metrics and best practices within information security management. Finally, section 2.13 concludes the chapter and outlines future work.

2.2 What Are Intrusion Detection and Prevention Systems?

Network based Intrusion Detection System (IDS) is the Internet equivalent of a burglar alarm which monitors all packets passing through a router, switch or firewall. This monitoring is typically performed using *deep packet inspection* (DPI), which means that the following data can be investigated: packet header information, e.g. IP addresses and ports; payload in each data packet; reassembled streams of data spanning several data packets; and entire communication sessions between a client machine and a server.

This means that all communication performed between a client application, for example an email reader and the corresponding email server, in principle can be intercepted and investigated in detail by an IDS, as long as the communication sessions are not encrypted. However encrypted data can also in special cases be monitored, for example if the monitoring is performed after the decryption function or if the IDS is trusted with the secret key necessary to decrypt encrypted data sessions.

The two main types of IDS are signature-based IDS, which relies on matching known attack patterns in the data traffic, and anomaly-based IDS, which interprets statistical anomalies in the data traffic as possible attack activities. Intrusion Prevention Systems (IPS) extend IDS with functionality to automatically block traffic from senders that triggers an IDS rule or traffic anomaly.

The market leaders/challengers for IDS/IPS devices: Cisco, HP, IBM, Juniper, McAfee and Sourcefire [56], all rely on using DPI, based on a combination of signature-based and anomaly-based detection techniques. This avoids any security blindspots that may occur using either technology [28, 84, 18, 149, 89]. IDS focuses on identifying possible incidents, and supports incident response efforts to identify successful compromise of a system due to an adversary exploiting a system vulnerability [114]. Typical use of IPS technology involves acting as a shield protecting vulnerable machines from known attacks [56]. This is in particular important for Critical Infrastructures, which may have long patch latencies due to strict safety requirements on testing of patches before deployment. Other uses of IDS include identifying security policy problems, documenting the existing threat to an organisation and deterring individuals from violating security policies [114].

The privacy concern related to IDS or IPS comes both from policy-based IDS/IPS rules and from false alarms or anomaly patterns in attack detecting rules which may leak Personally Identifiable Information (PII) or other confidential information. Privacy leakages may also occur due to activities caused by malicious actors, for example computers compromised with malicious software or web bugs that reveal sensitive information about compromised users. The latter may cause privacy leakages both towards the malicious actors, which should be detected and deterred by IDS/IPS, as well as to MSS providers monitoring the IDS alarms. Privacy leakages can be controlled using a combination of privacy leakage metrics and privacy enforcement mechanisms. Access to such information should furthermore be logged, to reduce access to those that strictly need to know it, as well as maintaining transparency and traceability on the MSS operation.

2.3 The Effect of Outsourcing Security Monitoring

Outsourcing security monitoring to MSS providers has gained popularity for two main reasons. First, the cost of providing 24x7 monitoring is only a fraction of what such monitoring would cost in-house [38]. Second, MSS providers have in general got more experience in handling security incidents and more updated monitoring technology, by specialising in this area, than the average customer. A large client base also contributes to service quality improvements because an MSS provider, monitoring a large set of networks, easier can correlate attacks and identify new attack patterns. They can also share information about attacks and attack mitigation strategies between its customers, which is one of the factors that have been shown to reduce the risk of attacks from adversaries [115]. One concern firms have when considering to outsource security services, is that the MSS provider may shirk (avoid doing its duties) secretly to increase profits. In economics this behaviour is commonly referred to as the Moral Hazard problem. The optimal way to avoid such behavior on a contractual basis is to use a performance-based contract, however the degree of performance dependence may decrease if the reputation effect becomes significant [38].

It should be noted that the Moral Hazard problem not only is applicable to the security of the monitored data. It is also applicable to the privacy and confidentiality of the monitored data. Both in the sense of handling more private and confidential information than strictly necessary and in the sense of potentially leaking or abusing private or confidential information. This does in the end mean that a principal (here the customer of security services) should require that both the *security* and *privacy* performance for outsourced MSS should be part of a performance-based contract with the MSS provider. This means that the MSS provider should be *accountable* for both the privacy and security part of the operation which means that suitable performance metrics and activity logging procedures are needed for both privacy and security, so that the performance in these areas can be reported and audited if necessary. This is in line with the 6. foundational Privacy by Design principle, that the privacy-enhanced design must ensure transparency [25].

2.4 What Is Privacy?

Privacy is a broad concept that can have different meaning in different contexts. Warren and Brandeis early on defined privacy from a legal perspective as *the right to be let alone [144].* Other definitions focus more on privacy as an intellectual property from a utility perspective, where the data owner should be ensured *self determinism* about private data [112]. This amongst others means that the data owner must be able to give and revoke consent to access private data [25]. This has for example lead to actuarial models that aim at estimating the perceived cost of privacy leakage for insurance contracts [60].

Information theorists provide a more technical definition of privacy and often define privacy as equivocation (level of ambiguity). This definition is based on the observation that some level of privacy and anonymity can be ensured by requiring that private information is hidden in a sufficiently large crowd of other information, so that the data owner cannot easily be identified. Equivocational privacy metrics are for example based on entropy [113, 29], or they directly specify a level of equivocation like k-anonymity [111, 27, 126] or l-diversity, which in addition to level of equivocation considers the diversity of the data [86].

From an information theoretic viewpoint, privacy leakage can be modelled based on the assumption that the utility of the data is inversely proportional to the level of perturbation of the data and the level of privacy can be quantified as the level of equivocation [113].

The dissertation considers privacy or confidentiality as an intellectual property that has a subjective value by the information owner, and therefore should be protected from unnecessary disclosure. This furthermore means that access to such information should be transparent and auditable.

2.5 How Can Privacy be Improved?

One way to illustrate how technology can improve privacy from a high level perspective is airport security. Many are willing to trade some convenience and privacy for added security. It is therefore accepted in our society that all passengers undergo privacy-invasive security control checks when traveling by airplanes to increase the perceived safety. The privacy-invasive security controls aim at reducing the possibility that adversaries, like terrorists or psychologically unstable persons, bring weapons, explosives or other dangerous items on board the airplane.

There has been quite extensive research on more efficient ways to detect hidden weapons on people. One efficient technology, that recently has been deployed, is backscatter X-ray scanners [24]. These scanners expose the person to be checked with small amounts of X-ray radiation, and use the backscatter X-rays to produce photo-quality images that can see through clothes. This technology is used as an alternative to personal searches, since it easily can reveal hidden weapons. If a suspicious item is detected, then the security officer will perform a manual search to verify what the suspicious item is.

This technology causes a privacy concern, since it essentially shows a naked picture of the person being scanned. Privacy enhancing technologies have therefore been implemented to deal with this problem. The techniques include using blurred pictures or stylistic images, emphasising items that are not considered normal body features. Such techniques mean that the privacy of all people who are not being suspected of carrying illegal items need not be violated, which limits the amount and degree of privacy violations.

The Internet analogy of this is surveillance techniques like IDS using deep packet inspection. This means that the MSS provider effectively can see any cleartext traffic that triggers IDS alarms between a customer performing a service on the Internet and the service provider. There may therefore be a conflict between the privacy and security objectives[1] for managed security services. However, a larger problem may be the lack of transparency on what is being monitored, and why. My experience is that IDS rule sets being implemented are typically considered company secrets by MSS providers - partly because of the risk that an attacker may abuse this information to attack customers of the MSS, and partly because some rules may implement possibly privacy invasive IT monitoring policies, for example monitoring use of peer-to-peer traffic, if the IT policy disallows this. The network owner may

[1] There will also be synergies between privacy enhancing technologies and security, as will be discussed later. Aiming for such synergies is recommended by the 4. Privacy by Default principle, which states that one should aim for a win-win situation between privacy and security [25].

not want to reveal such monitoring practices, since such information may be considered sensitive from a business or reputation perspective. An important principle should be that the monitoring invades privacy and confidentiality as little as possible for normal, unsuspicious traffic. However, just like in the airport example, a more thorough investigation will be required if suspicious Internet traffic is detected, to verify whether the data traffic is hostile or not.

The decisions and actions security analysts perform should be logged, regardless of whether the analyst decides to investigate an event in detail or not, since this will provide transparency on what is being investigated and why, both from a security and privacy perspective. Such transparency can be expected to be instrumental in improving the MSS operation both from a work efficiency, attack detection efficiency and privacy impact perspective, since it would allow identifying and putting effort into mitigating bottlenecks, blindspots or overly privacy invasive sides of the operation.

2.6 Information Leakages from Intrusion Detection and Prevention Systems

The first question one perhaps should ask, is whether there really is any significant leakage of private or confidential information in IDS alarms? The market leaders claim that false alarms is not a problem for a properly managed IDS/IPS in their technical documentation. However, a recent comparative analysis of commercial IDS and IPS shows that more than 92.85% of all IDS alarms on a campus network from a testbed of seven different commercial IPS/IDS products, tested over a period of two years, are false alarms [63]. Other studies have also indicated that network monitoring technologies create a significant amount of false alarms [5]. Some of the reason for this, is applications that do not follow the protocol specifications [63]. Furthermore, around 91% of the false alarms were not related to security issues, but management policies, for example that IDS rules were set up to identify peer-to-peer (P2P) traffic, that was not allowed according to the IT policy [63].

Our own investigations show that some IDS rules may be overly broad, in an attempt to generalise the rule to match different attack vector variants [134]. A side-effect of this, is a significant amount of false alarms which may contain private or confidential information. Another problem is that Internet applications do not follow the standards, and therefore may trigger false alarms, for example on web-based IDS rules that check for standards conformity [63].

Another similar area is IDS rules for identifying web bugs. These web bugs may in themselves be a risk for privacy and data confidentiality. However, in this case, the good intention of security monitoring may be its own worst enemy, because detecting privacy leaking web bugs causes a significant privacy leakage in itself. For IDS rules detecting web bugs, it is only interesting from a security perspective to detect the presence of such potentially malicious browser plugins. It should not in general be necessary to view the privacy-leaking payload, addresses or advertisements that these plugins cause, perhaps apart from a limited analysis of attack behaviour for the web bugs.

I have had discussions with practitioners in this area, and one of the ethical dilemmas they sometimes have, is when attack detection technologies trigger alarms on more than just real attacks. This is problematic from a privacy and confidentiality perspective, because this effectively is a leakage of private or confidential information that goes beyond the initial purpose the data were collected for - to detect cyber-attacks. An example scenario is when side-information from such monitoring activities by chance detects illegal or criminal activities that are not related to the core purpose of the monitoring technologies. This may put the Managed Security Service (MSS) providers in ethical dilemmas on how to handle this side-information, and from a law-enforcement perspective, this may not even be considered legal evidence, because the monitoring technologies have detected activities beyond the intended purpose of detecting cyber-attacks.

In these cases, the information being collected may reveal sensitive side-information about user behaviour, which should be restricted only to personnel performing investigations of illegal activities. Furthermore, transparency and nonrepudiation on such activities is important, which means that secure logging schemes are required to be able to prove who have accessed the given information when. Anonymisation, pseudonymisation or encryption are general techniques that can be used to reduce the privacy impact in cases like this, given that suitable metrics exist for identifying *where, what* and *how* *much* sensitive information that leaks. Sensitive parts of the data should then be anonymised, possibly using a reversible anonymisation scheme based on a combination of anonymisation and encryption, so that only authorised stakeholders can access this information.

2.7 Network Monitoring from a Utilitarian and Human Rights Perspective

Host and network-based attack detection techniques may cause an invasion of privacy that has the potential for harm. Companies therefore need to be able to justify the practice from an ethical perspective. The situation description in the previous sections show that *security* interests in some cases may come in conflict with other important human rights like *privacy, freedom of expression* and *the right to be presumed innocent until proven guilty*. There is furthermore a large uncertainty on *what* is being monitored by a MSS provider, which is problematic both from a privacy and security perspective.

However, there will also in many cases be synergies and common interests between privacy and security objectives, especially related to handling confidentiality, which is considered one of the core security objectives. Keeping corporate private and personally identifiable information confidential is beneficial both from a privacy and security perspective. This means that privacy or confidentiality enforcement techniques like anonymisation, pseudonymisation or encryption of sensitive data, as well as transparency on who have accessed these data, is beneficial both from a privacy and security perspective.

Teleological principles have an account of the good which is fully independent from the right, and a fully dependent theory of the right, as that which maximises the good [108]. The best known example of a Teleological principle is perhaps Utilitarianism introduced by Jeremy Bentham and John Stuart Mill [90, 15], which deems the moral action as the one that aims at maximising a given good.

It is for example possible to define the best moral choice of monitoring techniques as the choice that provides the highest employee utility in form of minimising losses due to non-work related Internet usage, security incidents and liabilities for example from downloading pirated software or music [26]. However, such a narrow definition of the best moral choice is problematic. The objective for the moral choices should rather be to reduce the harm for the society at large based on accepted standards like the Human Rights [139], than to focus on narrow definitions of the moral choice.

The general concern with monitoring without regard to privacy or other human rights, is that the on-line community would end up being like an electronic Panopticon where the inspector could see anything and the inspected would be aware of being monitored, potentially at any time, but not *when* they were being monitored [14]. This could in the worst case lead to a legalistic society where everything was dictated by law, and the inner freedom of ethical choice was reduced to little or nothing. Another way to describe it is as an Orwellian society that observed and controlled all on-line information [96]. Even though security may be improved, the overall utility would be lower, since other human rights like privacy and free speech would be reduced.

This is not only of theoretical interest. Mandatory surveillance and censorship of on-line behaviour is frequently performed in totalitarian regimes. One example is China, where Internet Service Providers are required to perform the monitoring of the citizens [143]. There is also a pressure towards widening the scope of on-line surveillance also in democratic regimes both in Europe and elsewhere in the world, for example to detect terrorism and certain types of crime [21].

Awareness of such monitoring causes self-censorship, which means that people may be afraid of telling the truth because of the risks of punishment or retaliation from parties responsible for the monitoring. Another risk is that the monitoring organisation may not act morally right and abuse acquired knowledge from private or confidential information. Corrupt insiders in the monitoring organisation may for example sell private or confidential information, extort the information owner or use the information for their own advantage [106]. A more recent concern is the risk of radicalisation by insiders that have access to private or confidential information, especially for critical infrastructures. One important principle here, is that the monitoring organisations need to be accountable and auditable for the operations they do on private or confidential information. This requires that techniques for ensuring *transparency* and *non-repudiation* are built into the monitoring technologies, so that the monitoring organisation cannot deny having processed given private or confidential information.

2.8 How Does the Arms Race between Adversaries and the Computer Security Industry Affect Privacy?

Intrusion detection systems are not without their problems. The monitoring is challenging because of the proliferation of new attack vectors and the use of obfuscation techniques, which make detection difficult [102]. The reason behind the proliferation of malicious software (malware) is partially due to malware creation kits [94]. These kits can create Trojans or entire phishing web sites that are intended to lure users to install malware on their computers. To avoid detection, malware uses techniques like cryptographic obfuscation and self-mutating code [85]. It is therefore easy for adversaries to create new attacks that typically will go undetected by anti-virus and IDS systems, so called zero-day exploits, which open up a window of opportunity for the adversary to attack the system. Zero-day vulnerabilities and exploits are considered valuable by cyber criminals, and are frequently traded on underground black markets [106]. A problematic aspect with these markets from an ethical perspective, is that not only cyber criminals, but also grey market security companies and governmental backed agencies participate in the trade of zero day exploits [59]. This means that cyber criminals, governmental services and grey market actors have an economic incentive to keep information about such vulnerabilities secret, instead of disclosing them which in general would provide better overall utility to the society.

In addition, backdoors and control channels to large bot-nets of compromised computers are increasingly using encrypted communication. Recent research for example indicates that the majority of the traffic from the Tor anonymiser network is bot-net related traffic, and hidden services provided by or via these compromised hosts [19]. It is not feasible to intercept the malicious communication in these cases, since the attacker is the only person who knows the decryption keys. Furthermore, attacks may be hard or impossible to trace due to no or limited retention of traffic data on a worldwide basis. This means that more comprehensive strategies are required for efficient monitoring of malicious activities in the future.

There is in other words an arms race between adversaries (malware producers, organised crime, governmental agencies and malicious hackers[2]) and the computer security industry. Traditionally, malicious hackers running botnets are opportunistic and will pick the targets that are easy to attack using any attack vectors that give a reasonable success rate [106]. The attackers can target software vulnerabilities using exploits, social vulnerabilities using Trojans, or both. The aim of malicious hackers is to a large extent monetary gain [52]. They are harvesting information from the hacked computers that can be used for financial fraud, identity theft, password logging or extortion.

The advent of Advanced Persistent Threats (APTs) changes this picture, since governmental agencies may use a large amount of resources for attacking a target critical infrastructure as part of a war or cyber intelligence strategy. It is much harder to protect oneself against such threats, since the attackers may have political reasons and sufficient funding for choosing a given target almost at any cost, and will therefore not necessarily go for an easier or less protected target, as cyber criminals frequently do. This means that better and more resilient methods are needed for protection of private or confidential information, as well as improved methods for protecting the availability and integrity of information infrastructures against cyber-attacks, than what currently is available. This can for example be done using a *defence-in-depth* strategy which assumes that more than one barrier need to break to access private or confidential information.

A recent trend that is expected to be the next major advance in attack detection, is to merge Big Data analytics with IDS/IPS, so that all communication to or from a company can be stored and investigated over a time span of months. An early example of this is the time machine [88], which

[2] The chapter uses the term malicious hacker to describe a cracker or black-hat hacker that maliciously attempts to break into systems.

works in a similar way as a "Personal Video Recorder" for network traffic, being able to store the initial part of all network sessions. Now more powerful cluster-based technologies like Apache Hadoop have been combined with IDS technologies to log all network traffic in real time, and at the same time perform near real-time attack analysis on this traffic. An example of this is PacketPIG, which is capable of storing traffic from a 100Mbit/s link in real-time for months on a 3Tb disk [13].

An advantage with such technologies, is that they allow detecting formerly unknown attacks (so-called zero-day attacks) after the attack has happened, by performing a retrospective IDS analysis on stored data, using new attack signatures identified after the data was logged. This improves the capabilities for performing data forensics significantly. However, these techniques also cause a concern both from a privacy and transparency perspective, since the operation on such big data is concealed in legal and commercial secrecy [72]. Another problem is that these techniques normally detect and not deter attacks, since they are based on data mining of past traffic. There is furthermore a lack of mechanisms for protecting the privacy and confidentiality when accessing these big data, as well as lack of logging mechanisms for ensuring transparency and non-reputability. Much of the reason for this, is that big data based security analysis still is in its infancy.

2.9 Privacy and Security Interests Compete on Funding

Privacy-enhancing technologies and privacy metrics should be used to allow security monitoring being performed as precisely as possible, in order to minimise the privacy and confidentiality impact of MSS operations. A challenge is that security monitoring needs to be implemented within a commercial organisation that mainly aims to maximise the profit for its owners. This means that a customer of a MSS provider will have a limited budget available for security investments. It has for example been suggested that only a fraction of the expected loss due to security breaches (max 37%) should be spent on security investments for a risk neutral firm [57]. This also means that privacy and security interests need to compete on the funding to implement the best possible security and privacy handling.

There are in other words practical limits for how much money and effort that a monitoring company should put into both security and privacy to improve the service. This means that solutions for enhancing the privacy should be readily available, affordable and easy to configure, preferably over existing services, to reduce the implementation costs for adding privacy and confidentiality protection. There is otherwise a risk that security interests may trump the privacy interests given a limited budget. This is at the moment a major hurdle, since technologies for privacy-enhanced security monitoring are not yet readily available. Parts II and III of the dissertation aim at describing how this deficiency can be mitigated.

The monitoring organisation may also see benefits in better privacy handling from an economic perspective, for example if improved handling of private or confidential information has side effects like reduced operating costs from handling fewer false alarms, or better protection of corporate secrets or personally identifiable information. In addition, improved privacy handling reduces the risk of liabilities from privacy leakages, and it will improve the trustworthiness for customers where privacy and confidentiality is paramount. One example of such customers is health institutions who, due to very strict privacy requirements, will not allow sensitive data to leave the corporate network.

Better privacy handling of services related to IDS data forensics and related protocols used for exchanging privacy or security-sensitive information is also in-line with the 4. foundational Privacy by Default principle [25], since integrating privacy enhancing technologies can create a win-win situation by supporting both the privacy and security objectives.

2.10 The Value of Privacy

A Utilitarian way to describe the optimal utility level has recently been proposed based on information theory [113]:

> *"For a data source with private and public data and desired utility*

> *level, maximum privacy for the private data is achieved by minimising the information disclosure rate sufficiently to satisfy the desired utility for the public data."*

This implies that private or confidential information is disseminated strictly on *a need to know* basis. An advantage with this approach, is that it may be possible to quantitatively analyse the optimal solution and compare how close a real solution is to the optimal one, given that some objective criteria or *metrics* for the information disclosure rate are identified. Privacy metrics like differential privacy have been proposed as a method to quantify the maximum privacy for a given level of utility for cases where sensitive data in databases need to be sanitised, for example by adding noise to blur the precision of given data, while still maintaining important statistical qualities, like the mean and standard deviation over a sufficiently large sample [40].

A disadvantage with this model is that it does not consider the semantics and therefore not the *value* of revealed private data. Some data are typically considered more sensitive and therefore also more valuable than other. Econometric or actuarial models have been suggested for modeling the cost of revealing data [60, 150]. The practical challenge with these economic models, is that it may be difficult to get representative cost distributions, since they are based on peoples' subjective value of private data.

2.10.1 Estimating the Value of Privacy

I did some preliminary experiments as part of my research where security analysts attempted to classify the privacy leakage of IDS alarms. They found it very difficult to do this. In many cases they found it hard to understand, or even purely hypothetical, that the sampled IDS alarms even would contain any significant information that was sensitive from a privacy or confidentiality perspective. The information they sampled, was after all open (i.e. not encrypted/protected) in their opinion. This could mean that security analysts are less privacy conscious than others, for example that they are blinded by operating routinely on sensitive information. It could also mean that there actually is less really private or confidential information in the IDS alarms than one should think.

The perceived value of private information is after all highly subjective [60], so it is not given that the valuation by security analysts, which are the only people that have security clearance to access the IDS alarms, would give a representative picture of the privacy leakage. In practice, the only stakeholders that can give the correct valuation of the privacy impact from IDS alarms, are the users themselves. And it is in most cases not trivial and also not desirable to connect the users to the underlying data from a privacy perspective.

One possible way to get around this problem, to get realistic measurements of the privacy impact may be the following: Assume that the data controller compiles a top ten list of the most privacy concerning information leakages, for example from a given web service. The data controller then needs to ask a representative random sample of users in an anonymous poll, presented during use of the service, what they think their privacy is worth in monetary value, given that a security company may see how they used a given set of web pages. The results from this poll could then be used to estimate a privacy impact factor as a random variable for each given information leakage.

It may however in practice not be feasible to do this, because it would be difficult to get permission to do such an experiment in an outsourced scenario where you would have to consider the business concerns of both the MSS provider and the service provider being monitored. It is hard enough to get consent from the MSS provider to do research on IDS data, and may be even harder to get consent from customers of MSS services, due to concerns that such a detailed poll would affect the reputation of the service being monitored. This means that it will be challenging at best, maybe not even possible, to get a representative cost distribution for the privacy impact of the data that seem to leak most information, not to mention getting a representative cost distribution for an entire IDS rule set consisting of several thousand rules. Furthermore, privacy valuation is very sensitive to how the question is framed [2].

Another challenge, is that the value of private data changes over time, and may either increase or decrease [16]. Privacy valuation has for example been investigated based on option pricing theory, where the self-information of a private data item is simulated over time using a stochastic random walk [16]. It is however hard to predict whether the value of private information will increase or decrease in value over time, except in trivial cases. One such example is linkability between targeted advertisements (e.g. from doubleclick.net). Such targeted advertisements may be problematic from a privacy perspective, since they may reveal personal preferences, however these advertisements also typically time out after a relatively short period, meaning that the information after this becomes worthless.

2.10.2 How to Measure Privacy Leakage

The discussion in the previous section indicates that is is better to focus on measuring information leakage in IDS alarms based on objective criteria which correlate with the disclosure rate of sensitive information, for example based on Shannon entropy [118], rather than doing detailed privacy valuation analysis. This means that optimising the security monitoring from a privacy perspective effectively will reduce the information disclosure rate.

A privacy leakage metric for IDS, founded on the theory of quantitative information flow analysis [122, 121] and Shannon entropy is described in part III of the dissertation.This metric allows a data controller to set an impact value on given data, based on the perceived sensitivity of the data. This is not an exact valuation of the private data, however it can be useful to weigh up data that clearly is more sensitive from a privacy or confidentiality perspective, and it can also be used to reduce the impact of data that has no or little impact. This is a simplistic approach similar to what is common in risk analysis.

2.11 Attack Prevention or Surveillance, Which Is Better?

Intrusion Prevention Systems (IPS) is a network monitoring technology that extends IDS with the possibility to automatically enforce a computer security policy. A question is then: when is it acceptable from an ethical/moral perspective to automatically enforce a computer security policy, and are there any cases where it can be considered better to automatically enforce the policy than to use traditional monitoring techniques like IDS? A related question is whether blocking of undesirable content is more acceptable than surveillance covering use of undesirable content?

In general, IPS, firewalls and IDS may all leave electronic evidence in the form of system logs or alerts sent to a central security operations centre. It is possible to define rules that enforce a security policy without leaving electronic traces, however this is not common to do. The reason is that system logs are useful to detect and improve rules that perform poorly or incorrectly. It can also be useful to verify correct system operation.

Logging of what is being monitored may also be important for accountability, to audit what is being monitored either by the network owner or by third party quality certification organisations. It should however be noted that such logs also may contain private or confidential information. They should therefore be cryptographically protected both against unauthorised modifications by the MSS provider as well as against external attacks, and should use privacy enhancing technologies, for example anonymisation or pseudonymisation, to avoid showing private or confidential information in cleartext to unauthorised personnel. Since IPS rules typically perform automated actions, then there should normally not be a need to view detailed information from such events in cleartext. IPS alerts should therefore be suitable candidates for anonymisation/pseudonymisation.

A problem with automatic enforcement using IPS, is however that monitoring rules typically are neither perfect, meaning that false alarms may occur, nor complete, meaning that the rule is able to catch all attacks [51]. This means that using an IPS causes a risk that some legitimate traffic also will be denied. On the other hand, one should not be complacent because of having an IPS implemented, since the rule definitions typically are not complete, and may not detect all attack scenarios. A common way for IPSs to enforce preventive actions, is to block traffic from the attacker either permanently or for a given time interval. This can be problematic both from an ethical and business perspective since it may cause benign traffic to be blocked out. There is also a risk of targeted Denial of Service attacks against the IPS or firewall if the adversary uses forged attack traffic to disrepudiate a given user or to block the entire service. This shows that automatic filtering of attack traffic based on blocking traffic that matches given rules can be problematic from both an ethical, business and security perspective, although it clearly is more cost effective than manual 24x7 monitoring of IDS alerts. It is also more efficient since it actually may prevent an ongoing attack, given that the IPS is sufficiently fast and precise, to detect and deter the attack vector without harming innocent third parties.

A somewhat related area, is permanent blacklisting of traffic from certain hosts assumed under control by adversaries, or even censorship of web sites providing content that in a given legislation is deemed illegal. Is automatic enforcement of security policies, for example via rules that deny access to certain on-line resources in this case more acceptable than security monitoring? Is it for example worse to block inappropriate web sites or web sites that may be risky from a security perspective, than if humans investigate such events? This is a discussion on censorship versus surveillance - which one is better or worse. Content filtering is cheaper and may be a better choice from a purely economical perspective, however one may risk liabilities from legitimate users and customers whose service has been interrupted. The other extreme, is whitelisting where only traffic between approved entities is allowed. Such approaches may be useful in certain scenarios, for example for controlling access to critical infrastructures.

Content filtering can be considered better from a privacy perspective provided that IPS alarms are properly anonymised. However it is not necessarily better from an anti-censorship/free speech perspective. Knowing that systems in general log what is being filtered, then it can be discussed whether content filtering is a good argument from a privacy perspective, although it certainly is possible to create IPS rules that either anonymise or encrypt sensitive information or do not log any information at all. Also, a censored environment may give a deceptive perception of reality, something that is morally questionable.

Content filtering using IPS or firewall technologies is in other words useful and can be morally acceptable if used against attack scenarios, provided that the MSS provider aims at minimising the harm from both a privacy and freedom of speech perspective, as well as avoiding harm for innocent third parties. However a potential risk is that the IPS may be vulnerable to denial of service attacks.

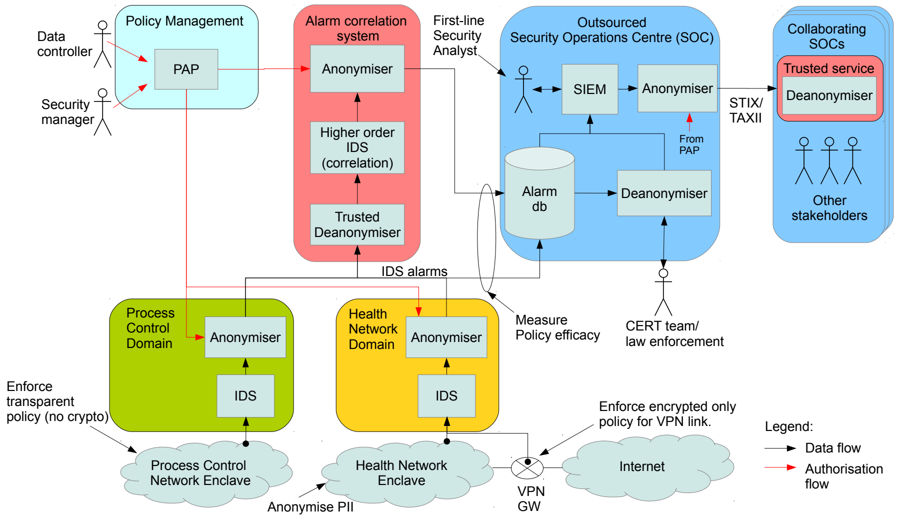


Figure 2.12.1: High-level architecture illustrating some use-cases for the proposed reversible anonymisation scheme supported by metrics-based privacy enforcement.

2.12 A Roadmap Towards Improved Privacy for Managed Security Services

This section proposes a roadmap for how technologies, tools, techniques and methodologies can be combined with ethical principles in order to improve privacy and security. Such a roadmap depends on existing privacy legislation, and may benefit from public procurement policies favouring security operations with certified processes for operating in a privacy friendly manner. It will depend on existing standards and best practices, like the ISO27000 set of security management standards, in addition to new methods, tools and techniques for measuring privacy leakages and performing privacy enforcement.

The privacy enhancing technology used to improve MSS requires a reversible anonymisation scheme, which can anonymise arbitrary information in security monitoring services and subsequently let authorised stakeholders, for example security analysts, access this information by reversing the anonymisation. The privacy policies are being managed by the data controller and security manager who must agree to deploy new privacy policies via the Policy Administration Point (PAP). This means that a mechanism for enforcing separation of duties constraints will be needed to enforce controlled privacy policy deployment. It is also envisaged that a trusted service may be allowed to operate on certain private or confidential information, in a similar way to the airport example mentioned earlier. Alarm correlation systems is an example of such a trusted service. Such systems are used to improve the precision of intrusion detection systems, by correlating information between different IDS sensors and attack detection technologies and also correlate information in time, to detect suspicious patterns for combinations of IDS alarms.

Different users or roles will typically have different privileges for accessing sensitive information. First-line security analysts may be trusted to see certain private or confidential information, for example to a given confidentiality level. A Computer Emergency Response Team (CERT) or law enforcement may be authorised to additional information on a needs basis, in order to investigate suspected attacks.

Another use case, is exchanging information related to attacks and threats with other stakeholders, for example other MSS providers, CERT or similar, however access to private or confidential information may be restricted on a needs basis, possibly allowing certain trusted services to give notification about attacks or best practices in order to improve attack coordination between different organisations. Access to private or confidential information should be logged to ensure transparency in all these cases.

2.12.1 Requirements and Use-cases for Privacy-enhanced Network Monitoring

Figure 2.12.1 outlines some use cases for the privacy-enhanced network monitoring service. Privacy policies will typically be defined after a privacy and security impact assessment, which identifies the data elements and services that need to be protected. The privacy impact assessment may also identify under which conditions data are sensitive, for example if an identifiable subset of IDS alarms should be anonymised. The privacy impact analysis involves an analysis of where private or confidential information may be leaking, what this information is and how much or how frequently information is leaking. Both the data controller and security manager then need to participate in taking an informed decision on what is necessary and sufficient information from a security perspective, in order to define a privacy policy.

The privacy policy also includes defining who will have access to what information under which conditions. The privacy policy should also support giving different stakeholders access to different parts of the private or confidential information according to their needs. This means that the solution should support a multi-level privacy or security based solution.

It is assumed that the anonymiser will be designed to anonymise information in XML-based protocols, and that the IDS alarms can be conveyed in an XML-based format, for example the Intrusion Detection Message Exchange Format (IDMEF) [62]. This assumption simplifies the implementation and makes it possible to design a solution that will work well in a service oriented architecture based on web services.

Figure 2.12.1 also illustrates that the anonymiser and deanonymiser can be used to build trusted alarm correlation systems, where the trusted service is authorised to correlate necessary parameters in order to do efficient attack detection. Private or confidential information in correlation alarms can subsequently be anonymised, to avoid leaking more sensitive information than necessary. This use case can be considered similar in some ways to the whole body scanner example mentioned earlier, since technology is allowed to look for suspicious patterns, however private or confidential data is being protected by encryption from non-authorised parties, and only nonsensitive parts will not be anonymised.

In addition, it is assumed that privacy leakage metrics can be used to verify the efficacy of a privacy enforcement scheme by measuring *how much* information that leaks. This also means that the privacy leakage metrics should be aggregatable, so that a correctly functioning privacy policy will show reduced privacy leakage after improving the policy, if the policy is working correctly. The privacy leakage metrics can also be used to identify *where* information is leaking, as well as supporting an analysis of *what* this information is by allowing investigation of the underlying data. The anonymiser should allow for defining both default PERMIT privacy policies, where any information that is explicitly being authorised may be anonymised, and default DENY policies, which by default anonymise all information, and where selected information deemed safe from a privacy or confidentiality perspective subsequently can be declassified. This means that privacy leakage metrics will be able to support a continuous improvement process based on the well-known Plan, Do, Check, Act method of improvement which is used by the ISO27000 set of security management standards.

The anonymiser may also be useful for anonymising exchange of threat information with collaborating Security Operations Centres (SOCs), for example using the Structured Threat Information Expression (STIX) and Trusted Automated eXchange of Indicator Information (TAXII) protocols defined by Mitre [130, 30]. This allows sensitive information in the exchanged threat information to be anonymised, however a trusted service may be allowed to deanonymise and operate on certain parts of this information.

The privacy policies should furthermore support defining both policies that explicitly describe which information that is being anonymised (default PERMIT policies) and policies that explicitly describe which information that is being allowed in the output (i.e. default DENY policies). The latter is needed to support the privacy by default criterion of the Privacy by Design guidelines [25].

The privacy enforcement method should also support enforcing different levels of information opacity. A process control network enclave may for example want to enforce a transparent network policy where all information is in cleartext, to be able to verify the programming of vulnerable Programmable Logic Controllers (PLCs). This would reduce the risk of some man-in-the-middle attacks (e.g. Stuxnet). A similar approach can be used to trigger alarms if the entropy for a Virtual Private Network (VPN) gateway is anomalistic by enforcing an opaque information policy where all information is required to be encrypted, in order to detect cleartext attacks on these gateways [151].

Another use case is monitoring a health network, where the anonymisation policy would be set up to anonymise potentially personally identifiable information in IDS alarms, for example by anonymising the payload of the alarms, and ensuring transparency by logging when security analysts accessed this sensitive information.

2.13 Conclusion

This chapter has identified several requirements for a privacy-enhanced networking monitoring service based on the ethical discourse and analysis of possible use cases.

A risk to consider for outsourced Managed Security Service (MSS) providers, is Moral Hazard or shirking. The network owner cannot assume that security monitoring outsourced to a MSS provider will be performed in a way that aims at reducing privacy invasiveness as much as possible, and at the same time with as good security as possible. Contracts to perform outsourced managed security services, as well as related metrics and indicators, should therefore be *incentive compatible*, which means that payment should be related to auditable performance metrics both related to security and privacy to give the MSS provider an incentive to improve both privacy and security. This also means that *transparency, auditability* and *nonreputability* is required on how the MSS operation is being performed. This information must be protected against unauthorised access.

It is recommended to use a *defence-in-depth* strategy to ensure that more than one barrier needs to break to access private or confidential information. Another requirement is being able to enforce separation of duties constraints, so that several stakeholders must collaborate in order to reveal a secret. Another example is using both perimeter defences (access control/firewalls) and encryption to access sensitive information in the organisation, instead on relying purely on perimeter defences.

Content filtering using IPS or firewall technologies can be considered useful and also morally acceptable against on-line attack scenarios, provided that the rules aim at minimising the impact both from a privacy and freedom of speech perspective. However a potential risk with automatic content filtering based on known attacks is being vulnerable to denial of service attacks, something that may harm benign use of the system.

Security analysis done by IDS/IPS should aim at operating strictly according to the need-to-know principle. One way to achieve this, is using privacy leakage metrics to identify leakage of private or confidential information. It is then possible to define a privacy enforcement scheme that minimises the measured leakages. This allows security analysts access to private or confidential information on a needs basis. However access to such information must be securely logged, to ensure transparency and nonreputability. Another technique that can be used is trusted applications that are allowed to decrypt and monitor sensitive information for signs of attack. This can be used to limit the number of people that need to access private or confidential information.

Privacy-enhanced IDS may give increased productivity through fewer false alarms, for example from identifying and tuning the most problematic IDS rules to be less privacy invasive.This may furthermore give the monitoring company a better reputation for handling of private and confidential information, something that is required in certain business cases, for example for health institutions or for critical information infrastructures. The chapter also outlines how privacy leakage metrics can be used with privacy enforcement, in order to support a continuous improvement process.

Certification and auditing improves adherence to the ethical guidelines. Businesses that provably perform unethical security monitoring would risk losing their quality certification, which would be detrimental for the reputation of a company in the security business. This can either be enforced via governmental regulation or voluntary regulation, as part of a quality accreditation.

The dissertation will use these ethical guidelines as requirements for designing the privacy enforcement mechanisms in part II and privacy metrics and methodology in part III.

Part II

Privacy Enforcement for Intrusion Detection Systems

Part II does an investigation of privacy enforcement mechanisms for intrusion detection systems. It starts out with a relatively simple solution based on a two-tiered privacy enhanced IDS architecture in Chapter 3 which uses irreversible coarse-grained anonymisation based on the eXtensible Access Control Markup Language (XACML). Chapter 3 also does a more thorough analysis of where intrusion detection systems may leak information. Chapter 4 elaborates on this solution, and proposes how to do fine-grained authorisation and anonymisation of information in XML documents using XACML-based privacy policies. The solution uses decision caching for increased performance, and supports default PERMIT policies. The last chapter of part II extends the solution to cover reversible anonymisation of information in XML documents down to octet-ranges of individual elements or attributes of the document. This solution supports multi-level security, separation of duties constraints based on threshold cryptography as well as data retention based on a secure logging scheme. The solution furthermore supports both default PERMIT and DENY anonymisation schemes in order to support privacy by default.

Chapter 3

Two-tiered Privacy Enhanced IDS Architecture

This chapter describes an architecture for privacy-enhanced intrusion detection systems, that separates privacy-invasive and privacy-preserving operations. This can be useful in cases where less sensitive network monitoring (e.g. first-line monitoring) is outsourced to a third party and more sensitive network monitoring operations and data forensics (second-line operations) are performed in-house or by law enforcement agencies. This is the first attempt at defining a privacy-enhanced IDS architecture, and is the starting point for the more elaborate privacy enforcement solutions that are described in the next two chapters.

3.1 Introduction

This chapter proposes an architecture for privacy enhanced Intrusion Detection Systems (IDS). It ensures that sensitive information, as defined by a security policy in the network being monitored, is not being exposed to third-party organisations performing the network monitoring. The objective is to achieve intrusion detection systems with good usability and detection efficiency for third-party organisations performing the network monitoring, while at the same time providing trustworthy confidentiality and privacy for the network owner and its customers.

The main contribution in this chapter, as indicated in Fig.3.1.1, is to provide better privacy preservation of monitored subjects in intrusion detection systems by using a two-tiered approach with:

- a *first line* privacy-preserving subsystem, which is operated by security analysts running a 24x7 service;
- a privacy-invasive *second line* that allows for further analysis by experts, but where all privacy violations are logged;
- built-in *privacy policy enforcement points (PEP)* using eXtensible Access Control Markup Language (XACML) based policies [127].

Making the monitoring organisation accountable for the privacy violations performed during normal operation should be a strong incentive to minimise the number of privacy violations.

The INCH working group in IETF has defined several standards that are relevant for exchange of data between IDS sensors. The Intrusion Detection Message Exchange Format (IDMEF) is a standardised format for IDS alarms [62]. It can be used in conjunction with the Intrusion Detection Exchange Protocol (IDXP) [11], for transporting alarms from IDS sensors and to a datawarehouse connected to a monitoring console. The Incident Object Description Exchange Format (IODEF) is a format for representing computer security information commonly exchanged between Computer Security Incident Response Teams (CSIRTs) [105]. It can be used together with the draft Real-time Internetwork Defence (RID) protocol over SOAP [69], to provide end-to-end security and support for requesting traffic traces. IODEF over SOAP/RID may be useful as interface towards a data forensic interface, like a Time Machine (TM) [66], or for communication between Security Operations Centres (SOCs). The architecture presented in this chapter is based on these standards.

This chapter is organised as follows: The next section covers potential privacy violation areas for intrusion detection systems. Section 3.3 shows an overview of the proposed privacy-enhanced architecture and how it covers the privacy violation areas. Section 3.4 shows an example of an XACML policy for IDS alarm handling. Section 3.5 discusses the efficiency of the proposed solution. Section 3.6 compares the proposed solution to other existing solutions, which leads to some concluding remarks and outlook on future research

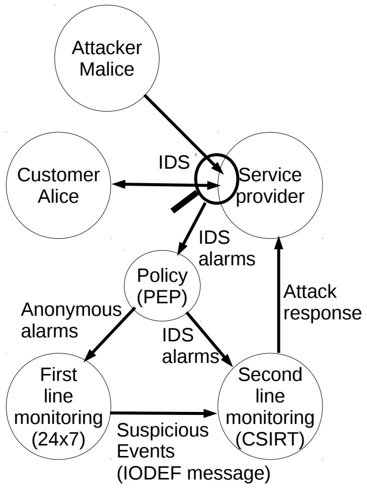


Figure 3.1.1: Overview over two-tiered approach.

directions in section 3.7.

3.2 Privacy violation areas

The three main areas where privacy or confidentiality may be compromised in intrusion detection systems are:

- alarm handling;
- forensic interface via a Time Machine or similar interface [66];
- and rule handling.

IDS alarms usually consist of an event classification, packet header data and relevant payload from the packet that triggered the alarm. The security analysts look at the payload sent with the IDS alarm to check whether it is a likely false positive or a real attack. Manual investigation of the alarms is usually needed, since current intrusion detection systems suffer from a high false positive rate. If it is difficult to categorise the alarms based on packet header and payload, then it is assumed that further analysis can be done by investigating the entire data session using a forensic interface.

One of the main research challenges is to improve privacy protection without significantly deteriorating detection efficiency and usability for the security analysts. Payload data and IP addresses will for example leak some privacy related information about the customer who is accessing the monitored network service. It is therefore a conflict between:

- the security analyst's requirement for a usable and efficient data mining interface to investigate intrusion alarms;
- and the customers and network owner's requirement for privacy and confidentiality.

One conceivable solution would be that the IDS did not leak any privacy related information, and that security analysts could perform assertions towards the real data, but not reveal the real data itself. From the security analyst's point of view, this would be like troubleshooting a black box where the functionality inside is unknown, which can be a frustrating experience. This means that the real reason behind the attack would not be *perceivable* for the security analyst, which would reduce usability of the IDS significantly. Such a solution would only work if the IDS had perfect rules. This means that the security analyst could have reasonable confidence that an alarm from the IDS indicated a real attack, and also that the IDS was able to verify whether an attack was successful or not. Until IDSs have improved to this level, it seems inevitable that some level of manual analysis will be required, which means that privacy in some cases will be violated.

It is however important that potential privacy and confidentiality violations are *accounted for*, so that the customer and if required, law enforcement agencies can verify which privacy and confidentiality violations that have been performed to do an efficient network monitoring service. With accountability on privacy violations, network monitoring companies and vendors could compete on providing the monitoring product with the lowest average level of privacy violations. Public authorities could then set requirements for what level of privacy violations that are acceptable under given circumstances.

One possible way to organise security monitoring in order to lower the overall number of privacy violations, is to employ the principle of disseminating information on a *need-to-know* basis. Our solution implements this by dividing security analysis into two tiers, where the first line consists of a group of people performing a 24x7 monitoring of the networks using privacy-preserving techniques. The second line consists of security experts that have security clearance and authorisation to perform necessary privacy violations to investigate whether attacks were successful or not. Second line analysts would when necessary provide forensics data to Computer Emergency Response Teams (CERTs) and law enforcement agencies that would investigate successful attacks.

In this way, it may be possible to outsource first line monitoring services, whereas second line services either can be kept within the same organisation, or outsourced to a trusted third party but kept within the same legislative area (for example within the same country) to ensure that it would be possible to prosecute illegal dissemination of private or confidential information by insiders in the monitoring organisation.

The Security Operations Centre (SOC) may furthermore be able to perform logging of all traffic over a specified period of time for forensic analysis or for retrospective IDS analysis. This can either be done using Time Machine based network recorders [66], or using simpler approaches with tcpdump logging of network traffic. Having the possibility to perform data mining of past network traffic will significantly increase the potential confidentiality and privacy threat unless access to such data is properly controlled.

For signature based IDS, IDS rule updating may also be a possible source of privacy violations, since the IDS rules can be designed to return sensitive information by a corrupt security analyst. The IDS rules continually need updating to make signature based IDS work, since new attack vectors will require new IDS rules to be added. Most of the IDS rules used are based on publicly available rule sets that security companies trust. It may therefore be possible for an adversary to attack the IDS by modifying the trusted IDS rule set. It is therefore important that IDS rule set updates are being peer-reviewed and integrity checked before being deployed. A configuration management system should be used to track changes of the IDS rule set. All IDS rule updates performed by the rule manager should therefore be logged to the activity log.

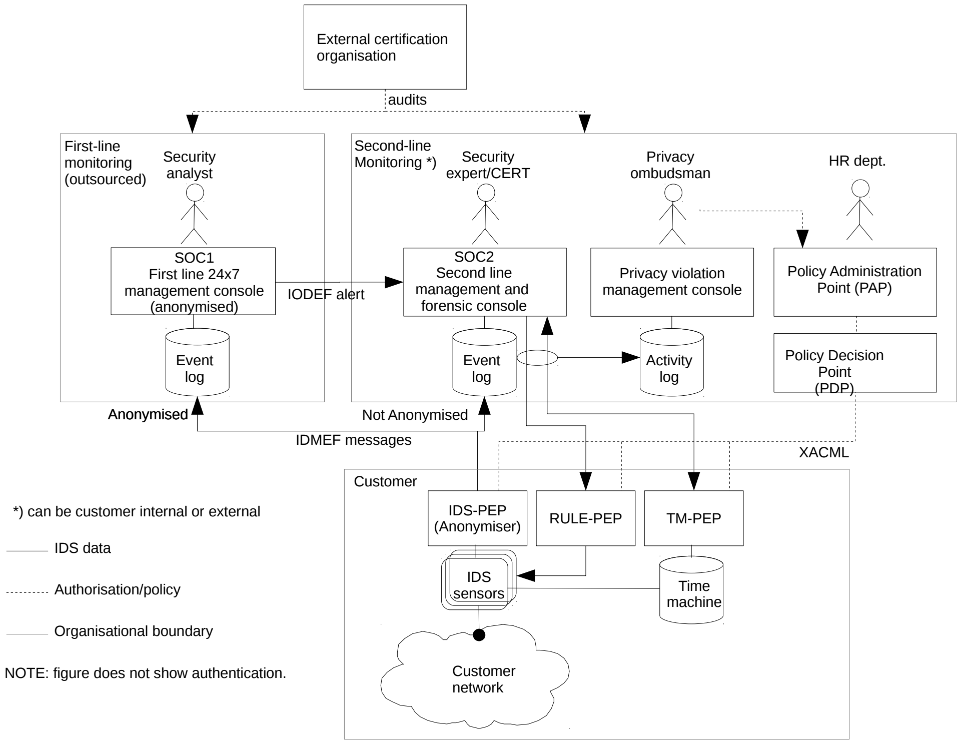

External certification organisation
audits
First-line monitoring (outsourced)
Security analyst
SOC1 First line 24x7 management console (anonymised)
Event log
IODEF alert
Second-line Monitoring *)
Security expert/CERT
SOC2 Second line management and forensic console
Event log
Privacy ombudsman
Privacy violation management console
Activity log
HR dept.
Policy Administration Point (PAP)
Policy Decision Point (PDP)
Anonymised
Not Anonymised
IDMEF messages
XACML
Customer
IDS-PEP (Anonymiser)
RULE-PEP
TM-PEP
IDS sensors
Time machine
Customer network
*) can be customer internal or external
IDS data
Authorisation/policy
Organisational boundary
NOTE: figure does not show authentication.

Figure 3.2.1: Privacy-enhanced IDS architecture.

3.3 Architecture

First line monitoring is assumed outsourced to a third-party organisation to reduce the operating cost of running a 24x7 monitoring service. The second line operation consists of a small set of trusted experts with sufficient security clearance, that have got access to confidential information which may violate privacy.

The proposed architecture in Fig.3.2.1 provides Policy Enforcement Points (PEPs) that act as intermediaries between IDS monitoring consoles and sensor(s) with intrusion detection, rule handling and data forensic (Time Machine) functionality. This authorisation and anonymisation function avoids privacy violations for first-line operations. Second line IDS operations can request the real data sessions and set up alarm correlation assertions for first-line operation, but all such requests are accounted for in the activity log.

A privacy-enhanced IDS should obey the principles of data avoidance and data reduction [104]. Data avoidance means that the user should be forced to only disclose the minimum amount of information necessary to the IDS. This implies that an IDS does not need to know the identity of a monitored user, until it provably detects an abuse. Data avoidance is supported by having a two-tiered architecture, where the most labour intensive part, first line monitoring of all incoming events, works in privacy preserving mode (see Fig. 3.2.1).

Data reduction is supported using XACML obligations. These obligations remove data that is regarded as sensitive in the IDS alarms according to the privacy policy. XACML was chosen as authorisation policy language, because it is a mature standard that can use the Security Assertion Markup Language (SAML) for authentication in a federated environment. It fits well into a Service Oriented Architecture (SOA) and has quite broad vendor support compared to other alternatives like the Enterprise Privacy Authorisation Language (EPAL) [23]. We considered XACML to be more general than the Platform for Privacy Preferences (P3P) [83], which focuses mainly on web based authorisation.

The IDS authorisation framework has a Policy Administration Point (PAP) that controls access to first- and second line data. The company's data controller together with customer and company management is responsible for managing roles (first line or second line) and privacy policies in the PAP. The Human Resources (HR) department defines which role employees belong to.

Law enforcement agencies and CERT teams can be granted access to second line monitoring in order to investigate ongoing attacks. The IDS Policy Enforcement Point (IDS-PEP) communicates with the Policy Decision Point (PDP) on authentication[1] and access control (authorisation) of the alarm data stream.

The entire monitoring organisation should in addition be audited by an external quality certification authority at regular intervals, and these audits should include an analysis of how privacy-invasive the operation is compared to other companies in the same sector.

After authentication and authorisation, the IDS-PEP accepts IDMEF messages carrying alarms from a set of IDS sensors. The IDS-PEP then forwards streams of IDMEF messages anonymised according to the security policy

[1] It is envisaged that the Security Assertion Markup Language (SAML) will be used for authentication. SAML also fits well into the authorisation architecture, since it supports transport of XACML request and response messages.

of the receiving manager application. Data streams authorised for first line operation will be anonymised according to the obligations presented in the XACML security policy for the role *firstLine*. It is sufficient to anonymise data in Policy Enforcement Points (PEPs), since the outsourced organisation does not have access to manage the sensors.

If the first-line Security Operations Centre (SOC) identifies a suspicious message, then it will alert the second line SOC using an IODEF message. The alert identifier can be used by the second line operations centre to identify the full non-anonymised alarm.

The security policy for the role *secondLine* will in general not have any restrictions on access to data, however second line operations will have the XACML obligation for access to data that all operations will be logged. Access to the TM is governed by a separate privacy policy enforced by TM-PEP. The TM server policy will only grant access to security analysts with role *secondLine* with the XACML obligation that all operations are logged.

The data controller will have read access to summary data from logged second line operations. This implies that the system also needs a policy for role dataController. A separate security policy role *ruleManager* is required for updating the IDS rule set, because the Security Operations Centre typically delegates this responsibility only to a subset of the second line security analysts.

A general problem with IDS, is that they are vulnerable to Denial of Service (DoS) attacks. This means that an excessive amount of bandwidth will be required to transport alarms during an attack. Some IDSs are able to mitigate DoS attacks. One example is PreludeIDS, which supports setting rate limitations on IDS alarms to avoid excessive bandwidth usage during DoS attacks. This functionality can be managed centrally by implementing an XACML policy that gives obligations for the amount of alarms per second the IDS can send out.

3.4 XACML policy example

This section provides an example of how the envisaged IDS XACML profile can be used. It does not focus on the authentication part, which is expected

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:1.0:context:schema:os"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context:schema:os
         http://docs.oasis-open.org/xacml/access_control-xacml-1.0\
         -context-schema-os.xsd">
  <Subject>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
      <AttributeValue>soc1@outsourced.example.com</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
               DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>IDS-PEP</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
               DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Figure 3.4.1: XACML request for first line operation

to be very similar to existing federated access control solutions using SAML to convey XACML requests [101].

Fig.3.4.1 shows the XACML request sent from the IDS-PEP to the PDP when the first line manager application (SOC1) attempts to connect to the IDMEF data stream via the IDS-PEP. The XACML request takes three parameters:

- the subject to be authorised is the manager application referenced by *soc1@outsourced.example.com*;
- the resource we want to connect to is the IDS-PEP;
- and we want to perform a *read* operation as described in the *Action* element.

The full XACML authorisation policy is not included in this chapter, but can be found in [93]. Fig.3.4.2 shows the XACML response message from the PDP when SOC1 requests to read the IDMEF data stream from IDS-PEP. The IDS-PEP has an *Obligation* to anonymise the payload. The *Attribute-Assignment* element uses the format *idmef:element:attribute* for identifying

```
<Response>
  <Result ResourceID="IDS-PEP">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="urn:ietf:idmef:anonymize" FulfillOn="Permit">
        <AttributeAssignment AttributeId="idmef:AdditionalData:meaning"
          DataType="http://www.w3.org/2001/XMLSchema#string">payload
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 3.4.2: XACML reply for first line operation

```
<Response>
  <Result ResourceID="IDS-PEP">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="LogPrivacyViolations" FulfillOn="Permit">
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 3.4.3: XACML reply for second line operation

the correct IDMEF element and attribute. In Fig.3.4.2 the IDMEF element *AdditionalData* and the attribute *meaning* are being referenced and the value of the *AttributeAssignment* element is the identifier to anonymise (payload).

Fig. 3.4.3 shows the XACML response message from the PDP when SOC2 requests to read the IDMEF data stream from the IDS-PEP. The IDS-PEP has an *Obligation* to log privacy violating activities for second line operations.

3.5 Efficiency of proposed solution

Performing XACML authentication and authorisation in the PEP is a one time initialisation cost. The anonymising policy in the IDS-PEP therefore only implies a linear search through all elements of IDMEF XML data and checking the content of relevant IDMEF attributes for each of the IDMEF elements. This can be done in $O(n \times o)$ time, where n is the number of IDMEF elements and o is the number of XACML Obligations. Similarly, authenticating towards the TM is also a one time cost, that is negligible. It can further be expected that the bandwidth required for manual analysis of TM data will be negligible, because the TM in general is not able to store full data sessions. For example, the TM developed for the BRO IDS [78] only stores the first 20 kb of each session by default [66]. Retrospective IDS analysis using the TM is assumed integrated with the IDS, so that the bandwidth requirements here are 0.

To test the bandwidth efficiency, an anonymiser plug-in was implemented for the PreludeIDS hybrid IDS [128]. This plug-in implements a simple privacy policy that removes the payload from IDMEF *AdditionalData* elements containing payload, to get an indication of the bandwidth saving of a simple anonymisation policy.

One Snort IDS sensor [84] was connected to PreludeIDS with all rules enabled, and PreludeIDS was set to log alarms both to a database and to an IDMEF XML log file. A security scan was then performed using Nessus [129] towards a Linux server, which triggered 1056 alarms. The results show that anonymising the payload of IDMEF IDS alarms (first line operation) cuts the amount of IDMEF XML data by 8.5%. The average alarm size was reduced from 4.47 kb to 4.09 kb. The mean payload size is only 380 bytes, and the median is 424 bytes. This indicates that the size distribution is somewhat skewed towards smaller payload sizes. Removing the payload will in other words reduce bandwidth demand somewhat for IDMEF alarms, however perhaps less than one would have thought. The reason for this may be that a Nessus scan mostly triggers attack rules, like for example malicious HTTP requests, which normally do not contain a vast amount of payload. Another reason is that payload from port-scanning preprocessors is small, since Snort stores summary data from port-scanning preprocessors in the IDMEF *AdditionalData* records containing payload.

3.6 Related work

The content of this chapter was initially published in the proceedings from the IDAACS 2009 conference in Rende, Italy [133]. The BRO IDS [78] supports a way to anonymise the payload of a packet instead of removing the entire payload [98, 78]. Our solution is different, since it anonymises IDS alarms instead of anonymising the captured packet traces. This is sufficient in a scenario where first-line security analysis has been outsourced.

There exists some earlier work on privacy-enhanced host-based IDSs that pseudonymise audit data and performs analysis on the pseudonymised audit records [64, 123, 50, 124, 104]. A similar approach is further elaborated in [51], which builds the privacy policy into the IDS rules by defining a privacy-preserving rule language that pseudonymises payload and other information that is defined as sensitive. Kerberos [91] is used for authentication. This approach focuses on reversible protection mechanisms using cryptographic techniques for pseudonymisation.

Our approach separates the information into two security domains, where the lower domain is anonymised and the higher domain is non-anonymised. Since the same alarm identities are used in both domains, then it is trivial to correlate alarms between first line and second line operations in case an event gets escalated from first to second line for further investigation. This avoids the computational overhead of a pseudonymised solution, however one loses the possibility for fine-grained enforcement of access to data. We argue that in order to do efficient incident handling, then it will be necessary for the data controller to give the second line operation/CERT team authorisation to operate autonomously on unrestricted data, when attacks are being investigated. This team is expected to be small and have a high security clearance and authorisation compared to the 24x7 first line team.

Another group of monitoring systems focus more on filtering and anonymising network traffic. The anonymised and filtered network traffic is called a traffic flow and IDS analyser agents can subscribe to these anonymised traffic flows. One example is LOBSTER [81], which has developed a common Monitoring API (MAPI) for acquiring traffic monitoring data from a set of distributed network monitoring sensors. It includes a comprehensive API for anonymisation (AAPI) of sensor data based on Virtual Organisation groups.

Another example is the conceptual system described by the PRISM IST project. This approach uses a privacy-enhanced architecture based on analysis of pseudonymised IP Flow Information Export (IPFIX) traffic flows [55, 10]. Using IPFIX for IDS analysis purposes has some disadvantages. It is limited to analysis of traffic on the network layer and higher protocols [10], since link layer information is not conveyed in IPFIX. This means that link layer attacks and attacks on other protocols than IP will go undetected. Also, IPFIX is only able to record traffic streams in one direction. This may impair some types of IDS analyses where bidirectional data stream reassembly is used for detecting attacks, like some web attacks. In addition, IPFIX is oriented towards flows of IP packets, which means that it is not trivial to convey information from host-based IDSs in this protocol. Another argument against a traffic flow based approach, is that it can be bandwidth demanding if the flow includes a large part of the transferred packets and matches a broad category of packets.

These deficiencies indicate that a traffic flow based IDS probably will be a supplement to existing IDSs rather than replace them. Our proposed solution will regardless be able to integrate alarms from traffic flow based IDS as well as other types of IDSs. Despite these deficiencies, such systems will have an advantage when it comes to detecting certain types of distributed attacks, like worms or distributed denial of service (DDoS) attacks.

Many existing privacy-enhanced solutions propose that the data controller should have a cryptographically enforced[2] veto right against disclosure of traffic data in case of attack investigation. This approach is similar to the one proposed in [51]. The relatively simple XACML-based privacy-enhanced IDS proposed here does not support this, however a solution that enforces vetoing based on secret key sharing will subsequently be described in Chapter 5.

3.7 Conclusion

This chapter proposes a two-tiered architecture for privacy-enhanced intrusion detection systems that separates out anonymous data processing in the

[2] Cryptographic enforcement can for example be based on threshold cryptography, where n out of k participants must agree to decrypt the ciphertext.

first line service and runs privacy-invasive operations in the second line service. A natural extension of this scheme, is to extend the scheme to *n* different security levels. This is investigated in chapter 5.

The architecture is based on existing standards for IDS alarm handling from IETF and security policy handling from OASIS. The XACML-based privacy policy enforces logging of all privacy violating operations, which can be audited by both a company-internal data controller and by external certification organisations.

Our approach limits the number of people that have got access to privacy sensitive data, which should lower the risk of misuse by insiders. In effect, this makes it more viable to outsource first-line 24x12 monitoring services to third parties. Creating an XACML policy for rate-limitation in the IDS-PEP allows federated management of alarm rate limitation policies to avoid using an excessive amount of bandwidth during attacks.

Most existing privacy enhanced IDS approaches enforce data protection mechanisms early after data has been captured. This requires a custom made IDS framework, whereas our approach focuses on integrating and anonymising alarm data from existing commercial IDSs for use in an outsourced scenario. We argue that data can be sufficiently well protected between the IDS and the anonymiser using encrypted data links.

A limitation with the proposed approach, is that the efficiency of the first line operation will be affected by not having potentially sensitive information like payload available. This can to some extent be compensated by using workflow based alarm correlation engines that emit anonymised alarms.

The next chapter elaborates on the solution proposed here and proposes a general XACML-based framework for fine grained anonymisation of XML documents, including anonymisation of IDMEF-based IDS alarms.

Chapter 4

Decision-cache Based XACML Anonymiser

This chapter describes a decision cache for the eXtensible Access Control Markup Language (XACML) that supports fine-grained authorisation and anonymisation of XML based messages and documents down to XML attribute and element level. This is an elaboration of the coarse-grained anonymisation protocol proposed in the previous chapter, and is based on [135]. The decision cache is implemented as an XACML obligations service, where a specification of the XML elements to be authorised and anonymised is sent to the Policy Enforcement Point (PEP) during initial authorisation. Further authorisation of individual XML elements according to the authorisation specification is then performed on all matching XML resources, and decisions are stored in the decision cache. This makes it possible to cache fine-grained XACML authorisation and anonymisation decisions, which reduces the authorisation load on the Policy Decision Point (PDP). The theoretical solution is related to a practical case study consisting of a privacy-enhanced intrusion detection system that needs to perform anonymisation of IDMEF based IDS alarms before they are sent to a security operations centre operating in privacy-preserving mode. The solution increases the scalability of XACML based authorisation significantly, and may be instrumental in implementing federated authorisation and anonymisation based on XACML in several areas, including intrusion detection systems, web services, content management systems and GRID based authentication and authorisation.

4.1 Introduction

The objective of this Chapter, is to use XACML for fine-grained authorisation and anonymisation of IDMEF XML messages from Intrusion Detection Systems (IDS), to control what information that can be disseminated to whom from an IDS service. A challenge with XACML is that the current implementations do not scale well [80]. It is a therefore a risk that the central rule processing engine in the Policy Decision Point (PDP) may be a bottleneck for a potentially large amount of authorisation requests from individual XML elements. Another challenge, that has not been solved as far as we are aware of, is how to do fine-grained anonymisation or pseudonymisation of XML documents or messages by using XACML. We propose how this can be mitigated by adding a decision cache as an XACML obligations service that can store decisions based on unique key values.

Our solution is not limited to the domain of IDS services. Fine-grained access control and anonymisation of XML documents backed up by a client-side decision cache may also be useful for GRID services to provide a more scalable authorisation that effectively can delegate simple decisions to a distributed set of decision caches. It can also be useful for authorisation and anonymisation of web services, middleware like for example JBoss or even content management systems, in order to ensure that some information deemed sensitive is not distributed via the service. In that respect, the solution can also be regarded as a simple XACML controlled application level firewall for content in XML documents and messages.

This chapter is organised as follows: The next section gives an introduction to XACML and an overview of the proposed solution. Section 4.3 describes the architecture and Section 4.4 covers the technical solution in more detail. Section 4.5 shows an example authorisation of XML resources based on the proposed solution including initial authorisation, individual element authorisation request and response and decision cache handling. Section 4.6 describes the efficiency of the proposed solution. Related work is subsequently discussed in Section 4.7 and Section 4.8 concludes the chapter and gives some suggestions for further research.

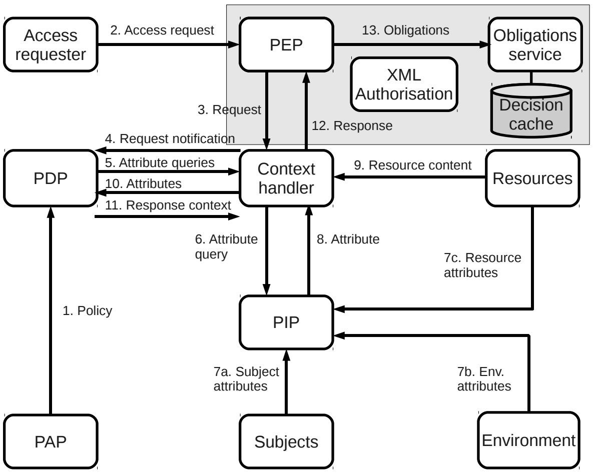


Figure 4.2.1: XACML architecture with decision cache.

4.2 Overview of the Proposed Solution

XACML is an access control policy language based on policies written in XML. It uses a model for access control that clearly separates policy decisions in the Policy Decision Point (PDP) from policy enforcement the Policy Enforcement Point (PEP) as shown in Figure 4.2.1. The Context Handler and Policy Information Point (PIP) ensure that subjects, resources and other environment attributes can be made available to the PDP when policies are being evaluated. Subjects, resources and environmental attributes can also be passed in via the XACML Request message. We use the latter approach, since the anonymisation and authorisation service basically is an extension of the PEP.

Our solution implements an XML authorisation service that is integrated with both the PEP and the obligations service. The obligations service furthermore manages the decision cache.

From an architectural and system management perspective, it is preferable to be able to reuse XACML as far as possible for fine-grained authorisation and anonymisation of XML documents and messages. This is viable under the assumption that access control decisions for authorisation or anonymisations can be regarded as final and do not change within a defined time span. This means that an access control decision to publish sensitive material will

Symbol	Description
$a_{p,q}$	The XACML authorisation decision number q by resource policy number p.
$b_{p,q}$	The block marker or pattern used to anonymise the data (optional).
q	Decision number.
$D_{p,q}$	Decision cache tuple representing decision number q performed by the XACML resource policy number p.
$K_{p,q}$	Unique dictionary key for decision q and policy p.
$policy_{p,q}$	Anonymisation policy to perform on the content of r_p for decision q.
p	XACML resource policy number.
r_p	Resource number p that needs authorisation.
s	Scope parameter number for XACML identifiers.
$t_{p,q}$	The absolute time (UTC) when the cached authorisation decision times out.
$\tau_{p,q}$	Last time this decision cache entry was used.
$v_{p,q,s}$	Parameter values identified by $scope_{p,s}$ that are required by the XACML policy p in order to perform decision number q.
$scope_{p,s}$	XPath scope expression that extracts required parameter values for the the XACML policy p.

Table 4.2.1: List of notations for decision-cache.

Parameter	Decision cache XACML AttributeId
$b_{p,q}$	$b_{p,q}$ is stored in the content of an *AttributeAssignment* with ID urn:prile:org:resource:p:policy:$function$
$policy_{p,q}$	urn:prile:org:resource:p:policy:$function$ where $function$ =[replace-with|pad-with|...] for decision q.
r_p	urn:prile:org:resource:p:id
$\Delta t_{p,q}$	urn:prile:org:resource:p:cache-timeout (PEP calculates $t_{p,q}$ from the current time plus $\Delta t_{p,q}$ for decision q).
$scope_{p,s}$	urn:prile:org:resource:p:assertion:s:scope
$v_{p,q,s}$	urn:prile:org:resource:p:assertion:s:value for decision q.

Table 4.2.2: Mapping of XACML response parameters for decision cache.

not be undone or reconsidered under normal circumstances.

Rules for access control policies will in many cases be static, meaning that they are based on some stable conditions. For example rules using fixed strings or rule patterns identifying IP addresses, e-mail addresses or URLs accessed. For static rules, it will be possible to have decision cache entries with infinite expiry time, that only will be ejected from the cache if the cache is invalidated, for example due to an updated authorisation policy. In other cases it may be useful to only grant access for a limited time period before authorisation needs to be renegotiated.

Utilising a decision caching authorisation system also means that cache entries and rules can be made much simpler than the original XACML expressions, however at the expense of using more memory. It can be expected that the cache has a minimum working set of active authorisations, which means that the decision cache will need at least a certain amount of memory for cache entries in order to operate efficiently. However, if the working set of cached decisions fit into memory, then the load on the XACML rule engine is expected to be tolerable. These assumptions make it viable to use a caching strategy for access control decisions.

4.3 Architecture

Figure 4.4.1 illustrates how the XACML-based anonymiser/proxy for IDMEF XML reports is implemented. Initially, the Managed Security Service (MSS) providers will be authorised towards the PEP. In this example, two MSS providers are shown: an outsourced first line service that only is allowed to see anonymised IDS alarms and a second line service, possibly run in-house, which can see non-anonymised IDMEF alarms. This initial authorisation opens a secure connection from the anonymiser thread and to the alarm database of the MSS provider.

Then the IDS sensors are authorised towards the PEP in order to open a connection from the IDS to a dedicated *Producer* thread in the PEP for each IDS. The *Producer* thread is responsible for copying IDMEF messages to all input queues of authorised anonymisers/proxies. Each *Anonymiser/proxy* thread will then read IDMEF messages and anonymise them according to the

XACML policy.

Policy decisions are cached in the decision cache to improve the overall efficiency, so that cached decisions which have not timed out will be reused to save the overhead on XACML requests. Different authorised sessions can then have different anonymisation policies based on security level. For example so that a first line outsourced IDS service, which handles the bulk of the alarms, operates with anonymised data; and a second line service operating in-house, can have access to the full alarms. This limits the amount of sensitive information that is visible to the outsourced first-line service.

4.4 Technical Solution

This section performs a more formal analysis of the technical solution. Figure 4.5.1 shows an example IDMEF report that matches the XPath expressions used in the case study and Tables 4.2.1 and 4.2.2 show the formal notations used.

The proposed solution uses the initial XACML authorisation request from the data consumer to return an obligation with a list of $N \geq 0$ XPath expressions identifying XML resources $\{r_1, r_2, ..., r_N\}$ that require further authorisation. This is illustrated in Figure 3.4.2, which shows a successful XACML Response that permits access to the PEP. However the response contains an XACML obligation with a requirement to authorise any XML elements (resources) referenced by the XPath expression */Alert/Additional-Data[@meaning='payload']*

and also a requirement to send the document element matching the scope XPath expression */Alert/Classification/@ident* as a resource attribute in subsequent XACML resource authorisation requests. The PDP can based on this information perform a decision on whether the payload for a given type of IDS alarm is considered privacy violating or not.

The other resource in the initial requests requires authorisation of all XML elements below the XPath expression */Alert/Source/Node/**. The cache specification furthermore requires that the scope XPath expression */Alert/Source/Node/Address/address* is retrieved from the XML document and passed to the XACML policy for evaluation to authorise the resource. Later, the scope

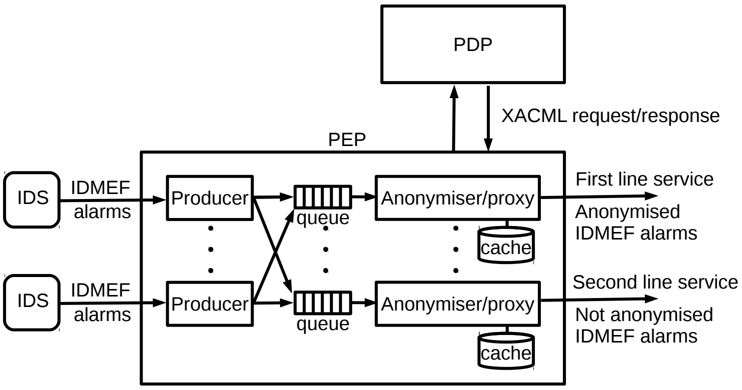


Figure 4.4.1: XACML-based IDMEF anonymiser/proxy with decision cache.

value is also used as part of the cache key for storing cached authorisation decisions.

The XACML obligations service in the Anonymiser/proxy will subsequently perform an XACML authorisation requests the first time a new (uncached) decision for a resource element is identified. The XACML response contains an access control decision from the PDP that will be cached for a retention time period as defined in the obligations of the access control decision.

Caching access control decisions require some knowledge about the authorisation policy being used, since checking for a cache hit requires that all relevant parameter values that the access control decision is based upon are known. These parameter values are, together with the resource id, used as keys when checking whether a cache entry matches the relevant set of parameters in the XML document being checked.

The decision process for XACML authorisation and anonymisation can be considered as a mapping from a resource and a set of parameter values that are required by a given XACML resource policy and to a decision. If this decision is positive, then the decision may have additional obligations, like an obligation to anonymise data or an obligation that expresses authorisation timeout. The parameters required by the system in order to *make a decision* are defined more formally below:

- r_p identifies the set of one or more XML resource(s) to be authorised

by the XACML resource policy p, expressed as an XPath expression on the current XML document, for example:
r_1 =*/Alert/AdditionalData[@meaning='payload']*
r_2=*/Alert/Source/Node/** (applies to any elements below *node*);

- $scope_{p,s}$ are the XPath scope expressions used to extract required parameters for the the XACML policy p and parameter number s.

- $v_{p,q,s}$ are the parameter values extracted from the XML document by applying the XPath scope expression $scope_{p,s}$. These parameter values are required by the XACML policy p in order to evaluate decision number q.

The *decision related* parameters are explained below:

- $a_{p,q}$ is the XACML authorisation, which can be either *Permit* or *Deny*.

- $b_{p,q}$ is the block marker or pattern used to anonymise the data. This parameter is optional, and the default block marker is 'X'.

- $policy_{p,q}$ specifies the anonymisation policy to perform on the content matching resource r_p for decision q, which can be one of a set of predefined anonymisation policies, for example to anonymise by removing or replacing content, anonymise by padding content using a block marker instead of the content (leaves the length of content intact), modify content using regular expression or perform a pseudonymisation policy, for example prefix-preserving pseudonymisation of IP addresses [97], or use an encryption policy.

- $t_{p,q}$ is the absolute time (UTC) when the authorisation decision times out. Different timeout values may be applicable for different authorisations. It is for example natural that authorisations which are based on dynamic variables may need a relatively short timeout period. On the other hand, decisions based on static parameters, like IP address ranges, may not need any timeout value, so the timeout value can be set very large or even infinite. It is then sufficient to have a notification service that can invalidate the policy cache in case the PDP reloads a new policy from the PAP. After $t_{p,q}$ times out, then the cached decision will

```
<IDMEF-Message>
  <Alert messageid="0c18ec3c-1b2e-11e0-99b2">
    <Source spoofed="unknown"
          interface="wlan0">
      <Node category="unknown">
        <Address category="ipv4-addr">
          <address>10.0.2.2</address>
        </Address>
      </Node>
    </Source>
    <Classification ident="1:5976"
        text="SNMP AgentX/tcp request">
    </Classification>
    <AdditionalData type="byte-string"
                 meaning="payload">
      REhDUEM=
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Figure 4.5.1: Simplified excerpt of IDMEF message used in the case study.

be discarded the next time the cache entry is used, and a new XACML authorisation will be performed;

- $\tau_{p,q}$ shows the last time this decision cache entry was used. (This is useful for debugging and optimising the Least Recently Used (LRU) cache.)

With these definitions a decision, denoted by $D_{p,q}$, is represented as a tuple $D_{p,q} = (a_{p,q}, t_{p,q}, \tau_{p,q}, policy_{p,q}, b_{p,q})$ which reflects the q^{th} decision performed by the XACML resource policy number p. The decision cache is implemented as a dictionary where the key $K_{p,q}$ consists of the resource policy number and all n values concatenated i.e.
$p||v_{p,q,1}||v_{p,q,2}||...||v_{p,q,n}$, so that the dictionary indexed on the key returns the cached access decision. The resource policy number p needs to be part of the key to avoid ambiguities between the values, for example that source IP address and destination IP address are being confused for different resource policies.

4.5 XACML Policy Example

This section provides an example of how the IDS XACML profile can be used. It does not focus on the authentication part, which is expected to be very similar to existing federated access control solutions using SAML to convey XACML requests [101]. We assume in the following sections that the XML schema namespace*: http://www.w3.org/2001/XMLSchema#* is denoted by *&xs;* and our own namespace *urn:prile:org:* is denoted by *&prile;*.

In this example, a company considers information about hosts residing on the network 10.0.2.0/24 as sensitive. The company does not want to reveal IP addresses in the IDS alarms. Furthermore, the payload is considered sensitive for certain classes of IDS alarms, as indicated by the *ident* attribute of the *Classification* element in the IDMEF report. IDMEF alarms from IDS sensors on this network can for example look like the simplified IDMEF excerpt in Figure 4.5.1.

4.5.1 Initial Authorisation

The initial XACML request is an ordinary XACML authorisation request to get read access to the Anonymiser/proxy in the PEP, similar to the one described in [133], and is not shown here. However, the XACML response is shown, to illustrate how the PEP is being made aware of the cache parameter specification necessary to manage the decision cache in the form of XACML obligations. The mapping between the notation used in this article and XACML identifiers is shown in Table 4.2.2.

The initial authorisation shown in Figure 3.4.2 returns a set of XML resource identifiers r_p which consists of XPath expressions that cover authorisation of one or more XML elements in the document. Each XACML response also contains s XPath expressions $scope_{p,s}$, which uniquely define the parameters required by the XACML policy to authorise the resources defined by r_p and that will be sent in subsequent XACML resource authorisation requests as resource attributes.

Since an XPath expression may return more than one element, it is then up to the XACML policy to define the attributes so that the cache is kept consistent. The simplest way to do this, is to require that $scope_{p,s}$ is defined to return *only a single element* from the XML document instance being authorised. If an assertion XPath expression returns more than one element, and their result is different, then the evaluation of the policy would also potentially be inconsistent. One element may claim access and the other may not. If it is necessary to do conflict resolution, then all individual assertion

```
<Response>
  <Result ResourceID="PEP">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="&prile;authorize-elements" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;resource:1:id"
          DataType="&xs;string">/Alert/AdditionalData[@meaning='payload']
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:1:assertion:1:scope"
          DataType="&xs;string">/Alert/Classification/@ident
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:2:id"
          DataType="&xs;string">/Alert/Source/Node/*
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:2:assertion:1:scope"
          DataType="&xs;string">/Alert/Source/Node/Address/address
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 4.5.2: XACML reply to initial authorisation of the IDS-PEP.

elements must be passed in to the XACML policy, which defines how the conflict resolution should be done. All XPath expressions from the initial authorisation are precompiled and stored in a two dimensional list indexed by resource number p and scope expression s.

4.5.2 XML Element Authorisation Request

After the initial authorisation, the XML parser of the Anonymiser/proxy in the PEP will get XML messages (IDMEF alarms) from the queue and start parsing them. The PEP then iterates through all XPath matches for all resources in R. If there is no authorisation cached for the XML resource elements r_p refers to, then the PEP will perform XACML authorisation requests for all non-authorised resources, asking for read access to the resource elements. An example authorisation request for an XML element is shown in Figure 4.5.3. The request authorises the subject *soc1@outsourced.example.com* for access to the resource:
r_1 =*/Alert/AdditionalData[@meaning='payload']*.

In addition, the XACML request contains additional resource context parameters representing the set of necessary parameters $scope_{p,s}$ that are required to evaluate the given security policy by the PDP. Here, the first element of the tuple $scope_{p,1}$ =*/Alert/Classification/@ident* refers to the IDMEF Alert classification of the XML message being authorised and $v_{p,1,1} = 1 : 5976$ refers to the unique identification of the alarm class in the XML document being inspected (See Figure 4.5.1). The next section describes how the decision cache works for a cache miss. A cache hit, is subsequently described in Section 4.5.4.

4.5.3 XML Element Authorisation Response

An accepted XACML response is illustrated in Figure 4.5.4. The obligations in XACML responses are mapped as shown in Table 4.2.2.The decision parameters $D_{p,q} = (a_{p,q}, t_{p,q}, \tau_{p,q}, policy_{p,q}, b_{p,q})$ are then being collected by the PEP. All of these except $\tau_{p,q}$ and $t_{p,q}$ are fetched from the obligations in the XACML response. Then $\tau_{p,q}$ is set to the current time and $t_{p,q}$ is set to the timeout value $\Delta t_{p,q}$ in the XACML response plus the current time.

Subsequently, the anonymisation policy $policy_{p,q}$ from the obligations in the XACML response will be applied to the content of all resources matching r_p. This can for example be to anonymise the content by padding it with the block marker *"X"* if $policy_{p,q} = pad - with$ and $b_{p,q}$ =”X”. The anonymisation policy will then be cached in the dictionary using the resource number and parameter values concatenated as key, i.e. $K_{p,q} = p||v_{p,q,1}||v_{p,q,2}||...||v_{p,q,n}$.

If an authorisation request is *denied*, then the XML message will be discarded, since it is not authorised to be sent to the resource consumer.

A *Deny* authorisation decision can be cached in the same way as a *Permit* decision, however this requires that the XACML response includes an obligation with the necessary parameters for the cache entry, as shown in Table 1. The anonymisation policy $policy_{p,q}$ can be omitted in this case, since a *Deny* decision implies that the XML message is dropped. This sequence is not illustrated, since it will be very similar to Figure 4.5.4, except that the decision is changed from *Permit* to *Deny*, and there will typically only be a cache timeout value as parameter.

4.5.4 XML Element Authorisation for Cache Hits

Checking for cache hits is performed for all resources matching the pattern r_p after the necessary scope values $v_{p,q,s}$ have been extracted from the XML document. A cache hit means that there exists a cached decision $q_{p,q}$ for a key $K_{p,q}$ in the decision cache. If the cache has timed out, then entry $q_{p,q}$ is deleted, and a full XACML resource authentication is performed.

Finally, the anonymisation policy $policy_{p,q}$ is enforced and the anonymised XML document is sent to the authorised data consumer.

4.6 Efficiency of the Proposed Solution

The XACML decision cache is implemented in Jython running on Sun Java 6. The Jython interpreter gives a performance overhead, so a native Java implementation can be expected to be somewhat faster, however testing this is left to future work. The implementation uses XimpleWare's Java based Virtual Token Descriptor XML parser (VTD-XML)[1] which has a small memory footprint compared to traditional DOM implementations (1.3-1.5 times the size of the XML document) and also has a very fast XPath 1.0 implementation.

The experiments are performed using Jython 2.2.1 on a 64 bit machine running Ubuntu with 8 Gb ram and 2.53 GHz Intel Core 2 Duo CPU. The decision cache was limited to 3000 entries, using a Least Recently Used (LRU) policy for pruning the cache when it runs full. The cache was tested with between one and thirty relatively simple anonymisation policies that performed simple regular expression match for '.*', i.e. any text content.

The LRU class was implemented in Jython based on the *LinkedHashMap* Java class by overriding the *removeEldestEntry()* method. LRU functionality was then achieved by first retrieving and removing the referenced cached entry and then reinserting it at the tail of the linked hash structure. The oldest entry was then automatically removed from the head of the data structure by *LinkedHashMap* when the cache capacity was exceeded.

The experiment consisted of first identifying a set of resources with corresponding scope values that needs to be cached. 30 resources in the IDMEF

[1] VTD-XML can be found at http://vtd-xml.sourceforge.net

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:1.0:context:schema:os"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context:schema:os
         http://docs.oasis-open.org/xacml/\
         access_control-xacml-1.0-context-schema-os.xsd">
  <Subject>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
               DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
      <AttributeValue>soc1@outsourced.example.com</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
          DataType="&xs;string">
      <AttributeValue>&prile;resource:1:id</AttributeValue>
    </Attribute>
    <Attribute AttributeId="&prile;resource:1:assertion:1:scope"
          DataType="&xs;string">
      <AttributeValue>/alert/classification</AttributeValue>
    </Attribute>
    <Attribute AttributeId="&prile;resource:1:assertion:1:value"
          DataType="&xs;string">
      <AttributeValue>1:5976</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
               DataType="&xs;string">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Figure 4.5.3: XACML request for XML element authorisation.

```
<Response>
  <Result ResourceID="&prile;resource:1:id">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="&prile;element-restrictions"
                  FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;resource:1:cache-timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1D
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:10:policy:pad-with"
          DataType="&xs;string">X</AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 4.5.4: XACML reply to successful XML element authorisation.

reports were selected that it would be reasonable to consider anonymising or that it would be reasonable to consider using as a scope variable for that resource. With the exception of payload, which uses the IDS rule classification as scope (as discussed in this chapter), the rest of the simple rules tested the same parameter they anonymised, amongst others: *source IP address, destination IP address, source port, destination port* etc. We attempted to stress the cache by including scope variables that referred to the TCP sequence and acknowledgement numbers.

We used relatively simple XACML policies in order to see the worst case performance of the decision cache compared to not caching decisions.

A simple XACML policy generator was then used to perform a random selection of n out of these 30 resources, and then test the decision cache on 5000 alarms generated by Snort 2.8 using the standard VRT rule set. Traffic was generated by replaying the 1999 KDD Cup data set[2]. A problem with this data set, is that it does not give a representative picture of the diversity of attack vectors today and also not the diversity of data seen by a large MSS provider. The cache hit rate (97% for 30 enabled rules with 3000 cache entries in the LRU cache) is therefore probably unrealistically high compared to what can be expected with real data. The experiments still give a representative picture of the cache performance, given that the cache hit rate is high.

Each result presented in Figure 4.6.1 is the average of 20 experiments, each anonymising 5000 alarms for a given number of resources n. The experiment was then repeated for $n = (1, 2, ..., 30)$. Using an ensemble of 20 experiments limits the effect of random selection of rules with varying cache hit rates. This makes it possible to better see the underlying trends. Only IDMEF Alert messages was sent to the cache. Heartbeat messages was not processed, since they are not relevant for the anonymisation policy.

Figure 4.6.1 shows the average response time of the decision cache as a function of number of anonymisation policies (i.e. number of XML elements being anonymised). There seems to be a linear relationship between the number of anonymisation policies and the time used, as can be expected. Also, the relative cache efficiency (fraction of uncached to cached time used) increases with increasing number of anonymisation policies, from a speedup

[2] KDD Cup 1999 data (DARPA IDS test set) http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

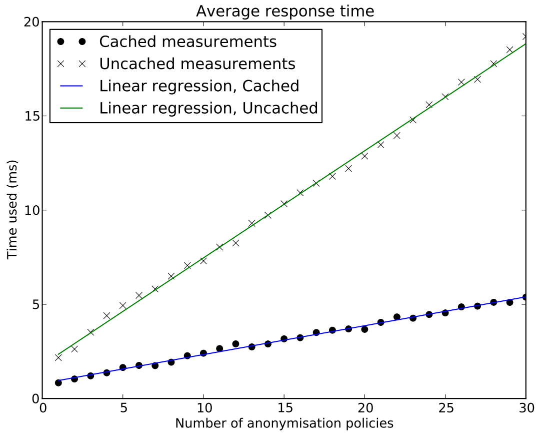


Figure 4.6.1: Average response time of decision cache as a function of number of anonymisation policies. Lines show linear regression of the measurement data.

factor of 2.6-3.0 for less than 5 policies to around 3.5 for 25-30 policies. This shows that the cached solution both performs better in terms of efficiency and scales somewhat better than the uncached solution with increasing number of anonymisation policies. The speedup factor can be expected to be even larger for more complex XACML policies, as long as the cache hit rate is kept sufficiently high.

30 anonymisation policies is probably sufficient for the IDMEF use case. Most of the remaining IDMEF elements and attributes were either constant or varied between a few values, which means they would fit into the cache without causing any significant additional load on the cache. For these 30 anonymisation rules, the decision cache will be able to process up to 185 IDS alarms/s (vs max 52 IDS alarms/s for the uncached solution). If this is not sufficient, then the architecture can easily be parallelised, for example by adding individual anonymising PEPs for each IDS sensor or even splitting traffic from single IDS sensors.

Memory usage is not a problem for the given experiment since the cache had a hit rate of 97% with only 3000 cache entries. The JVM heap size went down to 130 Mb between each garbage collection, and memory increased slowly after garbage collection, which is another indication that memory usage was not problematic when using the VTD-XML parser[3]. However, more

[3] This picture was however different for Javas standard DOM implementation, which showed heavy memory allocation/free patterns.

realistic data (for example from a MSS provider) are needed to verify that memory usage is not a problem.

The decision cache is in other words useful for increasing both the performance and scalability of XACML authorisations. This means that it should be viable to perform fine-grained access control of XML elements and attributes in IDMEF alarms from IDS by using an anonymising decision cache.

4.7 Related Work

The content of this chapter was initially published in [135]. This chapter extends and elaborates on the simple XACML policy for anonymisation proposed in the previous chapter. The previous chapter presented the idea of anonymisation based on an XACML obligations service for coarse-grained access control of IDMEF messages. This chapter extends the solution to provide fine-grained access control of XML messages in general with decision caching support and support for several different anonymisation policies.

There is as far as we are aware of no other similar solutions. However, some other systems cover part of the same functionality. A solution for controlling access to XML documents is proposed in [35]. However, this solution is not based on XACML and it does not support anonymisation policies. An XACML-based privacy-centred access control system is proposed in [8, 7]. This system focuses on credential management to provide users control over their data. Our solution is different, since it proposes an XACML caching solution with fine-grained access control and anonymisation of data.

An extension of XACML to improve the performance of decision making processes when dealing with stable conditions is explained in [76]. This solution aims at reducing the time that the Policy Information Point (PIP) uses for accessing remote services like SNMP agents and also the decision making time. Our solution is different, since it aims at performing access control of individual elements and attributes in XML documents using a decision cache based solution.

The BRO IDS [78] supports a way to anonymise the payload of a packet instead of removing the entire payload [98, 78]. There also exists some earlier work on privacy-enhanced host-based IDS systems that pseudonymises audit data and performs analysis on the pseudonymised audit records [64, 123, 50, 124, 104]. However neither of these solutions are based on XACML or provide native authorisation and anonymisation of XML document instances.

4.8 Conclusion and Future Work

The chapter proposes a viable solution for fine-grained XACML authorisation and anonymisation of elements and attributes in XML documents or messages. This allows for central management of authorisation and anonymisation policies for XML documents and IDMEF IDS alarms instead of using a hybrid solution with several different access control solutions or languages.

The decision caching protocol can easily be adapted to other authorisation schemes by choosing a different cache key generation scheme that reflects the authorisation scenario. Caching can then be enabled by adding the timeout parameter as an obligation in order to manage the cached decisions. This opens up a possibility to significantly improve the efficiency and scalability of other XACML based authorisation schemes.

A potential critique of the proposed solution, is that fine-grained access control decisions are delegated from the PDP to the PEP via XACML obligations. This violates the clear interface between policy authorisation and policy enforcement.

Future work involves adding more functionality and if necessary moving time critical parts to Java. It would also be interesting to support the Multiple Resources Profile of XACML in order to process several resources simultaneously by XACML. Last but not least, the anonymising decision cache should be tested under realistic conditions at a MSS provider.

Chapter 5

Reversible Anonymisation for XML-based Services

This chapter proposes a reversible anonymisation scheme for XML messages that supports fine-grained enforcement of XACML-based privacy policies. Reversible anonymisation means that information in XML messages is anonymised, however the information required to reverse the anonymisation is cryptographically protected in the messages. The policy can control access down to octet ranges of individual elements or attributes in XML messages. The reversible anonymisation protocol effectively implements a multi-level privacy and security based approach, so that only authorised stakeholders can disclose confidential information up to the privacy or security level they are authorised for. The approach furthermore supports a shared secret based scheme, where stakeholders need to agree to disclose confidential information. Last, it supports time limited access to private or confidential information. This opens up for improved control of access to private or confidential information in XML messages used by a service oriented architecture. The solution provides horizontally scalable confidentiality protection for certain types of big data applications, like XML databases, secure logging and data retention repositories.

5.1 Introduction

This chapter proposes an innovative approach for reversible anonymisation of private or confidential information in XML messages. It extends the XACML based decision cache and anonymiser for XML documentsin the previous chapter with support for reversible anonymisation of private or confidential information based on broadcast encryption. An advantage compared to the original approach, is that it supports multi-level privacy for confidential content in the XML messages, so that only authorised parties can access this information. This simplifies handling of data with multiple security levels, since these can be stored together, protected by encryption. It is assumed that all connections have basic security (e.g. encrypted using TLS/SSL), to avoid cleartext attacks on underlying communication channels.

The approach allows for both confidentiality and integrity protection of the XML data based on XML encryption. Such a solution can for example be useful for secure logging, data retention, protecting private or confidential information in SmartGrid-based systems (e.g. Demand-Response systems) or for Security Incident and Event Management (SIEM) systems. The approach furthermore supports location-aware authorisation and anonymisation of data, by using the GeoXACML framework [6].

The reversible anonymisation enforcement scheme has been successfully demonstrated for a Security Incident and Event Management (SIEM) system anonymising XML-based Intrusion Detection System (IDS) alarms in the Intrusion Detection Message Exchange Format (IDMEF) [62]. This means that the proposed approach can be used to implement privacy-enhanced IDS services. The proposed scheme is general, and it is envisaged that it in the future will be integrated in an Enterprise Service Bus (ESB), to provide on-demand policy controlled reversible anonymisation of information in any XML-based web service.

The original use case for the reversible anonymisation scheme is privacy-enhanced intrusion detection system based services. It can be noted that the IDMEF-based IDS alarm format is a semi-structured XML format, that supports arbitrary extensions via the IDMEF AdditionalData construct. Performing data mining of such semi-structured data can be a challenge with existing relational databases, meaning that a non-relational data representation may be required for efficient data processing. Horizontal scalability is typically required for efficient processing of IDS alarms. Horizontal scalability for IDS is often implemented by having local data repositories for IDS alarms, and using an event correlation system to reduce the number of IDS alarms sent for central processing by a Security Operations Centre. The proposed reversible anonymisation scheme supports such a use case by being inherently parallelisable, so that IDS alarms from different sensors can be processed by individual anonymisers.

The proposed solution can be considered as security control extensions for XML databases that include shared secret based authorisation (which can be used as a building block for multifactor authentication), data encryption and anonymisation. This can be used to address some of the privacy concerns on big data repositories based on XML databases. It can also be used to increase the transparency of anonymised services by supporting secure logging schemes. This means that the approach can be a first step towards solving the transparency paradox - that big data operators pervasively collect all manner of private information, however with the operations of big data itself being almost entirely shrouded in legal and commercial secrecy [72].

The chapter is organised as follows: The next section discusses the reversible anonymisation scheme, including notation used, background and motivation for implementing reversible anonymisation and a high-level description of the reversible anonymisation process. Section 5.3 explains the reversible anonymisation process in detail, focusing on XACML authorisations, detailed algorithm for anonymisation and deanonymisation and integrity checks. Section 5.4 describes how the reversible anonymisation protocol is extended to support a default DENY anonymisation scheme. Section 5.5 describes the adaptations required to support key sharing, and section 5.6 describes how time-based data expiry can be implemented, in order to support time-limited data retention. Section 5.8 describes the results from performance tests of the anonymiser and the deanonymiser, and section 5.9 discusses advantages and disadvantages with the proposed approach. Section 5.10 discusses related work, section 5.11 concludes the chapter and section 5.12 outlines future work.

5.2 Reversible Anonymisation Scheme

This section first describes the background and motivation for the reversible anonymisation scheme proposed in this chapter , and then outlines how the anonymisation process works from a high level perspective. The reversible anonymisation scheme is implemented as an extension of the anonymising decision cache described in the previous chapter.

5.2.1 Background and Motivation

The reversible anonymisation scheme is useful for dynamically configuring confidentiality protection of XML-based web services in a service oriented architecture. This provides a possibility to enforce the security and privacy of existing services by running these services through the anonymiser. Authorised users or services can subsequently deanonymise and use information in security levels they have clearance for. This provides a flexible, policy driven protection scheme for private or confidential information, where protection mechanisms can be added on demand.

General functionality that the reversible anonymisation protocol provides, is irreversible and reversible anonymisation of information in XML messages controlled by XACML policies. In addition, it supports key sharing, which can be used to enforce separation of duties constraints - for example so that different stakeholders need to agree to disclose confidential information to reduce the risk of insider attacks. The scheme can also be used to implement trustworthy deployment of system configurations. The approach furthermore supports time-limited access to sensitive data, which can be used to support data retention and secure logging mechanisms for XML databases.

It is expected that such a general policy-driven reversible anonymisation scheme will be useful in a range of different use cases, including outsourcing - for example to cloud-based services, e-health, e-commerce, critical infrastructures and managed security services, which is the practical use case considered in this chapter .

Reversible anonymisation here means that sensitive information in the XML messages is anonymised, however necessary information required to reverse the anonymisation process is stored encrypted in the XML message.

Reversible anonymisation is different from traditional pseudonymisation since no pseudonym is used to replace the anonymised XML element. Instead the encrypted information required to reverse the anonymisation is added to the XML message. There are several advantages by using such a strategy compared to a traditional pseudonymisation strategy. First, this means that there is no need to consider the cryptographic strength of a pseudonymisation scheme to avoid linkability between pseudonyms, since pseudonyms are not used. Second, the anonymised data can be any data, for example an informative text, replacing the anonymised data. This informative text may even contain nonsensitive parts of the original data (for example the most significant part of an IP range)[1]. Third, having larger chunks of encrypted data reduces the risk of leaking information via traffic analysis.

The reversible anonymisation scheme builds key distribution into the XACML policies, so that the Policy Enforcement Point (PEP) queries the Policy Decision Point (PDP) about which public keys that are authorised to access information in the XML messages.

It is assumed that the Security Assertion Markup Language (SAML) or similar is used both to authorise the data consumer to receive anonymised messages, and also to authorise individual users for access to sensitive information in the messages. SAML supports automatic logout of users by using the single logout protocol of SAML 2.0 [110]. All authorisation decisions should in addition be logged, to know which user that had access to the messages when.

5.2.2 Outline of the Reversible Anonymisation Process

The cryptographic problem that must be solved, is how to cryptographically protect confidential information, so that only authorised personnel can access the information on a needs basis. It is assumed that information may be split into $d > 1$ different security levels (for example Restricted, Confidential, Secret). There may furthermore be one or more consumers of XML messages, where each consumer is authorised to a given subset of security levels. The solution proposed here does not assume any semantics or rela-

[1] Note however that such a strategy should be used with care for private or confidential data to avoid reducing the anonymity set unduly for the underlying data.

Index	Meaning
d	Number of security levels.
i	Security level
j	Share index of encryption key $K_{i,j}$.
N	Number of resources.
n	Number of key shares.
m	Number of users.
p	XPath expression number.
q	Match number for XPath expression p.
u	User number.
z	Number of matches of r_p.

Type	Meaning
C	Set of classified XML elements.
$\underline{e_p}$	$\underline{e_p} = (e_{p,1}, e_{p,2}, ..., e_{p,z})$ identified by XPath resource r_p on *message*.
$e_{p,q}$	XML element q identified by evaluating XPath resource r_p on *message*.
K_i	Encryption key for security level i.
KM	Set of keymap tuples consisting of encryption key K_i and security level i.
l_i	Information on security level i.
$\underline{L}$	Vector of all confidential information l_i.
Λ_u	Keymaps user u is authorised for.
AM	Authorisation map from public keys PK_u to the set of keymaps $\Lambda_u \subseteq KM$ the user u is authorised for.
PK_u	Public key of user u.
r_p	XPath expression for p identifying resources that need authorisation.
R	Set of all r_p.
S	Set of encrypted keys.
SK_u	Secret key of user u.
t_{exp}	Key expiry time.
$t_{retention}$	Data retention time.
Q	Matrix of authorised elements $e_{p,q}$ for resource p and security level i.

Table 5.2.1: List of notations for reversible anonymisation scheme.

Mapping of XACML Parameters

Parameter	Decision cache XACML AttributeId
Declassify	urn:prile:org:resource:*p*:policy:declassify Obligation to declassify information for policy p.
l_i	level:*i*: authorisation level.
l_i related	level:*i*:share:*j*: share index j of split encryption key K_i.
K_i related	urn:prile:org:encryption-key:level:*i*:algorithm encryption key algorithm e.g: http://www.w3.org/2001/04/xmlenc#aes128-cbc
K_i related	urn:prile:org:encryption-key:level:*i*:timeout timeout value before encryption key regeneration.
XML namespace	urn:prile:org:xmlns:*nsx* declares the XML namespace *nsx*.
PK_u	urn:prile:org:public-key:*u*:pem-key base64-encoded public key of user u.
PK_u related	urn:prile:org:public-key:*u*:algorithm public key algorithm used, e.g: http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p
PK_u related	urn:prile:org:public-key:*u*:user Owner of u of key pair (PK_u, SK_u)
PK_u related	urn:prile:org:public-key:*u*:ephemeral-reference Reference to ephemeral key EK_u, e.g. ephemeral:1;
PK_u related	urn:prile:org:public-key:*u*:levels comma separated list of security levels PK_u is authorised for.
PK_u related	urn:prile:org:public-key:*u*:timeout optional authorisation timeout value for PK_u.
Policy type	urn:prile:org:default-policy:policytype policytype={PERMIT|DENY}
r_p	urn:prile:org:resource:*p*:id resource identifying XPath expression.
$t_{retention}$	urn:prile:org:encryption-key:level:*i*:retention-time data retention time.

Table 5.2.2: Mapping of XACML parameters for reversible anonymisation scheme.

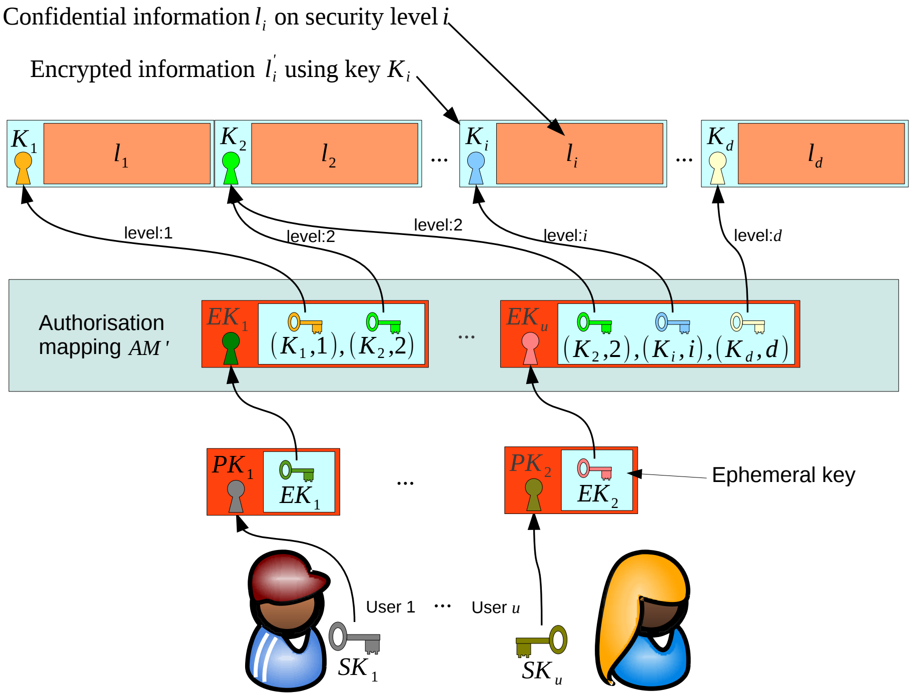


Figure 5.2.1: Overview over encryption scheme used to implement reversible anonymisation.

tionships between the security levels, apart from controlling who have access to which security levels. It is assumed that additional semantics, e.g. that access to level Secret also includes access to level Confidential and Restricted, can be enforced by the XACML policies controlling the anonymisation policy. Some stakeholders, for example law enforcement or CERT, may have a need to access information with a higher level of sensitivity than a first-line service. It is also desirable to enforce a shared secret scheme to ensure that two or more parties must agree by providing their key shares before sensitive information is disclosed (for example that a data controller and law enforcement must agree to disclose a given set of sensitive data).

The following notation is used in the chapter . A pseudo-random number generator is denoted by $rnd()$, an encryption function is denoted by $Enc(key, value)$ and decryption function by $Dec(key, value)$. Furthermore, $value'$ denotes the encrypted or anonymised value, i.e. $value' = Enc(key, value)$ and $value = Dec(key, value')$. The list of notation used in the chapter can be found in Table 5.2.1.

Fig. 5.2.1 gives overview over how the encryption scheme used to implement reversible anonymisation and multi-level security is implemented. The confidential information l_i in each security level $i \in \{1,...,d\}$ consists of both a specification on how to reverse the anonymisation using XPath expressions and the confidential information identified using these XPath expressions. The information in each security level l_i is encrypted using the corresponding encryption key K_i for the security level referenced by the security level index (or label) i. In this chapter it is assumed for simplicity that the security level index is a natural number[2].

Furthermore, users can be authorised to a subset of all security levels. In Fig. 5.2.1, User 1 has access to security levels 1 and 2, and User u has access to security levels 2, i and d. The authorisation mapping (AM) for each user is encrypted using a two-stage process, where a symmetric key EK_u is used to encrypt the AM, and the public key PK_u of the authorised user u is used to encrypt the EK_u. EK_u is regenerated each time a user is authorised for access to the data, and has a configurable key renegotiation timeout. The authorisation mapping, encryption keys and the number of security levels are controlled by XACML policies. An advantage by using a separate key for encrypting the key mapping, is to avoid having to use relatively slow public key cryptography for encrypting the AM. Another advantage is that standardised XML Encryption methods can be used for key wrapping.

A given user u, can then decrypt the EK_u using her private key SK_u, which in turn can be used to decrypt the tuples (K_i, i) that the user is authorised for. The index i can subsequently be used to retrieve the corresponding encrypted information $l_i^{'}$ which then can be decrypted using the corresponding key K_i. The confidential information in l_i both contains the confidential data that is anonymised in the original XML data and a specification (XPath expressions) that describes how the anonymisation for the given security level can be reversed. The approach used for encrypting information in security levels is similar to broadcast encryption [48].

This approach gives flexibility for authorising access to information on a given security level according to operative needs. It may for example be desirable to enforce separation of duties between information considered secret

[2] However it may also be implemented using textual labels (e.g. “secret”) if the underlying data structure is implemented as an associative array.

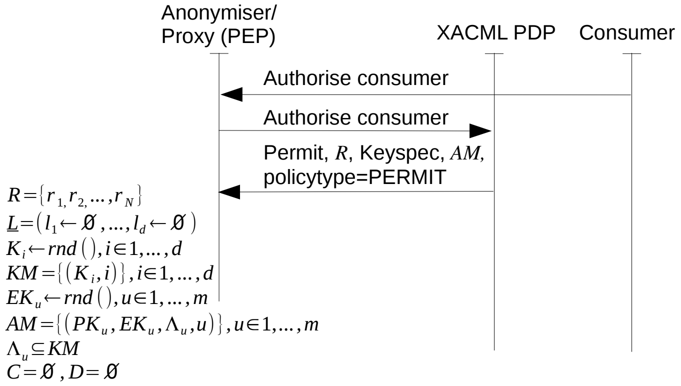

Anonymiser/
Proxy (PEP)
XACML PDP
Consumer
Authorise consumer
Authorise consumer
Permit, R, Keyspec, AM,
policytype=PERMIT
$R=\{r_1, r_2, ..., r_N\}$
$\underline{L}=(l_1 \leftarrow \emptyset, ..., l_d \leftarrow \emptyset)$
$K_i \leftarrow rnd(), i \in 1, ..., d$
$KM=\{(K_i, i)\}, i \in 1, ..., d$
$EK_u \leftarrow rnd(), u \in 1, ..., m$
$AM=\{(PK_u, EK_u, \Lambda_u, u)\}, u \in 1, ..., m$
$\Lambda_u \subseteq KM$
$C=\emptyset, D=\emptyset$

Figure 5.2.2: Initial authorisation sequence.

by customer A and information considered secret by another customer B, by authorising different trusted CERT teams to access this information in each organisation.

5.2.3 Reversible Anonymisation Protocol

This subsection describes the reversible anonymisation protocol. The detailed description of the algorithm is given in Fig. 5.3.8.

Initially, the consumer sends a SAML assertion with proof of authenticity to log in to the Anonymiser/Proxy PEP, as shown in Fig. 5.2.2. The PEP will then ask the XACML PDP for authorisation of the consumer. If the response is *Permit*, then the XACML response will contain a set of obligations that amongst others contain the set of $N \geq 0$ resource identifying expressions (XPath expressions) $R = \{r_1, r_2, ..., r_N\}$ which identify information that needs authorisation. The reply furthermore contains the Keyspec, which specify keys such as the public keys PK_u, and information on how to generate ephemeral keys EK_u, $u \in \{1, ..., m\}$. It also specifies the number of security levels d as well as how to generate the unique encryption keys K_i, $i \in \{1, ..., d\}$ that are used. Then the vector of confidential information $\underline{L}$ is initialised to the empty vector for all security levels, i.e: $\underline{L} = (l_1 \leftarrow \emptyset, ..., l_d \leftarrow \emptyset)$, and the encryption keys are initialised to a random number using the specified key generation algorithm, i.e: $K_i \leftarrow rnd()$, $i \in \{1, ..., d\}$.

The encryption keys are used to generate a set of keymaps $KM = \{KM_i | i \in$

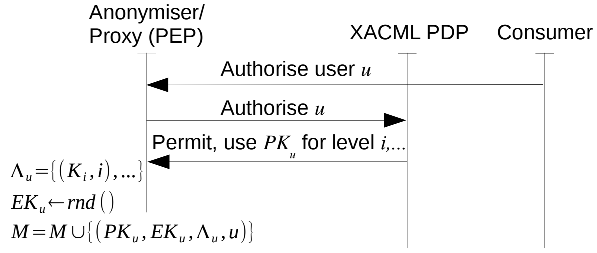

Anonymiser/
Proxy (PEP)
XACML PDP
Consumer
Authorise user *u*
Authorise *u*
Permit, use PK_u for level *i*,...
$\Lambda_u = \{(K_i, i), ...\}$
$EK_u \leftarrow rnd()$
$M = M \cup \{(PK_u, EK_u, \Lambda_u, u)\}$

Figure 5.2.3: Authorise user.

$1,...,d\}$, where each keymap $KM_i = (K_i, i)$ is a tuple consisting of the encryption key K_i and an index i referring to the confidential information l_i on security level i. The ephemeral keys EK_u for each $u \in \{1,...,m\}$ are subsequently generated.

The initial XACML response also contains a key authorisation mapping $AM = \{(PK_u, EK_u, \Lambda_u, u) | u \in 1,...,m\}$, which describes who (i.e. which user u's public keys PK_u) that are authorised to access data at which security levels (via the ephemeral keys EK_u). Here $\Lambda_u \subseteq KM$ is the subset of keymaps indicating which security levels i the user is authorised for.

It is assumed that deployment of XACML policies describing the key mapping is being controlled by the information owner, so that the key owners themselves will not be allowed to modify this mapping. Finally, the set of classified elements, denoted as C, and elements explicitly declassified, denoted as D, are initialised to the empty set. The relative complement $C \backslash D$ describes the set of XML elements or attributes that needs to be anonymised.

The process of subsequent authorisation of a user u is shown in Fig. 5.2.3. The user asks the Anonymiser to be authorised for access to a set of one (or more) security level(s) $L_u \subseteq \{1,...,d\}$. If the analyst has access according to the security policy, then the XACML responds with *Permit*. The response contains an obligation with the public key of the analyst PK_u and an authorisation mapping showing that this analyst is authorised for the set of security levels L_u. The PEP first generates the ephemeral key EK_u and then adds this decision to the key authorisation mapping: $AM = AM \cup \{(PK_u, EK_u, \Lambda_u, u)\}$, where $\Lambda_u = \{KM_i | \, i \in L_u\}$ so that the user is authorised for accessing confidential information in l_i.

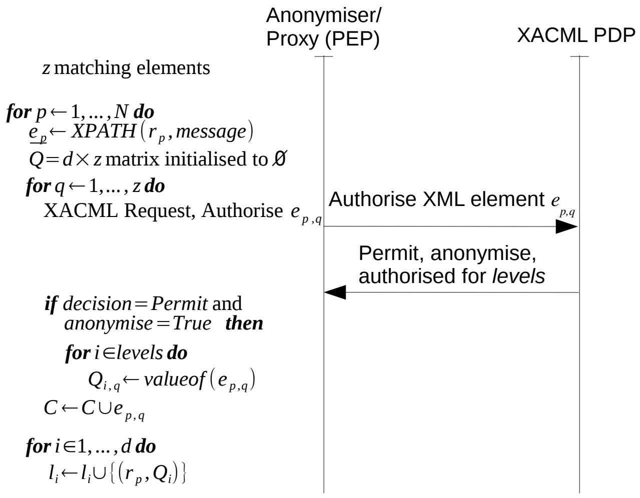

Anonymiser/ Proxy (PEP)
XACML PDP
z matching elements
for p ← 1,...,N do
e_p ← XPATH(r_p, message)
Q = d × z matrix initialised to ∅
for q ← 1,...,z do
XACML Request, Authorise e_p,q
Authorise XML element e_p,q
Permit, anonymise, authorised for levels
if decision = Permit and anonymise = True then
for i ∈ levels do
Q_i,q ← valueof(e_p,q)
C ← C ∪ e_p,q
for i ∈ 1,...,d do
l_i ← l_i ∪ {(r_p, Q_i)}

Figure 5.2.4: XML Element Authorisation

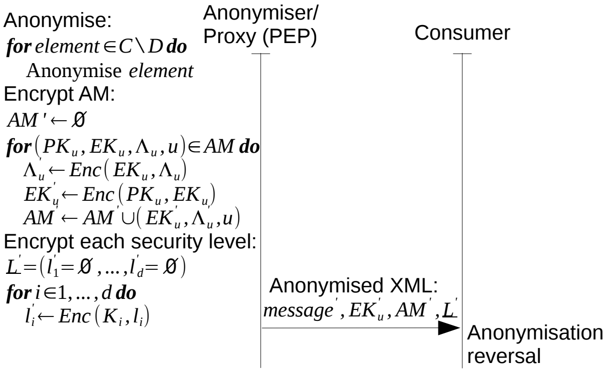
After that, the authorisation of elements and attributes in the XML message starts, as shown in Fig. 5.2.4. First each resource identifying XPath expression r_p, $p \in \{1,...,N\}$ is evaluated on the XML message to get a vector of z matching XML elements:

$$\underline{e_p} = (e_{p,1}, e_{p,2}, ..., e_{p,z}) \leftarrow XPATH(r_p, message). \qquad (5.2.1)$$

The PEP then needs to authorise each of the elements in $\underline{e_p}$ by querying the PDP for which security level(s) each element is authorised for. If access to the element $e_{p,q}$ is granted, then the XACML Response contains a *Permit* decision with an obligation to anonymise the given element which also contains which security level(s), denoted *levels*, that are authorised to reverse the anonymisation.

The anonymiser then iterates through all levels $i \in levels$ and stores the value of each element $e_{p,q}$ in a matrix $Q_{i,q}$, so that each row vector Q_i contains the confidential elements of $\underline{e_p}$ that security level i is authorised for, and $\emptyset$ otherwise. After that, the current element $e_{p,q}$ is added to the list of classified elements C, i.e: $C \leftarrow C \cup e_{p,q}$, that later will be anonymised.

The last part of element authorisation is to iterate through all security levels $i \in \{1,...,d\}$ and add a tuple (r_p, Q_i), consisting of resource identifier r_p and confidential information for the given resource identifier Q_i to the set



```
Anonymise:
$\textbf{for}\ element \in C \setminus D\ \textbf{do}$
    Anonymise $element$
Encrypt AM:
$AM' \leftarrow \emptyset$
$\textbf{for}\ (PK_u, EK_u, \Lambda_u, u) \in AM\ \textbf{do}$
    $\Lambda_u^{'} \leftarrow Enc(EK_u, \Lambda_u)$
    $EK_u^{'} \leftarrow Enc(PK_u, EK_u)$
    $AM^{'} \leftarrow AM^{'} \cup (EK_u^{'}, \Lambda_u^{'}, u)$
Encrypt each security level:
$\underline{L}^{'} = (l_1^{'} = \emptyset, \ldots, l_d^{'} = \emptyset)$
$\textbf{for}\ i \in 1, \ldots, d\ \textbf{do}$
    $l_i^{'} \leftarrow Enc(K_i, l_i)$
```

Figure 5.2.5: Anonymisation enforcement and reversal for default PERMIT policies.

of confidential information l_i for security level i, i.e: $l_i \leftarrow l_i \cup \{(r_p, Q_i)\}$. This essentially means that a specification on how to undo the anonymisation has been stored in l_i.

When all elements matching the set of resources R have been authorised, the algorithm proceeds to the enforcement part, as shown in Fig. 5.2.5. Enforcement starts with anonymising all classified elements in C that have not been explicitly declassified (that are not part of D) i.e. $C \backslash D$. The anonymiser then loops through the authorisation mapping tuples in AM, and encrypts Λ_u with EK_u, and then encrypts EK_u using the user u's public key PK_u. After that, the anonymiser adds a tuple consisting of the encrypted ephemeral key $EK_u^{'} \leftarrow enc(PK_u, EK_u)$, the encrypted encryption keys and security level references $\Lambda_u^{'} \leftarrow enc(EK_u, \Lambda_u)$ and a reference to the user u that can decrypt the authorisation mapping to the set of encrypted authorisation maps $AM^{'}$, i.e: $AM^{'} \leftarrow AM^{'} \cup \{(EK_u^{'}, \Lambda_u^{'}, u)\}$.

Finally, the confidential information l_i in each security level i is encrypted using the respective symmetric encryption key K_i, to protect the sensitive information from disclosure by unauthorised parties. The encrypted authorisation mapping $AM^{'}$, the ephemeral key(s) $EK_u^{'}$ and the vector of encrypted security levels $\underline{L}^{'} = \{l_1^{'}, \ldots, l_d^{'}\}$ are finally enveloped in an IDMEF AdditionalData element of $message^{'}$. The anonymised anonymised IDMEF XML message ($message^{'}$) is then sent to the receiver. It must be noted that the proposed solution is not restricted to IDMEF based IDS alarms. It works for any XML schema that supports an extension mechanism where the encrypted data and

XML signatures can be inserted.

On the receiving side, the deanonymiser goes through the $(\Lambda_u^{'}, u)$ tuples in $AM^{'}$ to search for a user u that matches the current $user$ as shown in Fig. 5.3.10. If this is found and decryption of the encrypted ephemeral key $EK_u^{'}$ using the secret key SK_{user} succeeds, then the deanonymiser will decrypt Λ_u using EK_u, and goes through all key maps $(K_i, i) \in \Lambda_u$ and decrypt the confidential information at the given security level: $l_i \leftarrow Dec(K_i, l_i^{'})$.

The deanonymiser can then loop through all tuples $(r_p, Q_i) \in l_i$, use XPath searches with the expression r_p on $message^{'}$ to retrieve the elements that need to be deanonymised, i.e. $\underline{e_p} \leftarrow XPATH(r_p, message^{'})$ and finally loop through all elements q of $\underline{e_p}$ to replace the original content using $e_{p,q} \leftarrow Q_{i,q}$ if $Q_{i,q} \neq \emptyset$, which reverses the anonymisation.

The XACML policy allows the encryption keys to be regenerated at regular time intervals, to reduce the risk of key recovery attacks and also to reduce the amount of confidential information that can be accessed with a given encryption key.

The confidential information is stored in random order, using random identifiers to avoid revealing explicitly which security level (or grading) the confidential information has. Each security level in addition contains a nonce (not shown in the figures), to make it harder to correlate sensitive information between IDS alarms. This nonce can also be used by the deanonymiser to detect and avoid data replay attacks.

5.2.4 XML Signature Based Integrity Checks

XML Signatures are added to ensure the integrity of the XML messages both before and after anonymisation, to ensure that the message has not been tampered with and also that the anonymisation reversal works correctly. Before anonymisation, an enveloped XML Signature[3] (the inner signature) is calculated over the original XML message [45]. This signature is added to verify the integrity of the XML message after successful anonymisation reversal of all anonymised elements. This acts as a regression test to verify correct operation of reversible anonymisation, and also verifies that the data used to

[3] Enveloped means that the signature is embedded as part of the message itself, instead of being added outside the definition of the message.

```
<IDMEF-Message>
  <Alert messageid="0c18ec3c-1b2e-11e0-99b2">
    <Source spoofed="unknown"
          interface="wlan0">
      <Node category="unknown">
        <Address category="ipv4-addr">
          <address>10.0.2.2</address>
        </Address>
      </Node>
    </Source>
    <Classification ident="1:5976"
        text="SNMP AgentX/tcp request">
    </Classification>
    <AdditionalData type="byte-string"
                 meaning="payload">
      Payload data...
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Figure 5.3.1: Simplified excerpt of IDMEF message used in the case study.

reverse the anonymisation have not been tampered with.

In addition, an enveloped XML Signature, the outer signature, is calculated over the anonymised message, including the encrypted section, to verify the integrity of the XML message before decryption operation starts.

5.3 Detailed Specification of Reversible Anonymisation Protocol

This Section contains a detailed specification of the reversible anonymisation protocol. It focuses on XACML policy interaction and formats required by the protocol, and also on how XML Signatures and Encryption are being used.

In the following, the XACML namespace is denoted as *&xacml*;, the XML Schema namespace is denoted as *&xs;,* the XML encryption schema is denoted by *&xenc;* and our own extensions are defined in the namespace *http://www.prile.org:,* denoted by *&prile*;. It is assumed that the reader has a basic understanding of XACML.

5.3.1 Initial Authorisation

The initial authorisation example assumes the IDMEF message shown in Fig. 5.3.1. The detailed steps of the anonymisation algorithm is shown in Fig.

5.3.8, and reversal of the anonymisation in Fig. 5.3.10.

The initial XACML authorisation is a standard XACML authorisation request with Subject attribute value *soc1@outsourced.example.com*, Resource attribute value *PEP* and Action value *authorise*. The response to this initial request is illustrated in Fig. 5.3.2. The response will for successful authorisation contain decision *Permit* with an obligation containing a cache specification with the elements that need to be authorised/anonymised. The cache specification for individual elements use the same scheme as [135].

The cache specification contains obligations defining a vector of d encryption keys $K = \{K_1, K_2, ..., K_d\}$, one encryption key for each security level i. It also contains obligations defining a set of m ephemeral keys $EK_u, u \in \{1, ..., m\}$, one for each authorised user. The obligation defines the encryption key *algorithm* and a *timeout* parameter which defines when the encryption key times out and a new encryption key must be generated. Generating a new encryption key is done automatically on timeout. The encryption key timeout can be changed by reinitialising the anonymiser, which means that the authorisation mapping AM also will be updated. The XACML AttributeIDs for the encryption key are as follows:

- *&prile;encryption-key:level:i:algorithm* encryption algorithm for K_i or EK_u in XML Encryption URL format;

- *&prile;encryption-key:level:i:timeout* timeout value of encryption key in W3C XQuery dayTimeDuration format.

Last, the XACML response will contain obligations defining a set of zero or more public keys that define which stakeholders (e.g. CERTs or law enforcement) that by default are authorised for access to sensitive information in the IDS alarms. Furthermore the response contains an XACML attribute specifying which security level(s) each key is authorised for, to create the authentication map AM from keys to levels, as shown in line 3 of Algorithm 5.3.8. Each of these obligations contain the following XACML Attributes:

- *&prile;public-key:u:pem-key* - base64-encoded public key PK_u in SSL PEM format[4];

[4] The protocol can easily be extended to also support X509 certificates to support chains of trust. It has not been considered important do demonstrate this for the proof-of-concept prototype, so implementing this is left as future work.

- *&prile;public-key:u:algorithm* showing the public key algorithm used in XML Encryption URL format;
- *&prile;public-key:u:user* indicating the owner of the key;
- *&prile;public-key:u:ephemeral-reference* refers to the ID of the ephemeral key, e.g. *ephemeral:1*;
- *&prile;public-key:u:levels* specifying a comma separated list of which security levels, or key shares below a given level this public key is valid for. Levels are addressed as a comma separated list of statements: *level:1, level:2 etc.* Key shares within a given level is addressed as *level:1:share:1, level:1:share:2* etc.;
- and last *&prile;public-key:u:timeout* specifying the timeout value of the public key, after which the public key needs reauthorisation. The timeout value is specified in W3C XQuery dayTimeDuration format.

5.3.2 Subsequent Authorisations

It is possible to authorise additional parties on a needs basis after the initial authorisation, as shown in Fig. 5.3.3. This is done using an XACML request with Subject attribute being the name of the user to be authorised (here *analyst1@outsourced.example.com*), the Resource attribute specifies the desired access level *(level:1),* and the action is *read.*

The subsequent XACML Response is shown in Fig. 5.3.4. If the user is authorised for access to the given security level (here level 1), then the XACML response will contain a *Permit* decision with an obligation conveying the public key of the user, ephemeral key definition and the security levels this key is authorised for. The format of this obligation is the same as the format of the public-key obligation in the initial XACML Response, except that this obligation also may contain the encryption-key timeout attribute in order to force reauthentication and regeneration of the encryption key. If the timeout attribute is present, then the authorisation will have the effect that the newly authorised user for example cannot access sensitive information analysed by other users during their work shifts. If attack analysis of data outside the authorisation window of the user is required, then this can be done by

```
<Response>
  <Result ResourceID="PEP">
    <Decision>Permit</Decision>
    <Status><StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/></Status>
    <Obligations>
      <Obligation ObligationId="&prile;default-policy" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;policytype:PERMIT"
         DataType="&xs;string">
        </AttributeAssignment>
      </Obligation>
      <Obligation ObligationId="&prile;authorize-elements" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;resource:1:id"
          DataType="&xs;string">/Alert/AdditionalData[@meaning='payload']
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:1:assertion:1:scope"
          DataType="&xs;string">/Alert/Classification/@ident
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:2:id"
          DataType="&xs;string">/Alert/Source/Node/*
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:2:assertion:1:scope"
          DataType="&xs;string">/Alert/Source/Node/Address/address
        </AttributeAssignment>
      </Obligation>
      <Obligation ObligationId="&prile;encryption-key:level:1" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;encryption-key:level:1:algorithm"
          DataType="&xs;string">http://www.w3.org/2001/04/xmlenc#aes128-cbc
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;encryption-key:level:1:timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1D
        </AttributeAssignment>
      </Obligation>
      <Obligation ObligationId="&prile;encryption-key:ephemeral:1" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;encryption-key:ephemeral:1:algorithm"
          DataType="&xs;string">http://www.w3.org/2001/04/xmlenc#aes128-cbc
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;encryption-key:ephemeral:1:timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1h
        </AttributeAssignment>
      </Obligation>
      <Obligation ObligationId="&prile;public-key:1"
                  FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;public-key:1:pem-key"
          DataType="&xs;string">-----BEGIN PUBLIC KEY----..........
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:1:ephemeral-reference"
          DataType="http://www.w3.org/2001/XMLSchema#string">ephemeral:1
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:1:algorithm"
          DataType="&xs;anyURI">http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:1:user"
          DataType="&xs;string">CERT
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:1:levels"
          DataType="&xs;string">level:1,level:2</AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:1:timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P365D
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 5.3.2: Initial XACML response to support default keys.

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:1.0:context:schema:os"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context:schema:os
         http://docs.oasis-open.org/xacml/\
         access_control-xacml-1.0-context-schema-os.xsd">
  <Subject>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
               DataType="&xs;string">
      <AttributeValue>Analyst1</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
          DataType="&xs;string">
      <AttributeValue>level:1</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
               DataType="&xs;string">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Figure 5.3.3: XACML Request to authorise user.

escalating the event, so that for instance the CERT team or law enforcement investigates the event. An alternative solution is to use key shares, so that the user may have one share, but needs to ask for permission, for example from the data controller, to get the necessary additional shares to disclose events that are outside her own shift. This reduces the privacy impact of day to day security analysis to be more according to a needs basis than the current practice, which typically means no restrictions on access to potentially sensitive data in the alarm database. It also opens up for more transparency in the form of logging how, when and by whom access to sensitive information is being done.

5.3.3 XML Element Authorisations

Authorisation requests for individual XML elements proceeds in a similar way as proposed for the Decision-cache [135]. The XACML Response to XML element authorisation is however slightly different, since it in addition contains an *<AttributeAssignment>* in the *element-restrictions* Obligation that allows access to one or more security levels, as shown in Fig. 5.3.5. In Fig. 5.3.5, only stakeholders authorised to security levels *level:1* and *level:2*

```
<Response>
  <Result ResourceID="level:1">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <!-- One or more public key and ephemeral key definitions -->
      <Obligation ObligationId="&prile;encryption-key:ephemeral:2"
        FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;encryption-key:ephemeral:2:algorithm"
          DataType="&xs;string">http://www.w3.org/2001/04/xmlenc#aes128-cbc
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;encryption-key:ephemeral:2:timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1h
        </AttributeAssignment>
      </Obligation>
      <Obligation ObligationId="&prile;public-key:2:authorise" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;public-key:2:pem-key"
          DataType="&xs;base64Binary">-----BEGIN PUBLIC KEY-----
        .......
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:2:ephemeral-reference"
          DataType="http://www.w3.org/2001/XMLSchema#string">ephemeral:2
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:2:algorithm"
          DataType="&xs;anyURI">http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:2:user"
          DataType="&xs;String">Analyst1
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:2:levels"
          DataType="&xs;String">level:1</AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;public-key:2:timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1D
        </AttributeAssignment>

      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 5.3.4: XACML Response to user authorisation request.

```
<Response>
  <Result ResourceID="&prile;resource:1:id">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="&prile;element-restrictions"
                  FulfillOn="Permit">

        <AttributeAssignment AttributeId="&prile;resource:10:policy:anonymise"
          DataType="&xs;string"></AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:10:authorisation:levels"
          DataType="&xs;String">level:1,level:2</AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:1:cache-timeout"
          DataType="http://www.w3.org/TR/2002/WD-xquery-operators-20020816#\
          dayTimeDuration">P1D
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 5.3.5: XACML Response to XML Element Authorisation.

can reverse the anonymisation of this element.

5.3.4 Reversible Anonymisation Protocol

The plaintext XML representation of the authorisation map *AM* and confidential information *L* is shown in Fig. 5.3.6. XML Signatures and all confidential information to be encrypted are stored in an IDMEF <*AdditionalData*> element with *meaning "EncryptedData"* and *type "xml"*. This element contains a set of one or more <*AuthorisationMap*> elements, where each element corresponds to AM_u for user *u*. Each <*AuthorisationMap*> element contains one or more <*KeyInfo*> elements that describe the key mapping KM_i for security level *i*. <*KeyInfo*> furthermore contains <*KeyName*>, which is a nonce referring to the sensitive information l_i, and <*KeyValue*> that stores the encryption key K_i for security level i. PK_u is not stored in the XML representation of *AM*, since the public key only is used by the anonymiser.

The confidential information in *L* is subsequently stored in <*SecurityLevel*> elements, where each <*SecurityLevel*> element corresponding to l_i has an *ID* attribute referring to K_i. The confidential information is stored in random order, using random identifiers to avoid revealing explicitly which security level (or grading) the confidential information has. Each security level in addition contains a nonce carried in a <*KA-Nonce*> element, to make it harder to correlate sensitive information between IDS alarms.

Then follows the <*DeAnonymisers*> element, which encloses the confidential information for the given security level. The <*DeAnonymisers*> element has a *level* attribute that refers to the given security level in cleartext, and contains a a set of one or more <*DeAnonymise*> elements that specify how to reverse the anonymisation. The <*Deanonymise*> element has a *resource* attribute that refers to the given XPath expression r_p, and furthermore contains a set of zero or more <*AnonymisedData*> elements referring to the confidential information $e_{p,q}$.

The data structure used to store the <*DeAnonymisers*> is shown in line 27 of Fig. 5.3.8. It consists of a set of tuples (r_p, Q_i), one for each security level i. The row vector Q_i contains a replacement list of authorised element values that r_p matches for the given security level i. This ensures that the same number of elements that originally was anonymised, will be replaced in the same position when the anonymisation is reversed.

For example, assume that the resource expression r_2 matches three elements $\{e_{2,1}, e_{2,2}, e_{2,3}\}$. The first and third XPath matches $e_{2,1}$ and $e_{2,3}$ are authorised for level 1 whereas $e_{2,2}$ is authorised for level 2. In this case $Q_1 = (e_{2,1}, \emptyset, e_{2,3})$ and $Q_2 = (\emptyset, e_{2,2}, \emptyset)$. This ensures that three XPath matches exist for both security level 1 and 2. The function reversing the anonymisation will ignore elements marked as empty, so that only the anonymisation of $e_{2,1}$and $e_{2,3}$ will be reversed for level 1 and $e_{2,2}$ for level 2.

Then, the data is encrypted using XML Encryption [44], as illustrated in Fig. 5.3.7. All encrypted data is stored in the IDMEF <*AdditionalData*> element with meaning *"EncryptedData"*, and type *"xml"* to comply with the IDMEF extension scheme. All ciphertexts are Base64 encoded, as required by the XML Encryption and Signature standards [44, 45]. The encryption scheme uses EK_u to encrypt the <*AuthorisationMap*> per user u, and encrypts EK_u with the user's public key PK_u. The encrypted key $EK_u^{'}$ is then wrapped into the <*xenc:EncryptedData*> inside the <*ds:KeyInfo*> element. This approach uses standard key wrapping in XML Encryption, and avoids encrypting the larger <*AuthorisationMap*> data sequences using RSA encryption, which would be slower than using a block cipher like AES.

All confidential information l_i inside each <*SecurityLevel*> element is encrypted using XML Encryption with K_i as key. The *ID* attribute of the <*Se-*

```
<AdditionalData xmlns="http://www.prile.org/anonymiser"
                meaning="EncryptedData" type="xml">
  <xml>
    <AuthorisationMap user="User1">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <KeyName>qgy3r1hS</KeyName> <!-- Ref. to level:1 -->
        <KeyValue>eiZ5/W/9K/1TW/1P///9</KeyValue>
      </KeyInfo>
      ... <!-- Possibly references to more security levels. -->
    </AuthorisationMap>
    ... <!-- Possibly references to more users. -->
    <SecurityLevel ID="qgy3r1hS">
      <KA-Nonce>W3YjB/W2</KA-Nonce>
      <DeAnonymisers level="level:1" xmlns="http://www.prile.org/anonymiser">
        <DeAnonymise resource="/Alert/AdditionalData[@meaning='payload']">
          <AnonymisedData>Payload data...</AnonymisedData>
        </DeAnonymise>
        <DeAnonymise resource="/Alert/Source/Node/*">
          <AnonymisedData>10.0.2.2</AnonymisedData>
          ... <!-- Possibly more elements matching XPath expression. -->
        </DeAnonymise>
        ... <!-- Possibly more DeAnonymise clauses. -->
      </DeAnonymisers>
    </SecurityLevel>
    ... <!-- Possibly more security levels. -->
  </xml>
</AdditionalData>
```

Figure 5.3.6: Plaintext XML representation of the authorisation map *AM* and confidential information *L*.

curityLevel> is used to identify the matching key K_i during decryption.

The anonymised XML message with the <*AdditionalData*> Encrypted-Data element is then sent to the receiving application.

5.3.5 Reversing the Anonymisation

To reverse the anonymisation, the anonymiser first checks the outer XML Signature of the encrypted message, to verify that it has not been tampered with. Then, the IDMEF message containing the IDS alarm will be parsed by an XML parser. The process of reversing the anonymisation starts by scanning the wrapped keys in the <*ds:KeyInfo*> elements of <*xenc:EncryptedData*> for a <*ds:KeyName*> element that matches the current user u. Any matching elements will be attempted decrypted using the user's secret key SK_u, and if successful, then EK_u can be used to decrypt the corresponding <*xenc:CipherValue*> to reveal the authorisation map AM_u.

The deanonymiser will then iterate through the <*KeyInfo*> elements of the decrypted <*AuthorisationMap*>, and attempt to locate a <*SecurityLevel*> element with an ID attribute that matches the content of the <*KeyName*> ele-

```
<AdditionalData  meaning="EncryptedData" type="xml">
  <xml>
    <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                        Type="&xenc;Element">
      <xenc:EncryptionMethod Algorithm="&xenc;aes128-cbc"/>
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
          <xenc:EncryptionMethod Algorithm="&xenc;rsa-oaep-mgf1p"/>
          <xenc:CipherData>
            <!-- Session key (AES) encrypted using public key (RSA). -->
            <xenc:CipherValue>gao0DeZu4Pdat3....</xenc:CipherValue>
          </xenc:CipherData>
        </xenc:EncryptedKey>
        <ds:KeyName>User1</ds:KeyName>
      </ds:KeyInfo>
      <xenc:CipherData>
        <!-- Encrypted AuthorisatiomMap for User1. -->
        <xenc:CipherValue>FtSxJHy+oLMACpmJJhGSl6DlN3...</xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData>
    ... <!-- Possibly reference to more users. -->
    <SecurityLevel ID="qgy3r1hS">
      <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
                          Type="&xenc;Content">
        <xenc:EncryptionMethod Algorithm="&xenc;aes128-cbc"/>
        <xenc:CipherData>
          <xenc:CipherValue>iSpiz64e310z30e/p0u2...</xenc:CipherValue>
        </xenc:CipherData>
      </xenc:EncryptedData>
    </SecurityLevel>
    ... <!-- Possibly more security levels. -->
  </xml>
</AdditionalData>
```

Figure 5.3.7: XML Encrypted representation of the authorisation map *AM* and confidential information *L* stored in IDMEF AdditionalData element.

ment. If a match is found, then the content of the SecurityLevel is decrypted using the key K_i stored in the corresponding <*keyValue*> element.

After that, the deanonymiser goes through each <*DeAnonymisers*> element, and performs an XPath search according to the *resource* attribute of the <*DeAnonymise*> element, then loops through all matching elements of the XPath expression, and replaces the anonymised text with the corresponding text in the list of <*AnonymisedData*> elements.

If all security levels have been deanonymised, then the inner XML signature can be checked, to verify that anonymisation reversal was successful. We use this as a regression test to verify correct operation of the anonymiser and deanonymiser.

The anonymisation of an IDMEF alarm with reversible anonymisation can in other words only be reversed by stakeholders who have one of the secret keys or necessary key shares used to decrypt $AM_u^{'}$. It is furthermore only possible to decrypt sensitive information to the level the secret key is authorised for.

5.4 Supporting Default DENY Protocol

The reversible anonymisation protocol described so far is a default PERMIT protocol. This means that any information in the IDS alarms, which is not explicitly being authorised by the cache specification, by default is being permitted. This strategy has the deficiency that parameters which are unknown by the policy will not be anonymised. This can be problematic from a privacy perspective. For example for IDMEF <*AdditionalData*> elements and attributes that are not standardised by RFC 4765 [62]. Different IDS vendors may for example decide to name the payload or other $< AdditionalData >$ elements or attributes slightly differently or may include new, possibly privacy violating fields in the <*AdditionalData*> extension field, which may cause significant privacy leakages.

A better strategy is then to support *privacy by default* [25], by introducing a default DENY protocol. This is also similar to common practices in computer security for firewall design, which typically use a default DENY scheme. The remainder of this section outlines how the building blocks for

```
Initial XACML authorisation:
$R = \{r_1, r_2, ..., r_N\}$
$KM = \{(K_i, i) | K_i \leftarrow rnd(), i \in 1, ..., d\}$
$AM = \{(PK_u, EK_u, \Lambda_u, u) | \Lambda_u \subseteq KM, EK_u \leftarrow rnd(), u \in 1, ..., m\}$

function ANONYMISE($message, defaultpolicy, defaultlevel$)
    $\underline{L} = (l_1 = \emptyset, ..., l_d = \emptyset)$  ▷ Confidential information per security level.
    $\underline{L}' = (l_1' = \emptyset, ..., l_d' = \emptyset)$  ▷ Encrypted confidential information.
    $C = \emptyset$  ▷ Set of classified elements to be anonymised.
    $D = \emptyset$  ▷ Set of elements to declassify.
    for $p \leftarrow 1, ..., N$ do
        $\underline{e_p} \leftarrow XPath(r_p, message)$  ▷ $\underline{e_p} = \{e_{p,1}, ..., e_{p,z}\}$ XPath matches
        Evaluate XPATH expressions to select scope parameters for $e_p$
        Create $d \times z$ matrix $Q$, initialised to $\emptyset$ for authorised elements.
        for $q \leftarrow 1, ..., z$ do
            Authorise element $e_{p,q}$ (XACML request)
            $(decision, anonymise, declassify, levels) \leftarrow XACMLReq(scope(e_{p,q}))$
            if $decision =$ "$Permit$" then
                if $anonymise = True$ then
                    for $i \in levels$ do  ▷ iterate through security levels.
                        $Q_{i,q} \leftarrow valueof(e_{p,q})$  ▷ Copy value of element.
                    $C \leftarrow C \cup e_{p,q}$  ▷ Anonymise element
                else if $declassify = True$ then
                    $D \leftarrow D \cup e_{p,q}$  ▷ Declassify element
                else
                    $C \leftarrow C \cup e_{p,q}$  ▷ Anonymise element
            else
                $C \leftarrow C \cup e_{p,q}$  ▷ Anonymise element
        for $i \in 1, ..., d$ do
            $l_i \leftarrow l_i \cup \{(r_p, Q_i)\}$
    if defaultpolicy=PERMIT then
        for $element \in C \backslash D$ do
            Anonymise $element$
```

Figure 5.3.8: Reversible anonymisation(part 1 of 2).

```
    else if defaultpolicy=DENY then
        Fetch all elements containing text and all attributes.
        $R_{defaultlevel} = \{$
        $"//*[name()]/*[normalize\text{-}space(text())]",$
        $"//@*"\}$
        for $r_{defaultlevel} \in R_{defaultlevel}$ do
            $allElements \leftarrow XPath(r_{defaultlevel}, message)$
            $Q_{defaultlevel} = []$
            for $element \in allElements$ do
                $Q_{defaultlevel}.append(valueof(element))$
                if $element \notin D$ or $element \in C$ then
                    Anonymise $element$
            $l_{defaultlevel} \leftarrow l_{defaultlevel} \cup \{(r_{defaultlevel}, Q_{defaultlevel})\}$
    $AM' \leftarrow \emptyset$
    for $(PK_u, \Lambda_u) \in AM$ do
        $AM' \leftarrow AM' \cup \{(Enc(PK_u, EK_u), Enc(EK_u, \Lambda_u), u)\}$    ▷ Encrypt AM
    for $i \in 1, ..., d$ do
        $l_i' \leftarrow Enc(K_i, l_i)$    ▷ Encrypt each security level
    return $(AM', \underline{L}')$
end function
```

Figure 5.3.9: Reversible anonymisation (part 2 of 2).

```
function DEANON($user, SK_{user}, msg', AM', \underline{L}'$)
    for $(EK_u', \Lambda_u', u) \in AM'$ do
        if $u = user$ and $EK_u \leftarrow Dec(SK_{user}, EK_u')$ then
            if $\Lambda_u \leftarrow Dec(EK_u, \Lambda_u')$ then
                for $(K_i, i) \in \Lambda_u$ do
                    $l_i \leftarrow Dec(K_i, l_i')$
                    for $(r_p, Q_i) \in l_i$ do
                        $\underline{e_p} \leftarrow XPath(r_p, msg')$
                        for $q \leftarrow 1, ..., z$ do
                            if $Q_{i,q} \neq \emptyset$ then
                                Restore content:
                                $e_{p,q} \leftarrow Q_{i,q}$
    return $msg'$
end function
```

Figure 5.3.10: Anonymisation reversal.

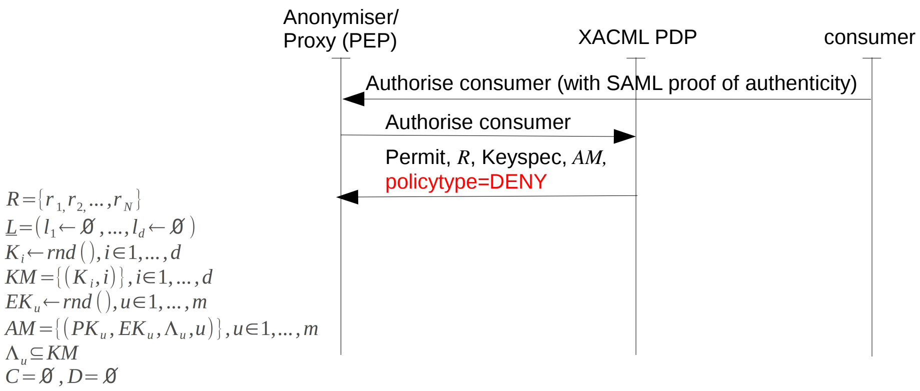

Anonymiser/
Proxy (PEP)
XACML PDP
consumer
Authorise consumer (with SAML proof of authenticity)
Authorise consumer
Permit, R, Keyspec, AM,
policytype=DENY
$R=\{r_{1,} r_{2,} \ldots, r_N\}$
$\underline{L}=(l_1 \leftarrow \emptyset, \ldots, l_d \leftarrow \emptyset)$
$K_i \leftarrow rnd(), i \in 1, \ldots, d$
$KM=\{(K_i, i)\}, i \in 1, \ldots, d$
$EK_u \leftarrow rnd(), u \in 1, \ldots, m$
$AM=\{(PK_u, EK_u, \Lambda_u, u)\}, u \in 1, \ldots, m$
$\Lambda_u \subseteq KM$
$C=\emptyset, D=\emptyset$

Figure 5.4.1: Initial authorisation for default DENY with assignment of default security level.

multi-level privacy/security can be used to implement a default DENY reversible anonymisation protocol for the anonymiser.

A default DENY scheme can be implemented by minor modifications of the proposed scheme. First, the *defaultpolicy* Obligation in the initial XACML authorisation contains an *<AttributeAssignment>* element with ID *&prile;policytype:DENY* instead of *PERMIT*, to specify that this is a default DENY protocol, as shown in Fig. 5.4.1. The value of this *<AttributeAssignment>* is the optional security level *i* (e.g. *level:1)* where the anonymised information for the default DENY policy is stored. If no security level is specified, then information for the default DENY scheme will be stored in a security level named *default*. Subsequent authorisation of other parties after the initial authorisation is done in the same way as for the default PERMIT scheme shown earlier.

Authorisation of individual elements can then be performed in two ways, depending on the outcome of the XACML element authorisation:

1. If the outcome of the XACML Response is PERMIT with an Obligation to *anonymise* information, then the *levels* specify that the information for this document element should remain anonymised, and moved from the default security level and to the security levels specified in *levels*. This is the same operation as shown in Fig. 5.2.4.

2. If the outcome is PERMIT with an Obligation to *declassify* informa-

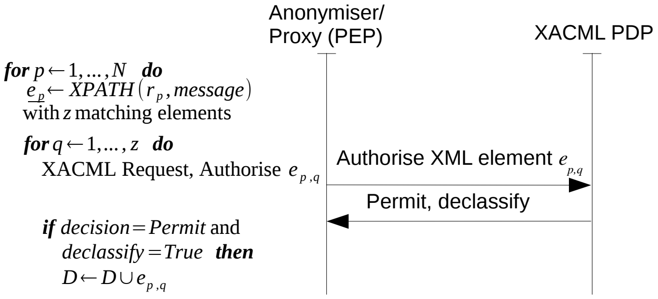

Anonymiser/
Proxy (PEP)
XACML PDP
for $p \leftarrow 1,\ldots,N$ do
$e_p \leftarrow XPATH(r_p, message)$
with z matching elements
for $q \leftarrow 1,\ldots,z$ do
XACML Request, Authorise $e_{p,q}$
Authorise XML element $e_{p,q}$
Permit, declassify
if $decision = Permit$ and
$declassify = True$ then
$D \leftarrow D \cup e_{p,q}$

Figure 5.4.2: Element declassify operation for default DENY scheme.

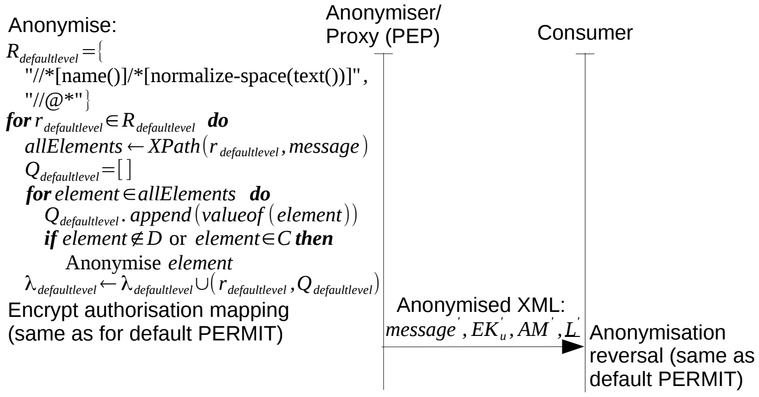
tion, then this means that the given element should be declassified, i.e. it should not be anonymised in the original IDMEF message. This operation is shown in Fig. 5.4.2.

3. If the outcome is DENY, or PERMIT with unknown or undefined parameters and the default policy is DENY, then nothing should be done, since the default DENY policy protects the information of the XML message.

This approach allows the default DENY scheme to implement policies supporting anonymisation of information by moving certain information to a different security level. The scheme also supports declassification of information that should remain visible in the XML message. Elements that are not explicitly authorised remain in the default security level for the DENY policy.

Declassification of information is controlled by the XACML policy. This means that resource elements are authorised as normal, however the authorisation decision for XACML elements that are declassified contains an obligation to *declassify* the given information instead of *anonymising* it. This is implemented as an XACML *<AttributeAssignment>* with *AttributeID &prile;resource:p:policy:declassify,* where *p* is the anonymisation policy of the element being referenced.

The anonymisation enforcement part of the default DENY scheme is described in Fig. 5.4.3. All information which needs to be anonymised by default, denoted by $R_{defaultlevel}$, can be identified using two XPath resource expressions. The first expression $//*[name()]/*[normalize\text{-}space(text())]$ selects the text attribute of all XML elements trimmed for whitespace, and



```
Anonymise:
$R_{defaultlevel} = \{$
  "//*[name()]/*[normalize-space(text())]",
  "//@*"$\}$
for $r_{defaultlevel} \in R_{defaultlevel}$ do
  $allElements \leftarrow XPath(r_{defaultlevel}, message)$
  $Q_{defaultlevel} = [\,]$
  for $element \in allElements$ do
    $Q_{defaultlevel}.append(valueof(element))$
    if $element \notin D$ or $element \in C$ then
      Anonymise $element$
  $\lambda_{defaultlevel} \leftarrow \lambda_{defaultlevel} \cup (r_{defaultlevel}, Q_{defaultlevel})$
Encrypt authorisation mapping
(same as for default PERMIT)
```

Figure 5.4.3: Default DENY scheme.

the second expression $//@*$ selects the value of all XML attributes in the XML document being anonymised[5].

The algorithm then iterates through these resources and selects the matching elements using XPath. Then the enforcement part loops through all matching elements for each resource in $R_{defaultlevel}$ and adds the value of the elements to the list of elements in the default security level $Q_{defaultlevel}$. In addition, elements that either are explicitly classified or elements that are not declassified are anonymised. Subsequently the data required to reverse the default security level is stored in $l_{defaultlevel}$, by executing
$l_{defaultlevel} \leftarrow l_{defaultlevel} \cup (r_{defaultlevel}, Q_{defaultlevel})$.

The remainder of the default DENY anonymisation scheme, including anonymisation reversal, is equivalent to the default PERMIT scheme. The complete anonymisation algorithm that combines the default PERMIT and default DENY schemes is shown in Fig. 5.3.8.

Anonymisation policies, especially default DENY policies, need to consider type casting issues if information is passed via SOAP calls. It is recommended to use '0' as block marker character for anonymising information, since this works both for text and numeric data. Enumerated XML data types need customised anonymisation functions to select one of the enumerated values. This can be done using the *replace-with* anonymisation function described in [135].

[5] These two expressions have not been combined to one using the XPath or ("|") operator, to ensure that the sequence of matches is well defined.

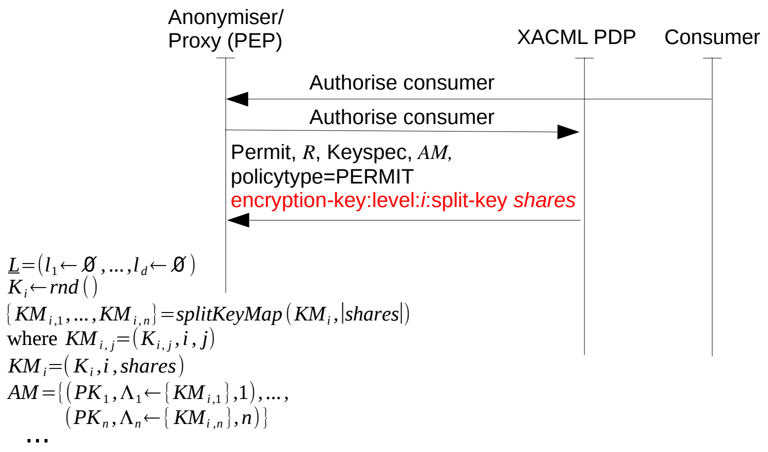

Anonymiser/
Proxy (PEP)
XACML PDP
Consumer
Authorise consumer
Authorise consumer
Permit, R, Keyspec, AM,
policytype=PERMIT
encryption-key:level:i:split-key *shares*
$\underline{L}=(l_1 \leftarrow \emptyset, \ldots, l_d \leftarrow \emptyset)$
$K_i \leftarrow rnd()$
$\{KM_{i,1}, \ldots, KM_{i,n}\} = splitKeyMap(KM_i, |shares|)$
where $KM_{i,j}=(K_{i,j}, i, j)$
$KM_i=(K_i, i, shares)$
$AM=\{(PK_1, \Lambda_1 \leftarrow \{KM_{i,1}\}, 1), \ldots,$
$(PK_n, \Lambda_n \leftarrow \{KM_{i,n}\}, n)\}$
...

Figure 5.5.1: Key sharing scheme.

5.5 Adaptations Required to Support Key Sharing

Key sharing is implemented based on a threshold encryption scheme. The easiest key sharing scheme to adapt, is the scheme of Karnin, Greene and Hellman [43], assuming that all n shares must be known to reveal the secret (i.e. $t = n$), and assuming that the PEP acts as a trusted dealer.

Assume that the secret key space is all numbers from 0 to $2^{keysize}$ where *keysize* is the size of the encryption key in bits. Key sharing can then be implemented by letting the anonymiser choose $n-1$ random shares of the same size as the original encryption key, and calculate the last share as the chosen encryption key minus the sum of the chosen random shares modulo $2^{keysize}$. The encryption key can then be reconstructed by adding up all the shares modulo $2^{keysize}$.

To support secret key sharing, the encryption key definition in the XACML policy needs a *split-key* operator and the implementation must be extended to support addressing of key shares. This can be implemented by adding another <*AttributeAssignment*> with AttributeID *&prile;encryption-key:level:p:split-key* to the key definition Obligations of the initial XACML Response, where the content is a list denoted *shares* containing share identities.

In addition, a naming scheme for shares is introduced, to be able to refer to the key shares. The proposed approach is to extend the existing naming scheme for security levels with a key share part, i.e. *level:i:part:j* for example *level:1:part:1* for the first share of the level 1 encryption key. A key sharing scheme can then be set up by authorising stakeholders to encryption key shares instead of encryption keys, as illustrated in Fig. 5.5.1.

On receiving the initial XACML authorisation response, encryption keys K_i are generated as normal, and these are used to encrypt the sensitive information in $(l_1,...,l_d)$. The XACML Response also contains an Obligation to split the encryption key K_i into $n = |shares|$ subkeymaps $\{KM_{i,1},...,KM_{i,n}\}$. Each subkeymap $KM_{i,j}$ consists of a tuple $KM_{i,j} = (K_{i,j}, i, j)$, where $K_{i,j}$ is share number j of the encryption key for security level i and n is the number of key shares. The key map KM_i is also extended to contain a reference to all *shares*, i.e: $KM_i = (K_i, i, shares)$. The authorisation mapping AM will then contain one key mapping share for each public key, i.e:

$$\begin{aligned} AM &= \{(PK_1, \Lambda_1 \leftarrow \{KM_{i,1}\}, 1), ..., \\ &\quad (PK_n, \Lambda_n \leftarrow \{KM_{i,n}\}, n)\} \end{aligned} \tag{5.5.1}$$

so that the owners of $SK_1,...,SK_n$ need to collaborate to reveal the confidential information. Note that if the parent encryption key times out and is regenerated, then the shares must also be updated. In addition, the <*AuthorisationMap*> data structure needs to add a <*Shares*> element containing a comma separated list of the shares that are needed to reconstruct the encryption key. The deanonymiser only needs minor adaptations to request all required key shares and calculate K_i before decryption can commence.

5.6 Adaptations Required for Time-based Data Expiry

The fifth Privacy by Design principle requires end-to-end security with full lifecycle management of private or confidential data from inception and until destruction [25]. This can be implemented by introducing time-based data expiry, assuming that the messages can be protected using encryption until they reach the anonymiser. Time-based data expiry means that the encryption key expires after a given retention time, so that confidential information in

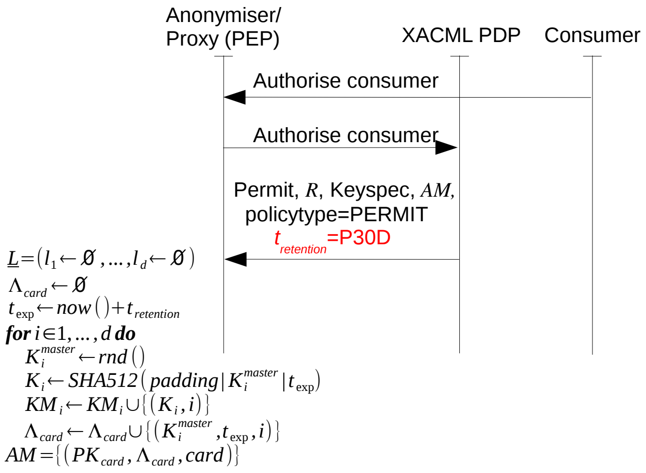

Anonymiser/
Proxy (PEP)
XACML PDP
Consumer
Authorise consumer
Authorise consumer
Permit, R, Keyspec, AM,
policytype=PERMIT
$t_{retention}$=P30D
$\underline{L}=(l_1 \leftarrow \emptyset, \ldots, l_d \leftarrow \emptyset)$
$\Lambda_{card} \leftarrow \emptyset$
$t_{\exp} \leftarrow now() + t_{retention}$
$\textbf{for } i \in 1, \ldots, d \textbf{ do}$
$K_i^{master} \leftarrow rnd()$
$K_i \leftarrow SHA512(padding | K_i^{master} | t_{\exp})$
$KM_i \leftarrow KM_i \cup \{(K_i, i)\}$
$\Lambda_{card} \leftarrow \Lambda_{card} \cup \{(K_i^{master}, t_{\exp}, i)\}$
$AM = \{(PK_{card}, \Lambda_{card}, card)\}$

Figure 5.6.1: Time-based data expiry initialisation.

the XML messages can not be accessed beyond this time. This ensures safe destruction of confidential data in the messages. Time-based data expiry can also be used to limit how long users will have access to the data they have analysed, for example to set up policies to avoid access to confidential data beyond the current work shift.

5.6.1 Implementing Time-based Data Expiry

The reversible anonymiser can with small adaptations support a time-based data expiry scheme similar to [74]. This scheme uses Smartcards to enforce the key expiration scheme (one card per user). Only the necessary adaptations will be discussed here.

The encryption key management of the reversible anonymiser needs to be adapted as shown in Fig. 5.6.1 to support the key derivation scheme in [74]. The scheme uses a key derivation function, with SHA-512 as hash function [1]. To support time-based data expiry, the XACML policy must return an obligation with the retention time $t_{retention}$ for element authorisation requests, so that the data retention time can be configured per encryption key. The retention time is then sent to the PEP in the initial XACML Response as part of the XACML Obligation. It will be defined in an *<Attribute-Assignment>* with AttributeID *&prile;encryption-key:level:i:retention-time,*

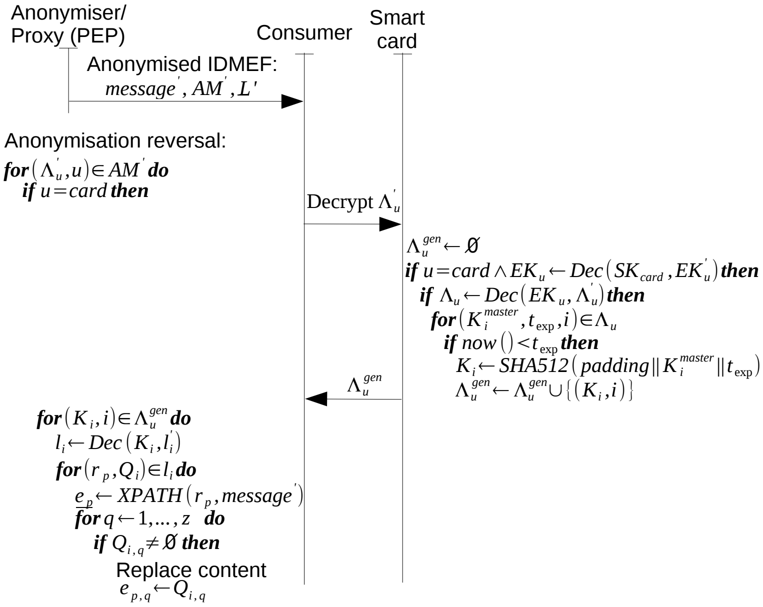

Anonymiser/ Proxy (PEP)
Consumer
Smart card
Anonymised IDMEF: $message', AM', L'$
Anonymisation reversal:
$\textbf{for}(\Lambda_u', u) \in AM' \textbf{ do}$
$\textbf{if } u = card \textbf{ then}$
Decrypt Λ_u'
$\Lambda_u^{gen} \leftarrow \emptyset$
$\textbf{if } u = card \wedge EK_u \leftarrow Dec(SK_{card}, EK_u') \textbf{ then}$
$\textbf{if } \Lambda_u \leftarrow Dec(EK_u, \Lambda_u') \textbf{ then}$
$\textbf{for}(K_i^{master}, t_{\exp}, i) \in \Lambda_u$
$\textbf{if } now() < t_{\exp} \textbf{ then}$
$K_i \leftarrow SHA512(padding \| K_i^{master} \| t_{\exp})$
$\Lambda_u^{gen} \leftarrow \Lambda_u^{gen} \cup \{(K_i, i)\}$
Λ_u^{gen}
$\textbf{for}(K_i, i) \in \Lambda_u^{gen} \textbf{ do}$
$l_i \leftarrow Dec(K_i, l_i')$
$\textbf{for}(r_p, Q_i) \in l_i \textbf{ do}$
$e_p \leftarrow XPATH(r_p, message')$
$\textbf{for } q \leftarrow 1, \ldots, z \textbf{ do}$
$\textbf{if } Q_{i,q} \neq \emptyset \textbf{ then}$
Replace content
$e_{p,q} \leftarrow Q_{i,q}$

Figure 5.6.2: Decryption for time-based data expiry.

and value $t_{retention}$.The key expiry time t_{exp} is then calculated as the current time plus $t_{retention}$.

To achieve time-based data expiry, the key expiry time t_{exp} must be cryptographically bound to the encryption key K_i. Assume that *padding* is a 64 bit number, that is initialised to zero. The time expiring encryption key K_i, that encrypts the classified information l_i, is derived from a master key K_i^{master} by using the key derivation function: $K_i = SHA512(padding||K_i^{master}||t_{exp})$. This encryption key is then used to encrypt the confidential information in l_i. The authentication map is the same as the basic scheme uses, i.e: $AM = \{(PK_{card}, EK_{card}, \Lambda_{card}, card)\}$. Λ_{card} is however modified to contain the master key K_i^{master}, the expiry time t_{exp} and the security level i. i.e: $\Lambda_{card} = \{(K_i^{master}, t_{exp}, i) | i \in 1, ..., d\}$. The parameters K_i^{master} and t_{exp} are stored in a <*KeyValue*> element in the XML representation of *AM*, using the convention that K_i^{master} is stored in a <*MasterKey*> element, and t_{exp} is stored in a <*TimeOut*> element. No further changes are needed to the remaining parts of the anonymiser.

The decryption algorithm must be modified to ask the Smartcard to decrypt Λ_u' as shown in Fig. 5.6.2. The figure only shows Smartcard based key retrieval. Smartcard initialisation and authorisation will be similar to [74]. The Smartcard will first initialise the generated key mapping $\Lambda_u^{gen} \leftarrow \emptyset$. Subsequently, it verifies that the key belongs to the card and decrypts the encryption key using $EK_u = Dec(SK_{card}, EK_u^{'})$. If this succeeds, then the Smartcard will decrypt Λ_u using $\Lambda_u \leftarrow Dec(EK_u, \Lambda_u^{'})$, and then iterate through all tuples $((K_i^{master}, t_{exp}), i)$ in Λ_u and verify that the current time is less than t_{exp}. If this test succeeds, then the Smartcard will generate the decryption key for security level i by using the key derivation function $K_i \leftarrow SHA512(padding||K_i^{master}||t_{exp})$ and generate a new keymap $\Lambda_u^{gen} = \Lambda_u^{gen} \cup \{(k_i, i)\}$. The Smartcard then returns Λ_u^{gen} to the Anonymisation reversal function, which reverses the anonymisation for the given security level i in the same way as shown previously.

Note that this scheme assumes trusted time sources, which can be implemented in a similar way as proposed in [74].

5.7 Other Declarations

This section describes how self-references and XML namespaces can be declared. Self references makes it possible to refer to the current XML element or attribute being authorised. XML namespace support is for example useful for SOAP Web Service Definition Language (WSDL) interfaces that provide IDMEF-based IDS services.

5.7.1 Declaring Self References

The Anonymiser supports declaring self-references, by declaring an XACML Attribute with AttributeID *urn:prile:org:resource:i:assertion:self:scope* and empty value. This approach also works for scope variables of other types, for example *urn:prile:org:resource:i:assertion:self:lcorr-shannon-entropy* to send the length-corrected Shannon entropy of the current element being authorised in the XACML Request. This is more flexible than extracting the current element using a scope XPath expression, which only works as long as the XPath expression matches a single element.

Self-references to the current element or attribute being authorised are sent as a Resource attribute in the XACML Request with AttributeID *urn:prile:org:resource:i:assertion:self:value*.

5.7.2 Namespace Declarations

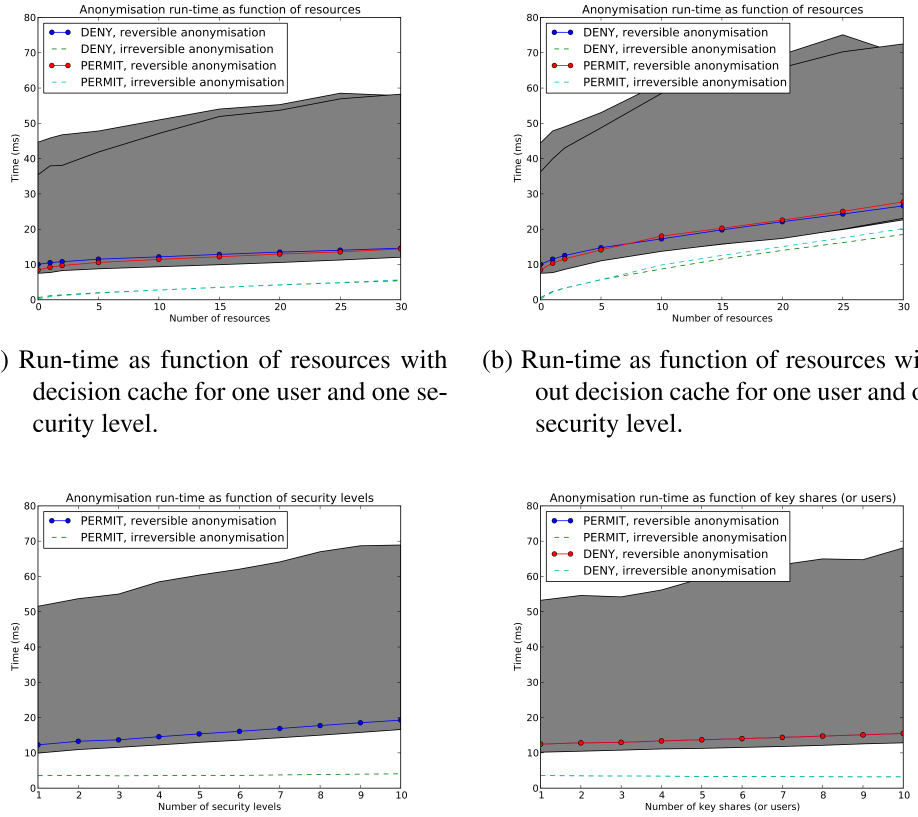
The Anonymiser supports XML namespace declarations for a namespace denoted as *nsx* by declaring an XACML Attribute in the *default-policy* obligation with *AttributeID urn:prile:org:xmlns:nsx* and value being the namespace URL, for example *http://iana.org/idmef*.

5.8 Experiments

The anonymiser and deanonymiser was tested on a server with 8 Gb RAM and a 3.3 GHz Intel Core i5 CPU. The anonymiser was connected to the deanonymiser using a SOAP web service with persistent HTTP connections, to verify the entire production pipeline. The performance can be expected to be somewhat higher if the anonymiser and deanonymiser are run separately. The experiments used the IDMEF alarm log from previous IDS experiments using PreludeIDS. These experiments are based on alarms from the 1999 KDD-Cup data set (DARPA IDS test set)[6]. This is an old synthetic data set, and will therefore have less diversity that one can expect from real traffic today. The cache hit rate and performance is therefore higher than what one can expect from a production system. The KDD-Cup data set was chosen despite these deficiencies, since this still is considered the gold standard for IDS measurements, and it is difficult to get access to real IDS data.

The software is implemented in Jython and based on the XACML Decision-cache based anonymiser in [135], which uses SunXACML with a Java HashMap based Least Recently Used cache and virtual token descriptor based XML parser VTD-XML for increased XPath performance. The solution in this chapter has been extended with the GeoXACML patches [6], to support more advanced XACML data types like pointlists which are used by the implementation. This also allows for supporting location-aware anonymisation and authorisation policies. Apache Santuario is used for XML encryption, Apache CXF as SOAP server for the deanonymiser and SUDS as SOAP client for

[6] KDD-Cup 1999 data (DARPA IDS test set) http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html



(a) Run-time as function of resources with decision cache for one user and one security level.

(b) Run-time as function of resources without decision cache for one user and one security level.

(c) Default PERMIT run-time for 15 resources, one user per security level and no key shares as function of security levels with decision cache.

(d) Default PERMIT run-time for 15 resources and one security level as function of key shares or users with decision cache.

Figure 5.8.1: Anonymiser run-time as a function of number of anonymised resources, default protocol type, number of security levels and number of key shares. The subfigures show the median and the 95% confidence band (in grey) of the measurements.

the anonymiser. The anonymiser uses three threads - one for reading and buffering IDS alarms, one for anonymising the alarms and an output thread for buffering and sending data to the deanonymiser. This strategy decouples the anonymiser from the deanonymiser to avoid any of the threads blocking the production pipeline.

Each statistical value is calculated as the average of an ensemble of 100 IDS alarms. Each experiment furthermore selects a random uniform sample of 1000 IDS alarms from a corpus of 130.000 IDS alarms from the KDD-Cup'99 test set. A maximum limit of 30 anonymisation rules, 10 security levels and 10 key shares was chosen, since this is expected to be around the maximum numbers needed for IDMEF anonymisation policies. A new XACML policy with a random selection of the current number of anonymisation rules was generated for each experiment in the ensemble. It uses an XACML policy generator to generate random anonymisation policies with between 0 and 30 anonymised resources, each resource consisting of two different policies matching the relevant Target section and containing a Condition section that matches one of two different policies per anonymised resource. The policy generator furthermore supports a configurable number of users, security levels and key shares.

5.8.1 Anonymiser Performance

Anonymiser run time as a function of anonymised resources, default protocol type, number of security levels and number of key shares is shown in Fig. 5.8.1. Each subfigure shows the median and the 95% confidence band (0.025-0.975 percentile) for reversible anonymisation, as well as the median for irreversible anonymisation indicated using stapled lines.

The Figures show that the distribution functions are significantly skewed towards lower run-times both for non-cached and cached results, which means that the median gives a more representative picture of the mode of the distribution than the standard deviation.

Figures 5.8.1a and 5.8.1b show that the anonymiser run-time for the default PERMIT scheme is nearly the same as the default DENY scheme as a function of number of resources in the interval between 0 and 30 resources. These experiments use one security level and no key shares. There is perhaps a trend that the default DENY scheme starts with somewhat higher run-time and scales slightly better than the default PERMIT scheme. The figures furthermore show that the decision cache reduces the average run-time from 33 ms to 18.5 ms (median from 27 to 14ms) for 30 anonymised or declassified resources when reversible anonymisation is used. This means that the reversible anonymiser performance is increased from approximately 30 to 54 IDS alarms/s for 30 anonymised resources by using decision caching.

The cache hit rate for the experiments is 98%, which is higher than one can expect in a production system due to lack of entropy in the KDD-Cup data set. The figures furthermore show that reversible anonymisation based on XML Encryption and XML Signatures adds an average cryptographic overhead of 11 ms (median 8ms) compared to using irreversible anonymisation for the given experiments. Irreversible anonymisation with decision caching would give a performance of around 130 anonymised IDS alarms/s on the given hardware.

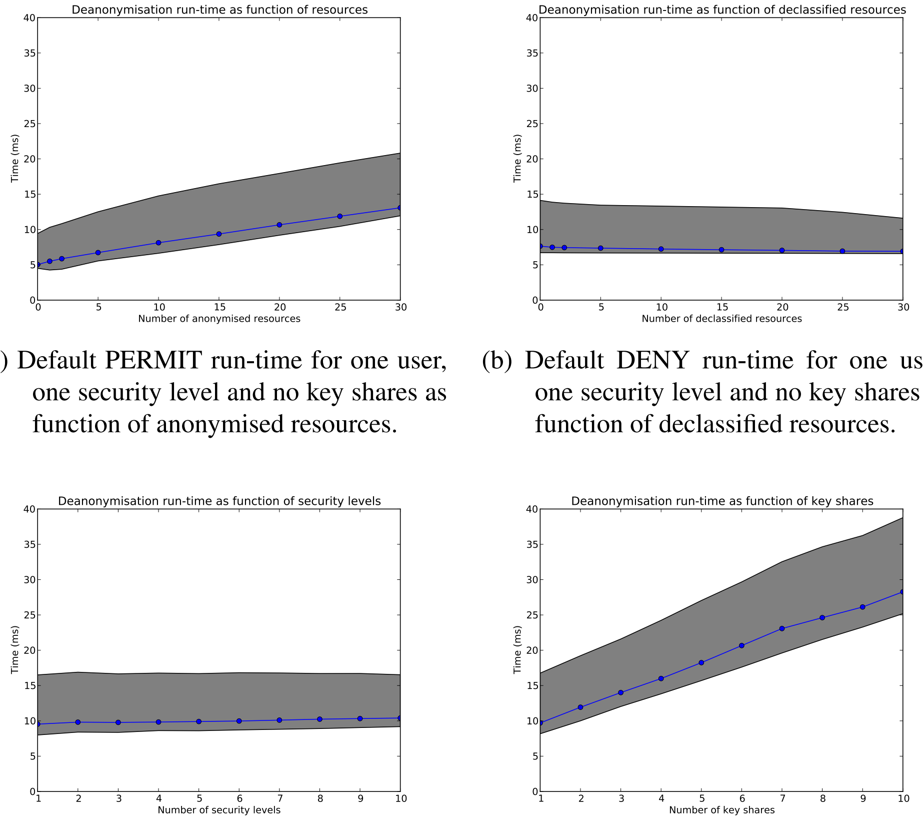
Fig. 5.8.1c shows the anonymiser run-time as a function of number of security levels for 15 resources, one user per security level and no key shares[7]. It shows that there is a linear dependency between run-time and number of security levels, where the run-time increases with 0.7ms per additional security level.

Fig. 5.8.1d shows the anonymiser run-time as a function of number of key shares or users for 15 resources and one security level. The logic for mapping access from a set users to a set of security levels or key shares is essentially the same for the anonymiser, hence using only one figure to show the performance as a function of of either users or shares. The Anonymiser run-time increases linearly and has little influence from number of shares, increasing only with 0.3ms per additional user or key share.

5.8.2 Deanonymiser Performance

Fig. 5.8.2a shows that the run time of the deanonymiser for the default PERMIT scheme increases linearly (0.27 ms/resource) after an initial transient part for the first 0-2 deanonymised resources, and is approximately 3 ms

[7] Note that this experiment cannot be performed for default DENY, since it by default uses one security level.



(a) Default PERMIT run-time for one user, one security level and no key shares as function of anonymised resources.

(b) Default DENY run-time for one user, one security level and no key shares as function of declassified resources.

(c) Default PERMIT run-time for 15 anonymised resources and no key sharing as function of security levels where one level is deanonymised by one user.

(d) Default PERMIT run-time for 15 anonymised resources and one security level as function of key shares (with same amount of users).

Figure 5.8.2: Deanonymiser run-time as a function of number of anonymised resources, default protocol type, number of security levels and number of key shares. The subfigures show the median and the 95% confidence band of the measurements.

faster than the anonymiser for anonymisation of 1-30 elements. The average value is 14 ms/alarm (median 13 ms/alarm) for 30 resources, which means that the deanonymiser manages to deanonymise 71 IDS alarms/s for 30 resources with the default PERMIT scheme in the experiments.

Fig. 5.8.2b shows that the deanonymiser run time of the default DENY scheme scales much better with number of deanonymised resources than the default PERMIT scheme. It has an average run-time of 8.8 ms (median 7.5 ms) and decreases slightly (by -0.02 ms/resource) with number of deanonymised resources. The reason for this is that all the work on deanonymising resources for the default DENY scheme is done in the anonymiser. Only one XPATH search is required to replace the content in the default security level, and less effort is required with more declassified resources, since these resources are not copied back from the default level. The default DENY scheme is in other words very efficient for the deanonymiser. The deanonymiser manages to deanonymise up to 113 alarms/s for the default DENY scheme, which can be an advantage, for example if the deanonymiser is used as part of an alarm correlation system.

Fig. 5.8.2c shows the deanonymiser run-time as function of number of security levels, assuming that only one of the security levels need to be deanonymised. The run-time only increases slightly (0.1 ms/security level) with increasing number of security levels.

Fig. 5.8.2d shows the deanonymiser run-time as a function of key shares. Adding key shares is relatively expensive, and adds 1.8 ms run-time per added share. The reason for this, is the relatively expensive RSA and ephemeral key decryptions that must be performed for each share.

The experiments indicate that both the anonymiser and deanonymiser should have sufficient performance to be usable at least for small to medium-scale deployments of privacy-enhanced IDS. The performance should also be sufficient for several other applications where the anonymiser and deanonymiser is used as part of a service oriented architecture, and where security or privacy is prioritised above performance. It can furthermore be noted that the XACML PDP, anonymiser and deanonymiser are parallelisable on an XML message level, meaning that the capacity can be scaled up by adding more hardware, if required.

5.8.3 Bandwidth efficiency of the proposed solution.

The original IDMEF message size is on average 3.8 kB. Each XML-signature user or key share adds approximately 1 kB of data to the message. There is furthermore a linear dependency where each additional anonymisation rule adds approximately 0.15 kB for the given test data. The anonymised message is 3.1 kB larger than the original message for 0 anonymised elements and 5.7 kB larger for 15 anonymised XML resources in the experiments with two signatures, one user, one security level and no key shares. For 15 resources and 10 users, each accessing an individual security level, the bandwidth usage increases by a factor of 11 to 41.6 kB per IDS alarm. This means that bandwidth usage will probably limit how complex anonymisation policies it is practical to implement with the proposed scheme. It is in particular limited how many security levels, users and key shares it is possible to implement without having too large bandwidth and performance overhead.

This means that it may not be desirable to operate with one key mapping per authorised user, since this solution scales poorly with number of authorised users. One way to mitigate this problem, at the expense of relying more on trust in the XACML authorisation, is to use a role-based authorisation scheme where roles are authorised using public keys for certain use scenarios instead of individual users. Such a scheme could for example be based on the Smartcard based encryption scheme proposed in Section 5.6, to securely deploy the secret role keys. It would in this case be natural to use the XACML Role-Based Access Control (RBAC) profile[8] for deploying role keys. Such a solution can be integrated with the proposed solution in a similar way as discussed in [136]. The details of this is however left as future work.

5.9 Discussion

Our approach has the advantage compared to existing schemes that pseudonymisation is not used, which eliminates the risk of traffic analysis attacks and known plaintext attacks on the pseudonyms. It is also an advantage that it is implemented as a proxy which allows for anonymising any XML protocol that can be sent via the proxy service. It is able to deal with IDS technolo-

[8] XACML RBAC profile: docs.oasis-open.org/xacml/3.0/xacml-3.0-rbac-v1-spec-cd-03-en.html

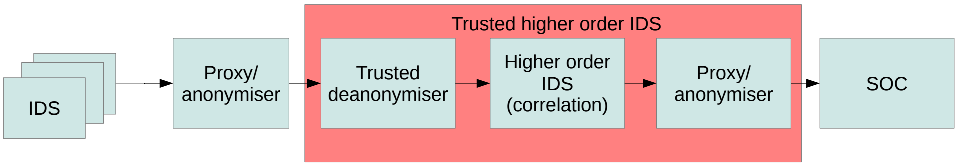
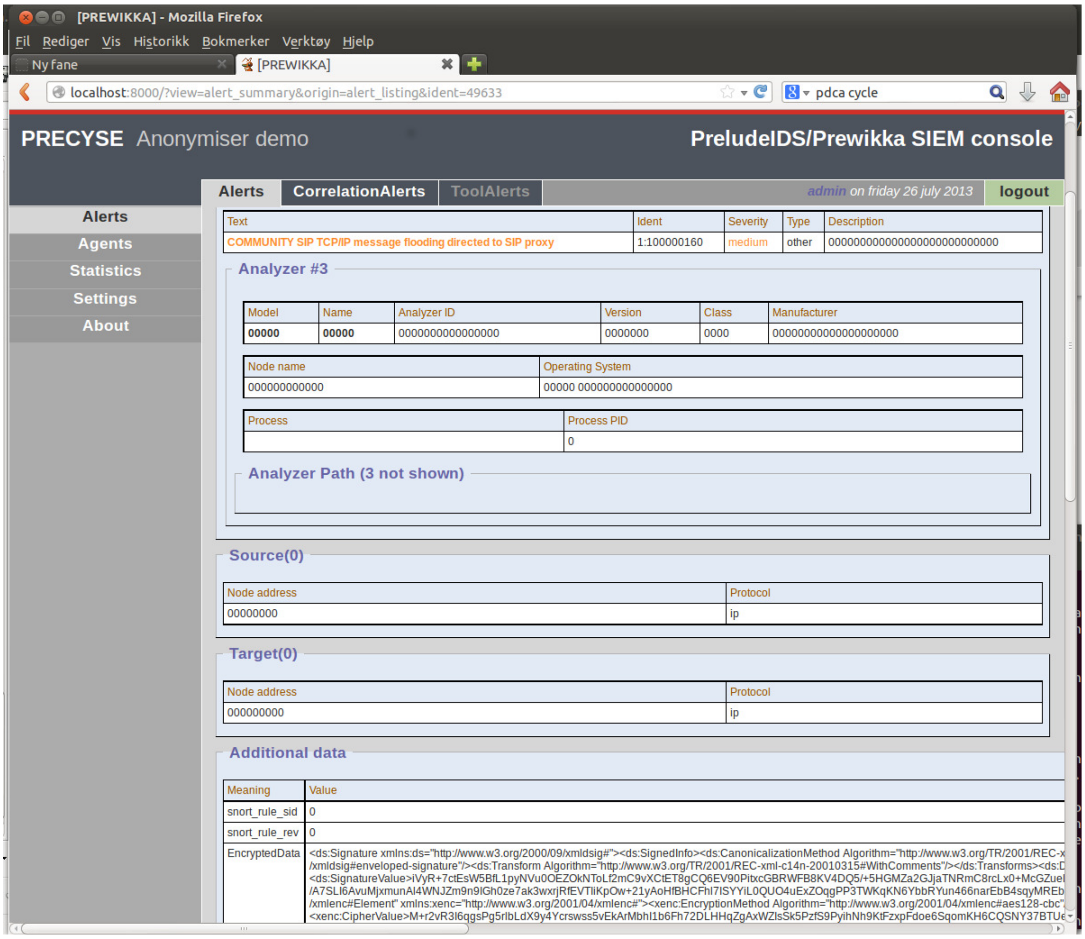


Figure 5.9.1: IDS alarm correlation analysis based on trusted application.


[PREWIKKA] - Mozilla Firefox
Fil Rediger Vis Historikk Bokmerker Verktøy Hjelp
Ny fane
[PREWIKKA]
localhost:8000/?view=alert_summary&origin=alert_listing&ident=49633
pdca cycle
PRECYSE Anonymiser demo
PreludeIDS/Prewikka SIEM console
Alerts
CorrelationAlerts
ToolAlerts
admin on friday 26 july 2013
logout
Alerts
Agents
Statistics
Settings
About

Text	Ident	Severity	Type	Description
COMMUNITY SIP TCP/IP message flooding directed to SIP proxy	1:100000160	medium	other	0000000000000000000000000

Analyzer #3

Model	Name	Analyzer ID	Version	Class	Manufacturer
00000	**00000**	0000000000000000	0000000	0000	00000000000000000000

Node name	Operating System
000000000000	00000 000000000000000

Process	Process PID
	0

Analyzer Path (3 not shown)

Source(0)

Node address	Protocol
00000000	ip

Target(0)

Node address	Protocol
000000000	ip

Additional data

Meaning	Value
snort_rule_sid	0
snort_rule_rev	0
EncryptedData	<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"><ds:SignedInfo><ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-x /xmldsig#enveloped-signature"/><ds:Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments"/></ds:Transforms><ds:D <ds:SignatureValue>iVyR+7ctEsW5BfL1pyNVu0OEZOkNToLf2mC9vXCtET8gCQ6EV90PitxcGBRWFB8KV4DQ5/+5HGMZa2GJjaTNRmC8rcLx0+McGZuel /A7SLI6AvuMjxmunAl4WNJZm9n9IGh0ze7ak3wxrjRfEVTliKpOw+21yAoHfBHCFhI7ISYYiL0QUO4uExZOqgPP3TWKqKN6YbbRYun466narEbB4sqyMREb /xmlenc#Element" xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"><xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc" <xenc:CipherValue>M+r2vR3l6qgsPg5rlbLdX9y4Ycrswss5vEkArMbhl1b6Fh72DLHHqZgAxWZlsSk5PzfS9PyihNh9KtFzxpFdoe6SqomKH6CQSNY37BTUe

Figure 5.9.2: Anonymised data in PreludeIDS.

gies that use IDMEF, which is a standardised XML-based alarm format. Fig. 5.9.2 shows that the Anonymiser is able to store anonymised IDS alarms in PreludeIDS, which uses a binary IDMEF-like data format internally[9]. The anonymiser here uses a default DENY protocol, and declassifies some elements amongst others alert text and rule identity[10]. This allows anonymised IDS alarms to be compatible with existing Security Information and Event Management systems that support IDMEF [2]. It must however be noted that anonymisation reversal may not be possible without altering the SIEM database, since this requires preserving the structure of the XML.

This problem can however be avoided by storing the anonymised IDS alarms directly into an XML database. This allows for easy access and deanonymisation of anonymised IDS alarms based on XQuery, without having to deanonymise all information. This works under the assumption that necessary information for accessing the IDS alarm, like a unique alarm identity, is available in cleartext. This approach should be sufficient for on-demand access to private or confidential information in the IDS alarms from the SOC. Another advantage is that this approach allows for implementing transparency on who have accessed sensitive information, by logging who is performing deanonymisation requests. The XML repository can be considered an example of a secure logging service that is implemented on top of the anonymisation framework.

It can also support hierarchical intrusion detection systems, for example in the form of trusted applications that are allowed to do alarm correlation based on sensitive information, as illustrated in Fig. 5.9.1. Alarms from higher-order IDSs can subsequently be reanonymised, if necessary, by adding another Anonymiser after the higher order IDS. Furthermore, such solutions can be placed within the organisational boundaries of an organisation which has outsourced managed security services, for example a health institution, so that sensitive information never leaves the organisational boundaries without either being anonymised or encrypted.

Transparency and accountability can be implemented by adding XACML obligations and necessary functionality in the IDS alarm database, to ensure

[9] Prelude-IDS: https://www.prelude-ids.org/

[10] Note that the anonymisation policy is set up to demonstrate the default DENY anonymisation scheme. Some more information may need to be revealed in a production environment, however this is a policy issue that needs to be agreed between the security manager and data controller.

that all access to anonymised information is logged. Events that in addition may be logged for increased transparency and accountability, are access to sensitive information in IDS alarms, manual classification of IDS alarms, investigations via a data forensic interface etc. This can be implemented using existing secure logging schemes, for example based on [74]. The details of this is however considered beyond the scope of the dissertation.

An advantage with the chosen approach for handling default DENY policies, is that no changes are needed for the declassifier, since the default security level is defined the same way as other security levels. The default security level should however be considered defined as the most restrictive (or highest) security level, unless classified information can be reliably identified using other means (e.g. if the XML document structure explicitly supports indicating classification level). Individual elements can if needed be moved to lower (or less restrictive) security levels by explicitly authorising these elements to a lower security level.

A potential weakness with the proposed scheme, is that it by default does not encrypt the digital signatures. It is however possible to encrypt the inner signature which verifies the original IDS alarm, by adding the resource of this signature to the caching/anonymisation policy, and marking it as sensitive. There is a circular dependency that does not permit the outer signature to be included in the encrypted text, since this signature is calculated over the encrypted text.

A disadvantage compared to the schemes proposed in [145, 51], is that our solution does not support any calculations (e.g. equality testing) on pseudonymised values. However, our objective is not to build in cryptographic capabilities into the anonymiser. IDS rules change frequently and may have very complex patterns for attack detection, so it is not clear how such schemes can be extended to support state of the art IDS-es today. It is for example not clear that such a scheme is possible to implement for commercial IDS rule sets where the rule definition is kept secret. It is therefore in our opinion better to support trusted applications that are authorised to do the necessary data mining instead of trying to keep track of when and under which conditions IDS alarms should be disclosed or not. Sensitive information in correlation alarms resulting from these trusted applications can subsequently be anonymised using our proposed approach.

It is important to mention that successful reversal of the anonymisation requires that the anonymisation function does not semantically change the structure of the XML document, since this causes the XPath expression for reversing the anonymisation to fail. One example of this, is if one or more text nodes are removed, since this alters the DOM tree. Furthermore, if XML data is used to represent the anonymised data, then these data must be quoted.

A more subtle limitation is that the default DENY protocol does not support replacing anonymised data by whitespace, since this causes the function *normalize-space()* to ignore the anonymised nodes. The latter problem can be worked around by defining an XPath extension function that recursively iterates through the DOM tree and identifies all text nodes enclosed by an element node. This has been implemented for the VTD-XML based parser. Note however that the problem persists if *normalize-space()* is used in other resource identifying expressions in the XACML policies. All in all, these are relatively minor limitations that can be detected and mitigated during regression tests of the XACML policies, since the inner XML signature calculated over the original message will fail if the anonymisation reversal is not done correctly.

Another limitation that was identified is that the sequence of matches for XPath resources containing several sub-expressions separated by the *or* operator "|" is undefined. This means that the resource XPath expressions must define two different resource expressions instead of using "|", to ensure that anonymisation can be reversed[11].

The XML encryption and signature part can probably be optimised by porting them to the VTD-XML parser. The code is also expected to be somewhat faster if ported to Java. This is however left as future work. The experiments illustrate that privacy-enhanced intrusion detection services based on reversible anonymisation is viable, at least for small to medium size IDS deployments.

[11] Note that this restriction only applies to Anonymise obligation functions, since the order of reversing anonymisation information must be defined. The Declassify function does not have this restriction.

5.10 Related Works

There are some examples of prior work that describes reversible anonymisation schemes that are not based on pseudonymisation. A reversible anonymisation scheme for anonymisation of DICOM images using automatically generated policies was proposed in [95]. The policy definition consists of a list of attribute rules that describe how the document shall be anonymised. Anonymised information is stored in a separate difference file, in order to later reverse the anonymisation by merging in this information. Our solution is more general and can anonymise any XML-based format using XACML-based policies, which is a standardised policy language. Our solution is furthermore different by embedding the information required to reverse the anonymisation in the messages, as well as supporting a multi-level security based scheme where different stakeholders can be granted access to information based on need. Our scheme furthermore supports both a default PERMIT, default DENY policy and key sharing, whereas this scheme only supports default PERMIT.

Another paper that suggests a reversible anonymisation scheme for protecting organisational data confidentiality in cloud-based services is [141]. Reversible anonymisation is however not yet implemented in this paper, so the performance measurements only show traditional irreversible anonymisation.

The chapter is also related to the field of privacy enhanced intrusion detection systems. Most previously proposed privacy-enhanced IDS schemes use some kind of pseudonymisation scheme, where sensitive information in the IDS alarms is replaced by pseudonyms, to later be able to reverse the pseudonymisation process on a needs basis.

Use of pseudonyms in audit logs was first suggested by Fischer-Hübner [119, 50]. Another early example of a privacy-enhanced IDS that uses pseudonymisation is the Adaptive Intrusion Detection system (AID) [124]. Both schemes use symmetric key encryption as pseudonym mapping, and focus mainly on encrypting subject identifying data. AID in addition contains a higher order IDS (expert system) that correlates the alarms, and discloses the pseudonymised data if suspicious sequences of events are detected. The pseudonyms of these early schemes are susceptible to traffic analysis attacks during the lifetime of the session key used for encrypting the pseudonym mapping.

An example of a privacy enhanced IDS that uses anomaly detection on pseudonymised data is [82]. The pseudonymisation strategy consists of a very simple static mapping between sensitive data and pseudonyms, however this paper also reflects over the need to pseudonymise other fields than directly user/subject identifying fields.

A Kerberos based pseudonymisation scheme is proposed in [104]. The scheme implements a hierarchical IDS solution where pseudonyms only are revealed if the higher-order anomaly-based IDS detects suspicious traffic. This scheme operates with group reference pseudonyms which correspond to UNIX user groups instead of pseudonymising users directly. The scheme uses public key encryption and relies on a trusted third party for initial pseudonym creation. An extended version of the protocol uses Mixes to avoid linkability towards the original data sessions. A similarity with our solution is that both are based on authorisation schemes (Kerberos and XACML), however our scheme is more flexible when it comes to policy-based fine-grained authorisation and anonymisation of XML data. A pseudonymisation scheme based on homomorphic encryption is proposed in [99]. This allows for performing certain equality or inequality tests on encrypted information without revealing the underlying information.

A privacy enhanced intrusion detection scheme for UNIX audit records is proposed in [51, 67]. This scheme is based on Shamir's threshold cryptography, and the general idea is that pseudonymised information shall only be disclosed when an attack scenario has been identified. An attack scenario here means that a sufficient amount of shares have been recovered from IDS alarms to recover the secret key used by the pseudonymiser. This approach proposes to use transaction pseudonyms to avoid linkability between pseudonyms, however the proposed implementation has some weaknesses that cause the authors to reintroduce linkability between transactions. This scenario has the same weakness as the other pseudonymisers, since it may be vulnerable to traffic analysis attacks in the intervals between rekeying of the pseudonymiser. Some of the ideas in this scheme have been extended to support multilaterally secure ubiquitous auditing in [146, 145]. It combines transaction pseudonyms based on threshold cryptography with secure multi-party computations to support secure and privacy enhanced tracking of mobile rescue units. The solution also supports verifiability via log attestation. This solution implements transaction pseudonyms in a semantically secure way, to mitigate the risk of traffic analysis attacks against the pseudonyms.

A secure logging scheme for retained data of an anonymity service (AN.ON) is described in [74]. This solution is based on smart cards in order to provide time restricted access to system logs from the anonymity service according to the requirements in the Data Retention Directive [47]. This scheme uses a similar hybrid encryption scheme to ours in that symmetric encryption is used for the log entries and asymmetric encryption is used for access to the keys. However our scheme is different by supporting reversible anonymisation with several security levels and not only encryption. This scheme is therefore complementary to the scheme proposed here.

Our solution is also somewhat related to anonymisation of network logs. A NetFlow anonymiser which supports multiple anonymisation strategies is proposed in [120]. The intrusion detection system BRO has support for anonymisation of packet traces [98]. However neither of these solutions support reversible anonymisation of XML messages.

5.11 Conclusion

This chapter proposes a reversible anonymisation scheme for protecting sensitive information in XML messages. The scheme has been applied to IDMEF-based intrusion detection system alarms, and we expect the reversible anonymisation protocol to be useful for policy based confidentiality and integrity protection of sensitive information for a range of services in a service oriented architecture.

The solution is based on existing standards like XML, IDMEF, XACML, XML-Encryption and XML-Signature, and uses a proxy-based reversible anonymiser based on an earlier proposed XACML decision cache for authorisation and anonymisation of XML documents [135]. The solution furthermore supports location-based anonymisation policies via the GeoXACML framework [6].

Using XACML gives flexibility when it comes to defining privacy or security policies for controlling access to sensitive information. It also solves deployment of encryption keys in an efficient way as part of the privacy policy. The scheme allows for defining parties that by default are authorised for access to sensitive information, but it can also support on-demand time-restricted access to sensitive data for authorised users.

A secret sharing scheme is supported, to enforce separation of duties constraints where more than one stakeholder need to agree to reveal the sensitive data. The scheme allows for policy-based control of rekeying intervals, data authorisation and anonymisation schemes. Furthermore, time-based data expiry is outlined, based on the scheme in [74], to support secure deletion of sensitive data after a configurable retention time.

This approach also provides a method to improve the privacy of certain types of big data implementations for problems that scale horizontally, assuming that a large number of smaller individual data sources can be anonymised before they are aggregated and stored in a big XML database.

The proposed approach has been integrated into the existing Security Information and Event Management systems (SIEM) PreludeIDS[12], which supports IDMEF. Anonymised IDS alarms can be stored in the SIEM database using the proposed approach without any modifications, since the alarms follow the standard IDMEF extension schema. However implementing support for anonymisation reversal may require some modifications of the SIEM tools, since the structure of the IDMEF XML needs to be maintained unmodified for successful anonymisation reversal. One way to mitigate this limitation, is to store the anonymised data in an XML database.

The performance of the proposed approach has been tested and should be sufficient for small to medium scale IDS deployments. However, larger data rates can be managed by running several anonymisers or deanonymisers in parallel. A useful feature for alarm correlation systems is that the deanonymiser is fast for default DENY policies, which allows for correlating alarms between several privacy-enhanced IDS sensors in business cases where this is acceptable from a privacy and confidentiality perspective.

[12] PreludeIDS: http://www.prelude-ids.org

5.12 Future Work

Implementing and testing time-based data expiry using the Smartcard-based solution is left for future work. More research is also needed on how to protect the XACML policies themselves, for example using XML encryption as proposed in [65]. Implementing support for role-based instead of user-based authorisation is also left as future work. An interesting idea is to extend the multi-level security based scheme proposed here to also cover XACML policies and policy handling. Details of logging procedures to ensure transparency of the operation is also left as future work. This can for example be implemented in a similar way as the AN.ON secure logging service [74]. It is also envisaged that the proposed scheme in the future can be extended to support operations on encrypted data, for example by using homomorphic encryption of the sensitive data elements as pseudonyms for the anonymised data. This could make the reversible anonymisation scheme more useful for XML databases, since it would allow defining certain standardised query operators (e.g. equality tests) on encrypted data, in a similar way as CryptDB does for relational databases [103]. Both anonymisation and deanonymisation are horizontally scalable, which make them suitable for performing data analysis and deanonymisation using tools like Apache Hadoop based clusters.

The next part of the dissertation investigates how privacy leakage in IDS alarms can be measured and subsequently enforced by defining suitable privacy policies for the privacy enforcement scheme described here.

Part III

Privacy Leakage Detection and Avoidance

Part III focuses on privacy leakage metrics and methods for privacy leakage detection and avoidance. Chapter 6 proposes a privacy leakage metric for detecting privacy leakages in IDS alarms based on quantitative information flow analysis founded in information theory. The metric is based on the standard deviation of Shannon entropy. Chapter 7 elaborates on how the privacy leakage metric can be used to support the privacy enforcement mechanism in Part II and proposes amongst others how the privacy metric fits into the well-known Plan Do Check Act method for improvement. This chapter also proposes how the privacy leakage metric can be used for more fine-grained detection of where information leakages are.

Chapter 6

Measuring Privacy Leakage for IDS Rules

This chapter proposes a measurement approach for estimating the privacy leakage from Intrusion Detection System (IDS) alarms. Quantitative information flow analysis is used to build a theoretical model of privacy leakage from IDS rules, based on information entropy. This theoretical model is subsequently verified empirically both based on simulations and in an experimental study. The analysis shows that the metric is able to distinguish between IDS rules that have no or low expected privacy leakage and IDS rules with a significant risk of leaking sensitive information, for example on user behaviour. The analysis is based on measurements of number of IDS alarms, data length and data entropy for relevant parts of IDS rules (for example payload). This is a promising approach that opens up for privacy benchmarking of Managed Security Service providers.

6.1 Introduction

The objective of this chapter is to develop an entropy-based metric that can be used for privacy leakage detection in intrusion detection system (IDS) alarms. The approach should be able to identify IDS rules that according to stakeholders' perception have a significant potential for leaking private or confidential information. It should also identify the worst IDS rules from a privacy or confidentiality perspective based on indicators that can be calculated automatically. For example IDS rules that:

- have a significant risk to leak information that is sensitive (privacy sensitive, security sensitive, business sensitive etc.);
- have an unclear or too simple definition of the attack detecting pattern, and therefore may trigger unnecessarily, in the worst case on person sensitive or confidential information.

Privacy policies can be used to define what information that is sensitive. Examples of sensitive information may be certain IP ranges of classified systems or sampled payload that may reveal private or confidential information. Information can also be defined as person sensitive by law, for example the sampled payload from a health institution which may contain person sensitive information. Another example is critical infrastructures that may contain security sensitive or confidential information in the data traffic about the processes being controlled. Last, but not least, payment databases handling financial transactions may reveal sensitive information like credit card numbers.

In these cases, the information is *per definition* sensitive, which means that *any* leakage of information that can be identified may be problematic. For such use cases, an objective information leakage metric will be sufficient to identify problematic leakage of private or confidential information.

In other cases, the privacy sensitivity will be subjective, and can only be evaluated in a representative way by the *owners* of the data being sampled - the users themselves. It may even in this case be possible for the data controller to get realistic estimates of the perceived privacy sensitivity by asking a representative random set of users, for example using a random poll on the service being used, about how they would value privacy leakages. However this approach will be expensive and does not scale well. It is therefore only viable for smaller evaluations of privacy impact.

It is therefore assumed possible for an authority like the data controller, that is overseeing the privacy interests, to estimate the privacy impact, denoted by $I \geq 0$, that an identified information leakage $L \geq 0$ causes. The privacy impact could for example be the subjective value or expected liability from privacy or confidentiality breaches, as proposed by [60]. The privacy leakage, denoted by π_R for a given IDS rule R can then be defined as the product of the information leakage metric L and the privacy impact I, i.e: $\pi_R = I \cdot L$. However, if investigation shows that the information leakage is caused by activities from attack vectors that do not cause any risk of revealing private, business sensitive or confidential information, then the privacy impact for a given IDS rule may be set low or even to zero. The combined metric π_R can be regarded as a privacy leakage risk metric, that can be used to measure and perform incremental improvements of the Managed Security Service (MSS) operation from a privacy perspective.

Current IDSs typically provide an all or nothing solution for handling private or confidential information in the alarms. The payload of the alarms is either being sent in cleartext or may be pseudonymised, for example by only sending references to where more information can be found in a data forensics system. There does not exist a more fine-grained management nor any measurements of sensitive information flows in such systems. It is in particular common that Open Source based IDS's like Snort, OSSEC or Prelude send payload in cleartext in the IDS alarms. Having a metric for how privacy invasive an MSS operation is will therefore be useful to benchmark the performance of different MSS providers from a privacy perspective. It will also be useful for tuning the IDS rulesets and for implementing anonymisation policies to reduce the privacy impact of the monitoring. Intuitively, such a privacy leakage model relates to the perceived *preciseness* of the IDS rule, i.e. how good it is at detecting only attack traffic without revealing non-attack traffic.

A promising candidate for a privacy leakage metric for IDS rules, is data entropy. This is a privacy leakage metric that is based on the *variability* of the underlying data. Examples of such metrics are Shannon-, Rényi or Min-entropy, which previously have been proposed as anonymity metrics [118, 29]. Entropy can also be used to measure *coding efficiency*, for example whether sampled payload excerpts most likely are encrypted or compressed [118]. This chapter investigates a model of privacy leakage from IDS rules that is based on the variation in entropy between IDS alarms. This is to the best of our knowledge the first comprehensive privacy leakage model for IDS rules based on quantitative measurements of information flow founded in information theory.

The proposed privacy leakage metric has several practical applications. First, it can be used to identify imprecise IDS rules, since such rules typically will have more variation in the underlying data, and therefore also a larger variance in entropy than more precise IDS rules. Furthermore, an advantage with the proposed metric is that it can detect two common ways of preserving privacy or data confidentiality: anonymisation and pseudonymisation. Both encrypted and anonymised information can be expected to have zero entropy variance, given sufficiently long input. On the other hand, the entropy variance of plaintext data will be significantly larger than for encrypted data, as will be discussed in Section 6.5.3.

This means that the entropy variance can be used as a metric to detect leakage of private or confidential information in message oriented data streams in general and IDS alarms in particular. It can also be used to verify whether an anonymisation/pseudonymisation or encryption scheme works as intended.

This chapter is organised as follows: Section 6.2 discusses the motivation behind introducing an entropy variance based information leakage metric, based on existing knowledge of how common attack vectors work. Section 6.3 describes the threat model and scenario that is assumed when using the privacy leakage metric. Section 6.4 develops the entropy-based privacy leakage model based on quantitative information flow analysis after introducing the necessary theoretical background information. The last part discusses how clustering based on the Expectation Maximisation algorithm can be used to identify the underlying attack vectors for IDS rules that detect more than one attack vector. Section 6.5 does a detailed analysis of the convergence speed as a function of amount of input data for the entropy algorithms and symbol definitions considered. This includes analysing the metrics' abilities to distinguish between plaintext and encrypted data. Section 6.6 analyses experimental results based on realistic measurements of IDS alarms. Section 6.8 discusses related works; Section 6.9 concludes the chapter and Section 6.10 suggests future work and research opportunities.

6.2 Motivation

A precise IDS rule will in many cases report only one or a few different attack patterns corresponding to real attack vectors, as will be discussed below. One common type of attack vector that follows this behaviour, is stack or heap buffer overflow attacks [140]. These attack vectors frequently use large sequences of characters corresponding to the NOP operation or similar to increase the probability of successfully exploiting buffer overflow vulnerabilities. The attacker does then not need to know the exact memory location of injected shellcode, since returning to any address within the NOP sled will cause the shellcode to be executed. This makes it simpler for the adversary to exploit such vulnerabilities. The entropy of this NOP sled will be zero, and variance zero, as long as only NOP operations are being used in the sled and the attack vector does not mutate (e.g. by changing the length of the NOP sled). This is clearly distinguishable from ordinary traffic, and also easy to distinguish for rule-based IDSs.

Such naive attacks are however not so common nowadays, because the IDS and anti-virus technologies easily can detect such anomalies in the input. It is therefore increasingly common that the adversaries obfuscate the attack vector. Obfuscation of the NOP sled can for example be done using metamorphic coding, which means that instructions in the sled are substituted with other instructions that effectively perform the same function [68]. Furthermore, it is now common practice that also the shellcode of the attack is being obfuscated by using encryption techniques. This means that the attack after the NOP sled contains a small decryption program, with a decryption key that decrypts the obfuscated shellcode before it is being run [125]. Even the decryption program can be hidden by using metamorphic coding techniques [125], although this is still not very common [102].

This means that obfuscated attack vectors can be expected to have *quite high entropy*, in some cases *indistinguishable* from encrypted traffic [125, 58]. This means that the *variation in entropy* can be expected to go towards zero for a sufficiently large data sample from a polymorphic attack vector, given that it is indistinguishable from a perfect encryption scheme. Such an attack vector will behave like random uniform data. This means that the entropy variance of sufficiently large attack vector samples from both traditional

NOP sled based attacks and modern obfuscated attacks also can be expected to have *low* entropy variance.

It can furthermore be observed that samples of encrypted user traffic, assuming that strong encryption is used, in itself does not leak any private or confidential information, hence can be expected to have low entropy variance. Ordinary non-encrypted user traffic, can however be expected to show a significant variance in entropy between different samples, as illustrated in Figure 6.5.2. This indicates that entropy variance may be an interesting metric for measuring whether IDS alarms leak information, in particular for buffer overflow type of attacks. However this metric does obviously not understand the semantics of the data traffic, and can therefore not be used to evaluate whether the leaked information is private or confidential.

There also exist attack vectors that are indistinguishable from plaintext data. Examples of such attacks are nonobfuscated Javascript Trojans or SQL injection attacks. This means that the entropy standard deviation not necessarily can be assumed to be close to the extreme points: encrypted data (entropy close to 1) or NOP sleds (octet-entropy close to 0). However, there are still some other useful characteristics of such plaintext attacks in particular, and malware in general, that can be exploited by such a metric:

- Attacks are to a great extent automated and performed by large botnets of compromised hosts.
- Attack vectors do typically not yet mutate or change dynamically[1]. This means that multiple attacks by a given host being controlled by an adversary typically has the same payload. Different hosts running the same version of a given malware can also be expected to typically have the same payload [102].
- Attack vectors are modular programs that are improved incrementally, which means that not all parts of a malware will change at the same time, and some parts of malware code are even shared between different malware families [102].
- Botherders, that manage large botnets of compromised hosts, will also have a self interest in a "well managed" botnet. This means that the

[1] Although proof-of-concept polymorphic self-mutating worms has been demonstrated [73].

> malware of a botnet at regular intervals will be upgraded to include patches and new functionalities, amongst others to avoid being detected by Anti-Virus and IDS [54]. It is therefore reasonable to believe that a large amount of the machines in a given botnet will run the same version of the malware and therefore also will use the same arsenal of attack vectors for attacking other hosts.

This means that if an IDS rule is able to detect a given attack, or attack variants, then there are several reasons to believe that the entropy variance between instances of the same attack vector may be small, even for nonobfuscated Javascript or SQL injection attacks. This furthermore means that if the underlying attack vectors detected by an IDS rule can be identified, then the entropy variance (or entropy standard deviation) around each attack vector can be considered a measure of the precision of that rule hence also an indicator of possible privacy leakages.

6.3 Threat Model

The chapter assumes that intrusion detection services have been outsourced to a third party Managed Security Service (MSS) provider. Security monitoring is furthermore subdivided into two different security levels. An outsourced first-line service that is doing 24x7 monitoring of the computer networks, and a trusted second-line service that will have full knowledge of the IDS service, including capabilities to perform data forensic analysis. It is assumed that the MSS provider operates using a privacy-enhanced IDS, so that changes to the IDS ruleset must be agreed upon by both the data controller and the second line security analyst responsible for updating the IDS ruleset, to avoid that excessively privacy violating IDS rules are being deployed.

It is therefore assumed that the IDS services run in a controlled environment, where enforcement of a privacy policy supported by privacy leakage metrics makes sense. An example of such an environment is critical infrastructures or hospitals where security services have been outsourced to a third party, and privacy metrics are required to ensure compliance both to privacy and security policies. These policies must ensure that the first-line security analysts, that are not trusted to see sensitive information, do not get access to information considered private or confidential by the owner of the critical infrastructure. The objective is a stricter enforcement of the need-to-know principle than what IDSs typically have today. However, in order to enforce such privacy and security policies, suitable privacy metrics are needed, which will be developed here.

This chapter mainly focuses on two adversaries: an external adversary that may want to manipulate the privacy metrics for example to reduce the chance of attacks being detected. The IDS ruleset is assumed public, so that an external adversary can investigate how the IDS rules work in order to perform targeted attacks on either privacy or security. However the external adversary will not know which IDS rules that are enabled.

Insiders are divided into two main groups. First-line security analysts are considered untrusted insiders, that only have limited authorisation to see information and no authorisation to modify information related to the IDS configuration. They do not have access to the data forensic tool to investigate attacks in detail. Second-line analysts are considered a trusted CERT team, that has authorisation to perform security investigations and reconfigure the IDS. A third actor is the data controller, who shares the responsibility for managing the IDS ruleset with the security officer, to ensure that both the privacy and security objectives are being considered. The chapter furthermore assumes that suitable enforcement mechanisms exist, for example anonymisation or pseudonymisation schemes for sensitive information in IDS alarms, so that the privacy leakage metrics can be used for verification of the security or privacy policies.

6.4 A Privacy Leakage Model of IDS Rules

This section will first provide an information theoretic analysis of privacy leakage from IDS alarms, assuming a simple model of a perfect IDS rule R_P that does not have any false alarms. This model is subsequently generalised to handle IDS rules that may leak potentially sensitive information, and we then show how this model corresponds to measuring the standard deviation of entropy from the IDS rule. It is finally shown how to measure the privacy leakage from IDS rules that detect more than one attack vector.

6.4.1 Basic Definitions

The definitions and notation in this section give a short introduction to quantitative information flow analysis, and is based on [122]. It is throughout this chapter assumed that the logarithm is taken to the base 2, i.e. $log(x)$ means $log_2(x)$. Shannon and Min-entropy can be considered instances of the more general Rényi entropy [107], and we therefore use the Rényi notation to describe the entropies. Any Rényi entropy metric is denoted as $H_\alpha(X)$, where α is the entropy degree; $\alpha = 1$ represents Shannon entropy and $\alpha = \infty$ represents Min-entropy. Given an IDS rule R, which may leak sensitive information from a set of input data X and to a set of IDS alarms Y, the objective is then to measure how much information R leaks.

Let X and Y be random variables whose set of possible values are $\mathscr{X}$ and $\mathscr{Y}$ respectively. The Shannon entropy is then defined by [118]:

$$H_1(X) = \sum_{x \in \mathscr{X}} P[X = x] log \frac{1}{P[X = x]} \tag{6.4.1}$$

Shannon entropy indicates the number of bits that are required to transfer X in an optimal way. The conditional entropy denoted as $H_1(X|Y)$ indicates the expected resulting entropy from input data X given a set of IDS alarms Y that pass through the IDS rule R [122]:

$$H_1(X|Y) = \sum_{y \in \mathscr{Y}} P[Y = y] H_1(X|Y = y) \tag{6.4.2}$$

where

$$H_1(X|Y = y) = \sum_{x \in \mathscr{X}} P[X = x|Y = y] log \frac{1}{P[X = x|Y = y]} \tag{6.4.3}$$

Min-entropy is another entropy metric that is calculated based on the worst case (maximum) symbol occurrence probability, defined as the vulnerability $V(X)$ that an adversary can guess the value of X correctly in one try [122]:

$$V(X) = \max_{x \in \mathscr{X}} P[X = x] \tag{6.4.4}$$

Min-entropy indicates the number of bits required to store $V(X)$, and is defined as [122]:

$$H_\infty(X) = log \frac{1}{V(X)} \tag{6.4.5}$$

The conditional min-entropy can be defined as [122]:

$$H_\infty(X|Y) = log \frac{1}{V(X|Y)} \tag{6.4.6}$$

where

$$V(X|Y) = \sum_{y \in \mathscr{Y}} P[Y = y] \max_{x \in \mathscr{X}} P[X = x|Y = y] \tag{6.4.7}$$

It is then possible to define the information leakage L_{XY} from X to Y using either Shannon or Min-entropy as proposed by [122]:

$$L_{XY} = H_\alpha(X) - H_\alpha(X|Y). \tag{6.4.8}$$

6.4.2 Perfect model IDS Rule

Assume a perfect model IDS rule R_P, that always detects the attack vector and does not have any false alarms or other entropy sources. Furthermore assume that the given attack vector does not change between different attack instances. The payload sample in the IDS alarm from R_P is also assumed to not contain any other entropy sources. The IDS will in this case always sample the *same* payload excerpt in every alarm according to the attack pattern definition.

This IDS rule is termed a perfect model IDS rule, since it is considered perfect according to the theoretical model of privacy leakage. R_P is in other words a perfect model of IDS rule *behaviour* from a privacy perspective. This is not a purely theoretical IDS rule behaviour. We observed three IDS rules that behaved like R_P in our experiments, for example the Snort IDS rule with

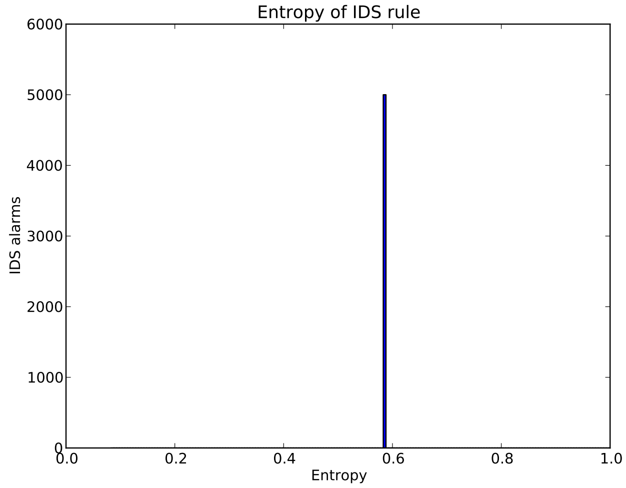


Figure 6.4.1: IDS rule 1:2003 SQL Worm Propagation attempt, behaving like R_P.

SID 1:2003 SQL Worm Propagation attempt, as shown in Figure 6.4.1. This is obviously a simplistic model of an IDS rule, since it does not handle the fact that many IDS rules and also non-rule based technologies like anomaly-based IDS will be able to detect more than one attack vector, and also variants of attack vectors. The model is furthermore oblivious to whether the source of entropies is adversarial or ordinary user activities. An entropy-based metric can only measure whether information is leaking or not. Therefore the privacy impact I will need to be evaluated, as discussed earlier.

The perfect model IDS rule will under these assumptions provide a *constant* leakage denoted as c of information in each alarm, corresponding to the pattern matched by R_P.

The *privacy impact I* of this constant information leakage as a *privacy leakage* is however not known. The privacy impact of the information leakage from each IDS rule must therefore be evaluated by a data controller, to determine whether the expected information leakage from the IDS rule can be considered necessary and acceptable from a security perspective, and also that the effective privacy impact from the rule can be considered negligible if the rule is effective over time.

This manual quality assurance procedure makes it possible to detect and avoid IDS rules where $I \cdot c$ in itself is judged to cause a significant privacy leakage, for example if the rule itself triggers on person sensitive information. The privacy leakage $I \cdot c$ from each installed IDS rule is therefore in the rest

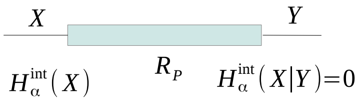

X
Y
$H_\alpha^{\mathrm{int}}(X)$
R_P
$H_\alpha^{\mathrm{int}}(X|Y)=0$

Figure 6.4.2: Channel model of a perfect model IDS rule R_P that detects a single, nonchanging underlying attack.

of this chapter considered as either necessary or negligible. If this constant privacy leakage is not considered tolerable, then it is assumed that this can be mitigated using anonymisation or pseudonymisation policies.

R_P will under these assumptions always triggers on the same attack pattern $Y = \{y\}$, as illustrated in Figure 6.4.2. The *inter-alarm entropy*, assuming a set of input data X, denoted as $H_\alpha^{int}(X|Y)$, is defined as the entropy between different IDS alarms, calculated over the entire payload excerpt (i.e. each IDS alarm is considered as one “symbol”). The inter-alarm entropy will in this case be $H_\alpha^{int}(X|Y) = 0$, since $P[Y = y] = 1$. This means that a perfect model IDS rule according to this definition from an *information theoretical* perspective does not reveal any *additional* information apart from what can be inferred from the limited and constant information leakage c in each alarm.

This does not mean that additional leakage of sensitive information cannot occur, since the resulting privacy leakage also will depend on the timing and context of the alarms. Additional information may for example be revealed by correlating the interdependencies between the IDS rules.

However, under the given assumptions, this means that when R_P triggers, then a known data pattern will have been sent in the input data stream. This information leakage is considered a tolerable privacy leakage under the assumptions in the previous subsection.

6.4.3 A Non-perfect IDS rule R

Then consider a non-perfect IDS rule R, which in addition to the assumed necessary and limited information leakage by the attack pattern, also may have false alarms or other entropy sources, as illustrated in Figure 6.4.3. However, it still only detects one attack vector, that does not change between

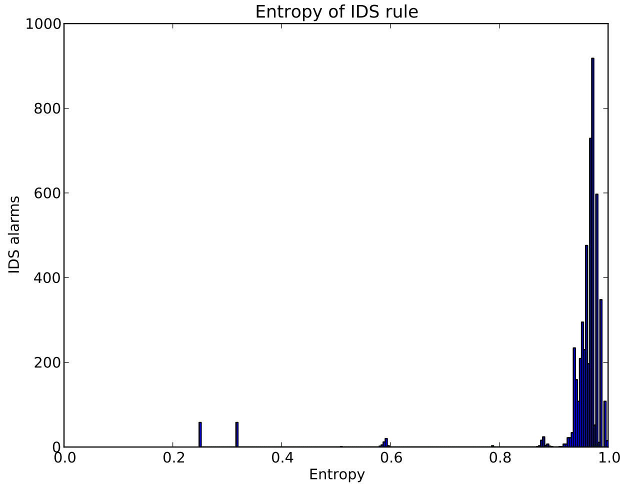


Figure 6.4.3: IDS rule 1:2925 1x1 GIF attempt (web bug), illustrating a privacy leaking IDS rule.

attacks. This means that the entropy distribution function will be unimodal, perhaps with some outliers as illustrated in Figure 6.4.3. This is a simplistic model of how an IDS rule behaves. It does not assume any particular IDS rule implementation (e.g. whether string matching or regular expressions are being used) and does not take any position on the type of IDS technology being used. Experimental results have however shown that a significant amount of all IDS rules (35-53% in the experiments we have performed[2]) actually *behave* in this way. However, this also means that many IDS rules actually do *not* behave this way. We will therefore later discuss how this restriction can be removed.

The model of a unimodal non-perfect IDS rule is illustrated in Figure 6.4.4. Assume that this IDS rule generates the ordered set of N IDS alarms denoted as $Y = \{y_1, y_2, ..., y_N\}$, where $P[Y = y_i] < P[Y = y_j]$ for $i < j$, $i, j \in 1, 2, ..., N$. The inter-alarm entropy will in this case be greater than zero for both Shannon and Min-entropy, because $\sum_{i=1}^{n} P[X|Y = y_i] = 1$ and $P[X|Y = y_1] < 1$.

[2] 53% of the IDS rules in the experiments performed here were unimodal, indicating one attack vector. A former pre-experiment at a commercial MSS provider indicated that 35% of the IDS rules were unimodal.

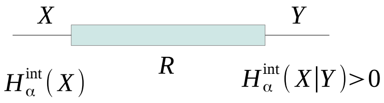


Figure 6.4.4: Channel model of a non-perfect IDS rule R that detects a single nonmutating underlying attack vector. R may have false alarms or other entropy sources, which means that $H_\alpha^{int}(X|Y) > 0$.

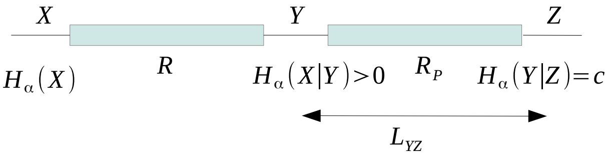


Figure 6.4.5: Channel model of privacy leakage from a non-perfect IDS rule R, measured relative to a perfect model IDS rule.

6.4.4 Privacy Leakage Model

The next question is how to model the *privacy leakage* from the non-perfect IDS rule R. One way to do this, is to measure the information leakage of the non-perfect IDS rule R *relative* to a perfect model IDS rule R_P, as illustrated in Figure 6.4.5. The communication channel then consists of a cascade of two IDS rules (or two IDS rules connected in series), where the output of the first IDS rule serves as input to the second IDS rule. Both IDS rules have the objective to trigger on the same attack vector, however the first IDS rule R is non-perfect, and may have false alarms or other entropy sources, whereas the second IDS rule R_P is considered a perfect model IDS rule. The advantage of using a cascading model, is that this allows for comparing known values, and it is not dependent on the unknown Internet traffic X. The set of alarms Y from R are known by the MSS provider and the set of expected alarms Z from R_P are also known given Y.

Focusing on the inter-alarm entropies is not a fruitful approach here, since the difference in inter-alarm entropies is $H_\alpha^{int}(X|Y) - H_\alpha^{int}(Y|Z) = H_\alpha^{int}(X|Y)$, because $H_\alpha^{int}(Y|Z) = 0$. What is needed, is therefore a measure of the limited information leakage that the perfect model IDS rule causes.

This initial information loss, denoted as the *intra-alarm* information loss

$H_\alpha(X)$, can be expressed by measuring the entropy of the IDS alarm in bits, instead of measuring the inter-alarm entropy H_α^{int} (the entropy between IDS alarms, considering the entire IDS alarm as one symbol). The intra-alarm entropy for a perfect model IDS rule R_P can be calculated by assuming that the IDS alarm consists of a large sequence of bits. This can be expressed formally by considering a given IDS alarm as $y \in \{0,1\}$ where $P[y=1] = 1-P[y=0]$.

Considering the perfect model IDS rule first, then this IDS rule will always return the same IDS alarm $Z=\{y\}$ where $y \in \{0,1\}$ with bit-probability $\{P[y=0], P[y=1]\}$. The information leakage is defined according to (6.4.8) as:

$$L_{YZ} = H_\alpha(X|Y=y) - H_\alpha(Y=y|Z=y) = H_\alpha(X|Y=y) \tag{6.4.9}$$

Since R_P is deterministic, then Z will be determined by Y, which means that $H_\alpha(Y|Z)=0$. Furthermore, for Shannon entropy:

$$H_1(X|Y=y) = \sum_{x\in\{0,1\}} P[X=x|Y=y] log \frac{1}{P[X=x|Y=y]} \tag{6.4.10}$$

$P[X=x|Y=y]=0$ for $x \neq y$, which means that this can be expressed as:

$$H_1(X=y) = \sum_{y\in\{0,1\}} P[X=y] log \frac{1}{P[X=y]} \tag{6.4.11}$$

which gives:

$$H_1(X=y) = P[y=0] log \frac{1}{P[y=0]} + \tag{6.4.12}$$

$$(1-P[y=0]) log \frac{1}{(1-P[y=0])} = c_1 \tag{6.4.13}$$

This shows that R_P has a constant privacy leakage $L_{YZ}=c_1$ for Shannon entropy. This can also be shown for Min-entropy by substituting into Equation (6.4.6):

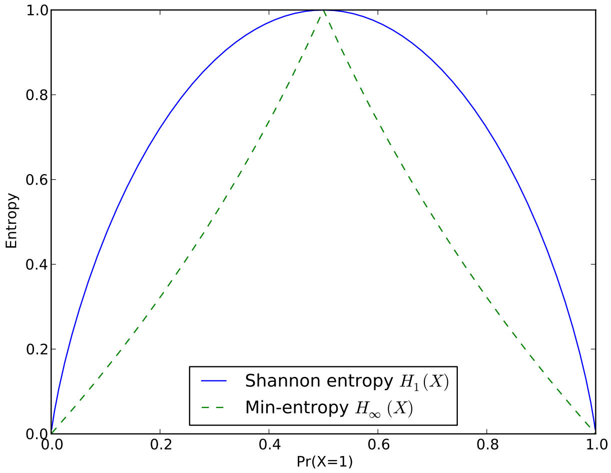


Figure 6.4.6: Shannon vs. Min-entropy.

$$H_\infty(X|Y=y) = log\frac{1}{V(X|Y=y)} \tag{6.4.14}$$

where the vulnerability $V(X|Y=y)$ can be expressed as:

$$V(X|Y=y) = \sum_{y\in\{0,1\}} P[Y=y] \max_{x\in\{0,1\}} P[X=x|Y=y] \tag{6.4.15}$$

$P[X=x|Y=y]=0$ for $x\neq y$, which means that this can be expressed as:

$$V(X|Y=y) = P[y=0]^2+P[y=1]^2 \tag{6.4.16}$$

which can be expressed as

$$V(X|Y=y) = 1-2P[y=0](1-P[y=0]) \tag{6.4.17}$$

This shows that the vulnerability is $V(X|Y=y)=1$ for $P[y=0]\in\{0,1\}$. The lowest vulnerability is $V(X|Y=y)=\frac{1}{2}$ for $P[y=0]=\frac{1}{2}$, as expected. This means that the Min-entropy for R_P can be expressed as:

$$H_\infty(X|Y=y) = log\frac{1}{1-2P[y=0](1-P[y=0])} = c_\infty \tag{6.4.18}$$

This means that R_P has a constant information leakage for both Shannon-entropy $L_{YZ} = c_1$ and Min-entropy $L_{YZ} = c_\infty$. However these constants are different, except in the special cases where $P[y = 0] \in \{0, \frac{1}{2}, 1\}$, as can be expected (see Figure 6.4.6).

Let the constant information leakage for either Shannon or Min-entropy be denoted as c_α. The relative information leakage from the IDS rule R can then be formally defined as follows:

Let R be a non-perfect IDS rule, that in addition to the assumed necessary and limited information leakage by the attack pattern, also may have false alarms or other entropy sources. Let R_P be a perfect model IDS rule with a limited privacy leakage c_α, $\alpha \in \{1, \infty\}$[3]. The relative information leakage L_{YZ} for an IDS rule R with input X, that generates a set of IDS alarms $Y = \{y_1, y_2, ..., y_N\}$, each with probability $P[Y = y_i]$, $i = 1, ..., N$ is then defined as the difference in intra-alarm entropy between R and a perfect model IDS rule R_P that both trigger on the same attack vector:

$$L_{YZ} = H_\alpha(X|Y) - c_\alpha \tag{6.4.19}$$

If the probability distribution function (PDF) of the IDS alarm entropies for a given attack vector is symmetric, then the *average* entropy denoted as $\overline{H_\alpha}(X|Y)$ for input X and a sufficiently large set of IDS alarms Y can be considered as a good estimator of c_α. For skewed distributions, the *median* may give a better estimate, given that the sample is sufficiently large. It can furthermore be observed that the precision of this estimator will improve with the precision of the IDS rule R. This means that the information leakage of R for a given IDS alarm y_i can be expressed as:

$$L_{YZ} = H_\alpha(X|Y = y_i) - \overline{H_\alpha}(X|Y) \tag{6.4.20}$$

where the average entropy can be expressed as

$$\overline{H_\alpha}(X|Y) = \sum_{i=1}^{N} P[Y = y_i] H_\alpha(y_i) \tag{6.4.21}$$

[3] It is possible to show that this definition generalises to any Rényi entropy, however that is beyond the scope of this chapter , since Min-entropy and Shannon-entropy are considered the best candidates for the privacy leakage metric [122].

for a set of input data X.

6.4.5 Information Leakage for a Sample of IDS Alarms

The average entropy per byte for a *sample* $y_1, y_2, ..., y_N$ of N IDS alarms generated by an IDS rule R that detects a single attack vector, can be expressed as

$$\overline{H_\alpha} = \frac{1}{N}\sum_{i=1}^{N} H_\alpha(y_i). \tag{6.4.22}$$

The information leakage for any IDS alarm y_j, denoted as $L_R(y_j)$ can then be expressed as:

$$L_R(y_j) = H_\alpha(y_j) - \frac{1}{N}\sum_{i=1}^{N} H_\alpha(y_i) \tag{6.4.23}$$

Further processing of the information leakage $L_R(y_i)$ for the IDS alarms $y_1, y_2, ..., y_n$ can now be calculated using traditional statistical analysis. The privacy leakage of the IDS rule can be expressed as the standard deviation σ_α, error margin $2\sigma_\alpha$ or the 95% confidence interval $\pm 2\sigma_\alpha$ of the IDS rule. This gives an indication of the expected precision of the IDS rule. Another useful metric, is to consider the worst-case information leakage denoted as L_R^{max} where $L_R^{max} = \max_{i=1}^{n} L_R$, or the minimum information leakage denoted as L_R^{min} where $L_R^{min} = \min_{i=1}^{n} L_R$. Both of these can be useful in statistical analyses, in addition to the standard deviation. Furthermore, the privacy leakage can be calculated as $\pi_R^L = L_R \cdot I_R$, where I_R is the privacy impact estimated by the data controller.

6.4.6 Sample Standard Deviation of Entropy σ_α

Normal Distribution

Assuming that the probability distribution of alarms can be approximated using a Normal distribution, then the standard deviation can be calculated using the second norm.

Assume that the IDS generates a sample of n IDS alarms $(y_1, y_2, ..., y_N)$.

Each alarm y_i contains payload or other potentially privacy leaking elements or attributes from the IDS alarms generated by an IDS rule R. The sample standard deviation of the entropy of the elements can then be expressed as:

$$\sigma_\alpha = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(L_R)^2} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(H_\alpha(y_i) - \overline{H_\alpha})^2} \tag{6.4.24}$$

The general properties of the *variance* of entropy measurements σ_α^2 will fulfill the same requirements as the standard deviation of entropy measurements. However, the standard deviation is considered more appropriate, since it operates with the same unit of measure as the entropy.

Laplacian Distribution

An alternative distribution that during the experiment was shown to fit the data well, is the Laplacian (or double exponential) distribution. The Laplacian standard deviation, denoted as σ_α^L is based the L^1 norm (or Manhattan distance), and can be expressed as the sum of absolute deviations:

$$\sigma_\alpha^L = \frac{\sqrt{2}}{N}\sum_{i=1}^{N}\left|H_\alpha(y_i) - \overline{H_\alpha}\right| \tag{6.4.25}$$

A well known advantage with σ_α^L, is that it will be less influenced by outliers in the tail of the PDFs than the standard deviation of the Normal distribution.

The standard deviation of normalised entropy is a measure of the *relative information leakage* from an IDS rule, under the assumption that it detects only one nonmutating attack vector. If an IDS rule detects the attack vector perfectly without any false alarms, then the entropy of the IDS alarms will always be the same, and $\sigma_\alpha = 0$. If the IDS alarm is precise at detecting the attack, then only a few bits of information will vary between IDS alarms. This means that all alarms will have similar entropy with low standard deviation and therefore also low information leakage. However if the IDS rule also has a significant amount of false alarms, or gets entropy from other sources then the entropy variance, and therefore also the information leakage from the IDS rule, will increase.

6.4.7 Aggregating σ_α

This subsection shows how the standard deviation of entropy metric can be aggregated for a set of IDS rules. Assume that an IDS uses a rule set denoted as R_{all} with m IDS rules $R_{all} = \{R_1, R_2, ..., R_m\}$. Each IDS rule R_i matches independently a set of N_i IDS alarms:
$Y_i = \{y_{i,1}, y_{i,2}, ..., y_{i,N_i}\}$, $i = 1, 2, ..., m$ where the number of IDS alarms N_i typically will vary between IDS rules. Furthermore, assume that the IDS alarms are independent and non-overlapping, i.e. $Y_i \cap Y_j = \emptyset$ for $i \neq j$. This means that all IDS alarms, denoted Y_{all}, can be expressed as $Y_{all} = \bigcup_{i=1}^{m} Y_i$.

Assume that an IDS rule R_i has entropy standard deviation denoted as σ_i and resulting standard deviation denoted as σ_{all}. The aggregated metric should furthermore fulfill the following criteria in order to provide meaningful aggregation:

C1 If all IDS rules have the same standard deviation, say σ_i, then σ_{all} should also be the same, i.e. $\sigma_{all} = \sigma_i$.

C2 The resulting entropy standard deviation should be weighted according to how many alarms that trigger on a given IDS rule R_i.

Each IDS rule should be assessed individually, in the same way as each underlying vulnerability should be assessed individually. This means that a *weighted average*, weighted by number of alarms from each IDS rule, can be used as aggregation function for σ_{all}, i.e:

$$\sigma_{all} = \frac{\sum_{i=1}^{m} N_i \sigma_i}{\sum_{i=1}^{m} N_i} \tag{6.4.26}$$

This function fulfills criterion C1, since the resulting average weighted sum is the same if σ_i is the same for all IDS rules R_i and it fulfills C2 by weighting the standard deviation against number of IDS alarms.

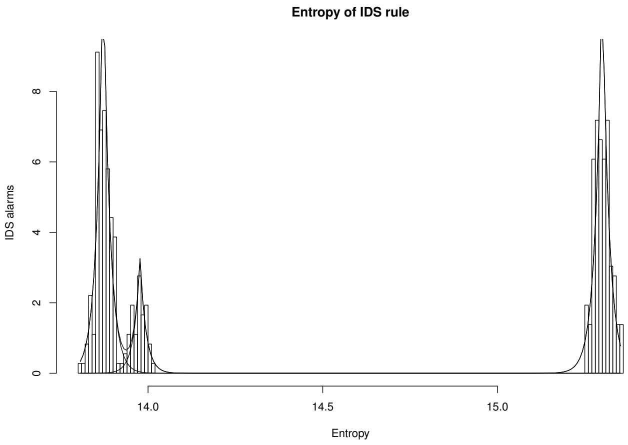


Figure 6.4.7: Payload length corrected Shannon octet-entropy distribution of IDS rule (Snort SID 1:11969) matching three attack vectors.

6.4.8 IDS Rules Detecting Several Attack Vectors

A significant part of the IDS rules will detect more than one attack vector, as illustrated in Figure 6.6.1. The data set used in this chapter has 47% of the IDS rules with more than one attack vector. An earlier preliminary experiment at a commercial MSS provider shows even higher percentage (65%). An indication of an IDS rule that detects several attack vectors, is that the entropy probability distribution is multi-modal. Figure 6.4.7 shows an example IDS rule that matches three privacy leaking attack vectors. The Figure shows the payload entropy distribution of the Snort IDS rule with SID 1:11969 VOIP-SIP inbound 401 Unauthorized. A payload length correction causes the metric to be larger than one, and is required to make the metric incentive compatible[4]. The details of this can be ignored for now, since this will be discussed in Section 6.5.4.

A clustering algorithm is needed to identify each underlying attack vector for multi-modal distributions. Each individual cluster will in this case represent an attack vector, which behaves in a similar way as a non-perfect IDS rule described in Section 6.4.3. This means that the privacy leakage of each attack vector cluster can be calculated as the entropy standard deviation over all samples belonging to the cluster, and the resulting privacy leakage for the IDS rule can be calculated by aggregating the data over all IDS rules in the

[4] Incentive compatibility – a characteristic of mechanisms whereby each agent knows that his best strategy is to follow the rules, no matter what the other agents will do [79].

cluster using Equation 6.4.26.

6.4.9 How to Perform the Clustering

There are two main types of clustering algorithms: hard clustering and soft clustering. Hard clustering algorithms assign each sample to a given cluster. Examples of a hard clustering algorithm is the popular k-means and k-medians algorithms [87, 20]. Hard clustering is however not appropriate for clustering the IDS rules, since it cuts off the samples at the tail of the distribution where two distributions overlap. This will give a bias towards lower entropy standard deviation than can be expected.

Soft clustering is then a better approach, since it assigns the probability that each sample belongs to a given cluster, instead of assigning each sample to a given cluster. A commonly used soft clustering technique is the Expectation Maximisation (EM) algorithm [36]. This soft-clustering method provides a Maximum Likelihood estimate of the underlying data distribution as a mixture of assumed probability distributions. The EM-algorithm is basically a two-step hill-climbing technique where the first step (E-step) calculates the expectation of the log-likelihood using the current estimate of the parameters of the underlying probability distributions. The second step (M-step) computes the parameters that maximise the expected log-likelihood identified during the E-step.

There are however some drawbacks with the EM-algorithm. It is prone to get stuck in local minima, which means that it is sensitive to the initial cluster parameters. We use the cluster centres identified by k-means, since this is a generally recommended method of initialising the cluster centres[5]. Another issue is the selection of number of clusters. Too many clusters may cause EM to overfit the data, whereas too few clusters may give a poor representation of the distribution of the samples.

It is commonly assumed that the underlying probability distribution either is a mixture of Gaussian or Laplacian probability density functions. Both outliers and skewedness have been found to be significant during the experimental analysis in Section 6.6. We have therefore decided to model the probability distribution as a mixture of Laplacian probability density functions using

[5] We used k-means from the Python module scikit-learn to initialise the EM algorithm [100].

the method proposed in [31]. This method is based on order statistics (uses a weighted median instead of the mean), and is therefore more robust against outliers and skewedness than using a Gaussian mixture [31]. The remainder of this section highlights the necessary theory and notation to understand how we have implemented the Laplacian mixture model based clustering.

6.4.10 Laplacian Mixture Model

This section defines the general notation, which is based on the well-known theory of learning finite mixture models [12, 49]. Furthermore, the Laplacian Mixture Model used here, is based on [31]. Our implementation is simplified compared to the original solution, since only univariate clustering is needed. Let $\mathscr{H}_R$ be a random variable representing the IDS alarm entropies of an IDS rule R, with H_α representing one particular outcome of $\mathscr{H}_R$. This random variable is expressed as:

$$P[\mathscr{H}_R = H_\alpha|\Theta] = \sum_{k=1}^{K} \beta_k P[\mathscr{H}_R = H_\alpha|\Theta = \theta_k] \tag{6.4.27}$$

where $\beta_1, ..., \beta_K$ are the mixing probabilities, each θ_k is the set of parameters defining the k-th component of the mixture and $\Theta = \{\theta_1, ..., \theta_K, \beta_1, ..., \beta_K\}$ is the complete set of parameters that define the mixture. Being probabilities, β_k must satisfy $\beta_k \geq 0$ and $\sum_{k=1}^{K} \beta_k = 1$. It is assumed that all the components of the mixture are Laplacian distributions $P[\mathscr{H}_R = H_\alpha|\theta_k] = \mathscr{L}(H_\alpha|\theta_k = (\tilde{\mu}, \lambda))$. The Laplacian distribution is defined as:

$$\mathscr{L}(H_\alpha|\tilde{\mu}_k, \lambda_k) = \frac{1}{2\lambda_k} exp\left(-\frac{|H_\alpha - \tilde{\mu}_k|}{\lambda_k}\right) \tag{6.4.28}$$

where $H_\alpha(y_i)$ is the entropy of the IDS alarm y_i, $\lambda_k > 0$ is the scale parameter and $\tilde{\mu}_k$ is the median for mixture component θ_k. In the remainder, assume the shorthand notation that $\mathscr{L}_{\alpha,i,k} = \mathscr{L}(H_\alpha(y_i)|\theta_k)$.

6.4.11 EM-Algorithm for Laplacian Mixture Model

The implementation of the EM-algorithm is based on [31, 49]. Assume that the EM-algorithm is performing cluster analysis on a *sample* of N ordered entropy values $\mathbf{H}_\alpha = (H_{\alpha,}(y_1), H_\alpha(y_2), ..., H_\alpha(y_N))$, where $H_\alpha(y_i) < H_\alpha(y_j)$ for $i < j$, $i, j \in 1, 2, ..., N$. These entropy values are calculated over the IDS alarms $y_1, y_2, ..., y_N$ generated by an IDS rule R. The Expectation Maximisation algorithm for the Laplacian Mixture Model then consists of two steps that are iterated until convergence is detected:

E-step: calculate the conditional expectation of the complete log-likelihood $w_{i,k} = log(P[\mathscr{H}_R = H_\alpha(y_i)|\Theta = \theta_k])$ that $H_\alpha(y_i)$ comes from the k-th component of the mixture:

$$w_{i,k} = \frac{\beta_k \mathscr{L}_{\alpha,i,k}}{\sum_{k=1}^{K} \beta_k \mathscr{L}_{\alpha,i,k}} \tag{6.4.29}$$

M-step: estimate new model parameters $\theta_k = (\tilde{\mu}_k, \lambda_k)$ and weights β_k that maximise the log-likelihood $log(\mathscr{L}_{\alpha,i,k})$ of the model:

$$\tilde{\mu}_k = wmedian(\mathbf{H}_\alpha, k) \tag{6.4.30}$$

$$\lambda_k = \frac{1}{\sum_{i=1}^{N} w_{i,k}} \sum_{i=1}^{N} w_{i,k} |H_\alpha(y_i) - \tilde{\mu}_k| \tag{6.4.31}$$

$$\beta_k = \frac{\sum\limits_{i=1}^{N} w_{i,k}}{\sum\limits_{i=1}^{N}\sum\limits_{k=1}^{K} w_{i,k}} \tag{6.4.32}$$

where the algorithm to calculate the weighted median for a given cluster k, according to [31], is described in Algorithm 6.1.

The algorithm uses the Minimum Message Length (MML) as stop criterion [142], assuming one-dimensional data. We do not go into details on the MML criterion and just present the implemented solution here. The detailed derivation of the MML criterion used can be found in [49].

$$\begin{aligned} MML &= \sum_{k=0}^{K} log(\frac{N\beta_k}{12}) + \frac{K}{2} log\frac{N}{12} + \frac{3K}{2} - \\ & max_k \left\{ \sum_{i=1}^{N} log(w_{i,k}) \right\} \end{aligned} \tag{6.4.33}$$

Algorithm 6.1 Weighted median.

```
function WMEDIAN($\mathbf{H}_\alpha, k$)
    $Q = (q_0 = 0, q_1 = 0, ..., q_N = 0)$
    sum=0
    for $i \leftarrow 1, ..., N$ do
        $sum \leftarrow sum + w_{i,k}$
        $q_i = sum$
    for $i \leftarrow 1, ..., N$ do
        if $q_i > \frac{1}{2} q_N > q_{i-1}$ then
            return $(H_\alpha(y_i) + H_\alpha(y_{i-1}))/2$
        else if $q_i = \frac{1}{2} q_N$ then
            return $H_\alpha(y_i)$
end function
```

The last term of Equation 6.4.33 is derived from the fact that the minimum of the *MML* criterion over Θ can be obtained by using the negative maximum of the log-likelihood (the last term), since

$$max_{\Theta}\{log\,(P[\mathscr{H}_R|\Theta])\} = max_k \left\{ \sum_{i=1}^{N} log(w_{i,k}) \right\}. \tag{6.4.34}$$

The algorithm stops when the difference in MML length between two iterations is less than $\varepsilon_{MML} = 1 \times 10^{-4}$. In addition to the MML criterion, the implementation of the EM algorithm requires at least 40 iterations to converge initially, and at least 20 iterations to converge after modifications of the cluster definitions. This is to avoid accidentally hitting a local MML minimum before convergence has occurred.

6.4.12 Determining the Optimal Number of Clusters k

We initially tested the method for estimating the number of components in [49]. This method worked for nice continuous distributions, however it did not work equally well for for noisy or a mixture containing binomial distributions, since the EM-algorithm then easily got stuck in local modes. Overfitting was also a significant problem for binomial distributions.

Furthermore, to judge whether a cluster should be interpreted as an attack vector or not typically requires that the data controller does some investigation of the IDS alarms. This means that some degree of manual intervention typically will be required during the clustering to assert obvious clusters that the clustering algorithm has missed or delete clusters where overfitting occurs. A typical example of overfitting is where several components with the same median are used to represent a given cluster. Another example is for skewed distributions, where the EM attempts to fit the skewed curve by overfitting the data.

We implemented a simple user interface for managing the clusters. It supports configuration of the initial number of clusters k as well as managing the model definition Θ after the initial configuration. The program also supports selecting type of entropy data and IDS rule to analyse from the datasets. The user interface for managing the clustering consists of the following functions:

setcl(k,$\tilde{\mu}_k$) Assert that the cluster number k has a mode at $\tilde{\mu}_k$.

delcl(clusterlist) Delete clusters at index *clusterlist*. Deleted clusters are marked with $\theta_k = (\mu_k = 0, \rho_k = 0, \beta = 0)$.

pickcl() Pick the cluster to be asserted by clicking the mouse at the position to be asserted in the histogram showing the frequency distribution of the IDS alarm entropies. If there are no clusters that are marked as deleted, then the least significant cluster (with lowest β_k) will be chosen.

After having modified the clusters, the EM-algorithm continues by typing the *cont* command in the debugger. When the data controller is satisfied with the cluster definition, typing *cont* without modifying the cluster causes the algorithm to finish and print out the calculated privacy leakage for each cluster and also the aggregated privacy leakage for the IDS rule R.

6.4.13 Calculating the Privacy Leakage for Clusters

The privacy leakage for the identified clusters is calculated after the data controller has asserted that the relevant clusters have been identified and that the EM-algorithm subsequently has converged. All probability mass is then assigned to the clusters, which means that the privacy leakage can be calculated for the given IDS rule R.

First, the model Θ will in itself give an indication of the privacy leakage in the form of the entropy standard deviation of the Laplacian function $\mathscr{L}(H_\alpha(y_i)|\theta_k)$ for a given cluster k. It is a well known fact that this can be calculated from the scale parameter λ_k for a Laplacian distribution as $\sigma_k^{\mathscr{L}} = \sqrt{2}\lambda_k$. However to be able to aggregate the entropy standard deviation over all clusters, the relative proportion of the samples for a given cluster θ_k must be estimated, which is exactly what β_k indicates. This means that the resulting entropy standard deviation for the IDS rule R can be calculated as the weighted average using Equation 6.4.26, substituting N_i with β_k:

$$\sigma_R^{\mathscr{L}} = \sum_{k=1}^{K} \beta_k \sigma_k^{\mathscr{L}}. \qquad (6.4.35)$$

A disadvantage by using $\sigma_k^{\mathscr{L}}$, is that this only will be correct if the model fits the data reasonably well. This may be true in some cases, however the sample distributions in the experiments do in several cases deviate significantly from the model due to outliers, heavy tails or noise. In these cases, it will be more correct to have a measure of σ_α that is based on the underlying samples $H_\alpha(y_i)$ weighted according to the conditional expectation $w_{i,k}$ of the model distributions defined by Θ, so that the weighted entropy is described by $w_{i,k}H_\alpha(y_i)$. This means that the model distributions is used to specify how the samples are divided between the clusters, instead of defining the clusters directly. The mean value of the cluster entropies for cluster k can then be expressed as:

$$\mu_k = \frac{\sum_{i=1}^{N} w_{i,k} H_\alpha(y_i)}{\sum_{i=1}^{N} w_{i,k}} \qquad (6.4.36)$$

and the Normal standard deviation can be expressed in a similar way as:

$$\sigma_k = \sqrt{\frac{\sum_{i=1}^{N} w_{i,k} \left(H_\alpha(y_i) - \mu_k\right)^2}{\sum_{i=1}^{N} w_{i,k}}}. \qquad (6.4.37)$$

Furthermore, the Laplacian standard deviation, based on the L^1 norm, can be expressed in terms of the conditional expectation $w_{i,k}$ and the median of the mixture component $\tilde{\mu}_k$ as:

$$\sigma_k^L = \sqrt{2}\frac{\sum_{i=1}^{N} w_{i,k}\,|H_\alpha(y_i) - \tilde{\mu}_k|}{\sum_{i=1}^{N} w_{i,k}}. \tag{6.4.38}$$

The resulting aggregated entropy standard deviation for the IDS rule R can in both these cases be calculated from Equation 6.4.35 by substituting the relevant standard deviation into the equation. The clustering analysis tool prints out both the individual standard deviations per cluster as well as the resulting standard deviation for the IDS rule based on both the standard deviation of the model $\sigma_R^{\mathscr{L}}$, Normal standard deviation σ_k and Laplacian standard deviation σ_k^L. It is useful to compare these, since a large deviation between $\sigma_R^{\mathscr{L}}$ and the other standard deviations indicate a poor model fit, which may or may not be relevant depending on examination of the underlying data.

One can for example expect good model fit for IDS rules with some Gaussian or Laplacian noise, since this is close to the expected model of privacy leakage. However very noisy rules that match random traffic will get a poor model fit. An example of this is the IDS rule 1:1394000 in our experiments that detects random traffic. It has a standard deviation over all data of 6.7 for both Normal and Laplacian standard deviation, but only a model standard deviation of $\sigma_{1:1394000}^{\mathscr{L}}$=1,44 . In such cases the standard deviation of the model $\sigma_R^{\mathscr{L}}$ will not be usable. Another example is if σ_k is significantly larger than σ_k^L, then σ_k may be unduly influenced by outliers, which means that σ_k^L would be the more robust estimate. In general, the Laplacian standard deviation can be expected to give the most conservative estimate, which is least influenced by skewedness and outliers.

6.4.14 Summary of EM-based Clustering

The Laplacian Mixture Model is implemented using the EM-algorithm. A semi-automatic process is used to identify the underlying clusters in the IDS alarms. The standard deviation of entropy metric is then calculated for each cluster and also the aggregated metric for the entire IDS rule. A possible attack on the clustering method, is an overfitting attack where a MSS provider decides to shirk by deliberately overfitting the attack vectors, by asserting too many clusters during the clustering process. It is therefore important that the role as data controller is separate from the role as security manager, and also that external quality assurance entities like certification organisations oversee the operation, to ensure that it is not overly privacy invasive. It must be emphasised that the objective not necessarily is to match the underlying probability distribution as closely as possible. The objective is rather to identify any likely attack vectors, and distribute the samples between these. The EM algorithm does this reasonably well.

The EM-based clustering generalises the privacy leakage metric to work for IDS rules that detect more than one attack vector. This generalisation is necessary, since our experiments have shown that a significant amount of all IDS rules trigger on more than one underlying attack vector. An advantage with this generalisation, is that it avoids the incentive incompatibility of the single cluster metric, which would encourage a shirking MSS provider to cheat by splitting up IDS rules into smaller IDS rules detecting a single attack vector.

6.5 Detailed Analysis of σ_α

This section does a more thorough investigation of the standard deviation of entropy metric σ_α. The objective of this discussion is to do an analysis of the convergence speed required to reliably detect random uniform input data as a function of the data length. It is expected that random uniform input data converges towards zero entropy standard deviation for a sufficiently long data series. This convergence speed is an important decision factor for the selection of entropy algorithm and symbol definition, since the IDS alarm entropies are calculated over a limited number of IDS alarms. Furthermore, it is discussed which metric and symbol definition that works best for distinguishing between plaintext and encrypted data. This analysis shows which entropy type (Min- or Shannon entropy) and symbol size (bit or octet) that is best for calculating privacy leakage in IDS rules.

6.5.1 Entropy Calculation

There are at least three obvious ways of selecting the symbol space that is used to calculate the entropies:

1. Define the payload of the IDS alarm as the symbol, i.e. calculate the *inter-alarm* entropy;
2. Use binary entropy, i.e. the *intra-alarm* entropy as described in Section 6.5.1;
3. Use octets, i.e. 8-bit words, which commonly are used to define the character set in computer systems.

Other word sizes are possible, however these are considered the most common and interesting ones for our purpose. Each of these symbol definitions have their advantages and disadvantages, and it is important to note that the entropy values calculated from each of these definitions typically will be different. It has already been shown that the intra-alarm entropy calculated from bit-entropy is different from the inter-alarm entropy by a constant value. Furthermore, the inter-alarm entropy is not possible to use, since it can not be used to calculate the standard deviation of entropy.

Bit-entropy was used to develop the Equation 6.4.20, since it is the easiest way to develop the theory for the privacy leakage metric. The entropy standard deviation formula is however not dependent on any particular symbol definition, as long as the symbol definition ensures that the entropy standard deviation in the worst case, i.e. for random, uniform data, can be measured to be sufficiently close to zero for encrypted traffic. It is assumed that σ_α converges towards zero for random, uniform data as a function of input data length, however the convergence speed is unknown and must be investigated. It can furthermore be observed that for a perfect encryption scheme that is approximated by random uniform data, the symbol definition does not matter, since random uniform data does not leak any information. This means that if the objective is to purely detect whether the information conveyed is encrypted or not, then the entropy scheme with fastest convergence speed may make sense to use.

This means that the minimum length of data required to reliably detect that random uniform data has zero variance (i.e. speed of convergence) is an important design factor that this metric relies on. It can be expected that different entropy metrics will have different convergence speed. In particular, can Min-entropy be expected to converge more slowly, since it only considers the maximum symbol occurrence probability, and not a weighted sum of all symbol occurrence probabilities, as Shannon entropy does.

6.5.2 Entropy Bias of Finite Length Encrypted Data

A question that needs to be investigated, is therefore how different entropy standard deviation metrics σ_α (Shannon- or Min-entropy) respond to random uniform data strings of varying length, and also how it is influenced by the symbol width, i.e. whether bit-entropy or octet-based entropy is used. The reason for this, as discussed in Subsection 6.4.6, is that the metric shall be able to measure privacy leakage sufficiently close to zero in the following three cases:

1. For a perfect model IDS rule R_P which detects and displays one or more non-changing attack vectors perfectly;
2. for anonymised IDS alarms from the IDS rule;
3. as a limit case for encrypted (e.g. pseudonymised) IDS alarms from the IDS rule, as the number of bits n in the IDS alarm goes towards infinity.

The entropy standard deviation bias for finite length encrypted data, denoted as σ_α^{bias}, can be analysed by simulating the response function of σ_α^{bias} as a function of number of bits of data. The simulation is based on a set of Monte-Carlo experiments, one for each octet of data. Each standard deviation is the average of an ensemble of 10000 experiments. Bit-length is calculated for each octet as eight times the octet length, in order to have comparable x-axis values for bit- and octet-based data. The experiments are based on simulations using random uniform data selection, which means that a Normal distribution can be assumed.

Figure 6.5.1 shows a log-log plot of the entropy standard deviations. The bit-entropies both appear to be log-linear, which means that the bias for detecting a perfectly encrypted IDS alarm with length n bits can be expressed as $log_2(\sigma_\alpha^{bias}) = log_2(\gamma_\alpha + \psi_\alpha n)$, where γ_α is the offset and ψ_α is the slope

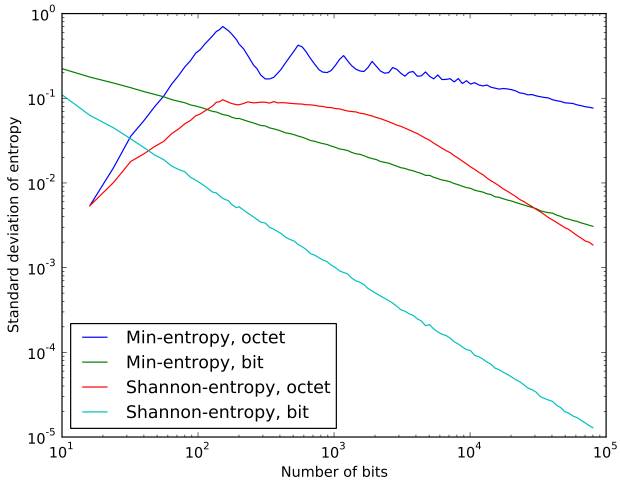


Figure 6.5.1: Log-log plot of entropy standard deviation as a function of number of bits input data for Min and Shannon entropy and bit and octet symbol definition.

of the log-log scale. This gives $\sigma_{\alpha}^{bias} = 2^{\gamma_\alpha} n^{\psi_\alpha}$, where 2^{γ_α} is constant. The slope can be calculated from the experimental data, which shows that that $\psi_1 = -1.005 \approx -1$ for Shannon bit-entropy and $\psi_\infty = -0.479 \approx -\frac{1}{2}$ for Min-entropy. This means that $\sigma_1^{bias} \approx \frac{2^{\gamma_1}}{n}$, whereas $\sigma_\infty^{bias} \approx \frac{2^{\gamma_\infty}}{\sqrt{n}}$, which means that Shannon bit-entropy converges by an order of $O(n^{-\frac{1}{2}})$ faster towards zero than Min-entropy[6]. Shannon bit-entropy has initially 2.7 times less bias than Min-entropy for perfectly encrypted (i.e. random uniform) data.

The octet-based entropies perform very poorly during the initial transient phase, but are then stabilised on a slope similar to the respective bit-entropy slopes, as shown in Figure 6.5.1. This means that there is a significant, but approximately constant, difference between the bit- and octet-based metrics after the initial transient phase. Shannon bit-entropy entropy ends up with a precision 143 times better than Shannon octet-entropy after 80 kbit. The difference in precision between bit- and octet-based Min-entropy is smaller, only 25 times.

A nice property is that the bias is systematic, which means that the entropy standard deviation calculations may be able to compensate for it by subtracting the expected bias from the entropy standard deviation, given that the number of samples (IDS alarms) is sufficiently large. *However, this only*

[6] This means that each factor in the bit-entropy calculations (one for Min-entropy and two for Shannon entropy) contributes with a convergence speed of $O(n^{-\frac{1}{2}})$.

makes sense if it is known that the data are encrypted. Since this in general is not known for the payload from IDS rules, and it will be wrong to correct for this bias for nonencrypted data, this means that the metric with fastest convergence speed is preferable.

It must also be noted that bit-based entropies (both Shannon and Min-entropy) are computationally less complex than octet-based Shannon entropy, which needs to calculate the weighted logarithm expression for each symbol in an octet. Counting the number of bits set to one in an octet or word (list of octets) can be done by calculating the Hamming weight, which is implemented in hardware on most modern Intel or AMD processors using the *popcnt* (population count) operator. This opens up for efficient implementations of bit-entropy calculations for up to 64 bits word chunks [32], which is more efficient than iterating to calculate the octet frequencies, as required by octet-based entropies.

6.5.3 Entropy Standard Deviation Difference between Encrypted and Plaintext data

Another foundational scenario that must be investigated, is how well the proposed entropy algorithms and symbol definitions distinguish between encrypted and plaintext information. The entire theory behind σ_α hinges on the assumption that there is a difference in entropy standard deviation between plaintext and as a limit case encrypted information. To determine whether this assumption is true or not, and which entropy configuration that works best, we set up another Monte-Carlo simulation, this time comparing the entropy standard deviation of plaintext data with the entropy standard deviation of random uniform data for both Min- and Shannon-entropy, using both bit and octet-based symbol definition.

The experiment configuration calculates the average and the 95% confidence band ($\pm 2\sigma$) from an ensemble of 10000 experiments. Each experiment calculates the standard deviation over 50 samples for varying input data length in bits, assuming that this is the smallest number of samples that in practice will be used to reliably distinguish between encrypted and plaintext data. If less samples are used per experiment, then the confidence band will widen out, meaning that longer payload will be needed to reliably distinguish

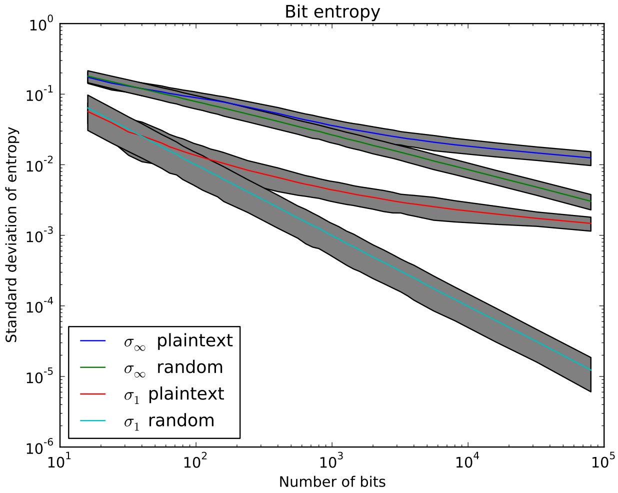


Figure 6.5.2: Difference and 95% confidence band between σ_α for plaintext and random data using bit-entropy for varying input data length in bits.

between encrypted and plaintext data. There is in other words a tradeoff between the payload length and the number of samples required to reliably detect encrypted content.

Random uniform data was measured in a similar way as the previous experiment. The plaintext data was extracted using randomly selected contiguous quotes from the Brown corpus [53], with varying data length in bits along the x-axis.

Figure 6.5.2 shows the difference between σ_α for plaintext and random data using bit- and octet-entropy respectively for varying input data lengths in bits. Shannon bit-entropy is the metric that distinguishes best between cleartext and encrypted data for data lengths greater than 400 bits (50 bytes).

Figure 6.5.4 shows that Shannon octet entropy is able to distinguish reliably between cleartext and encrypted data over a sample of 50 IDS alarms within a 95% confidence interval from 5 octets (40 bits) and onwards, despite the poor convergence properties for random traffic in the range $[5,131]$ octets.

However, due to the slightly hour-glassed shape of the entropy difference, it is not possible to achieve any larger precision between 40 and 3000 bits (375 bytes), unless the sample size is increased to narrow the confidence band sufficiently. Plaintext data is 11 times larger than encrypted data at 5 octets (40 bits), whereas at around 128 octets (1024 bits), it is down to 1.8 times larger than the encrypted data, before the random data reaches its knee

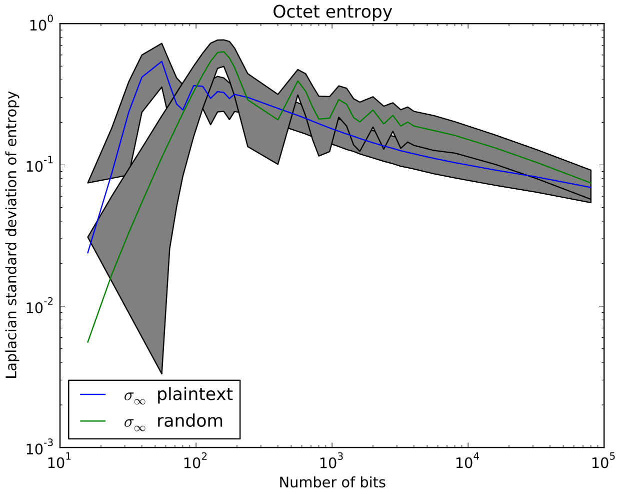


Figure 6.5.3: Difference and 95% confidence band between σ_∞ for plaintext and random data using Min-entropy for varying input data length in bits.

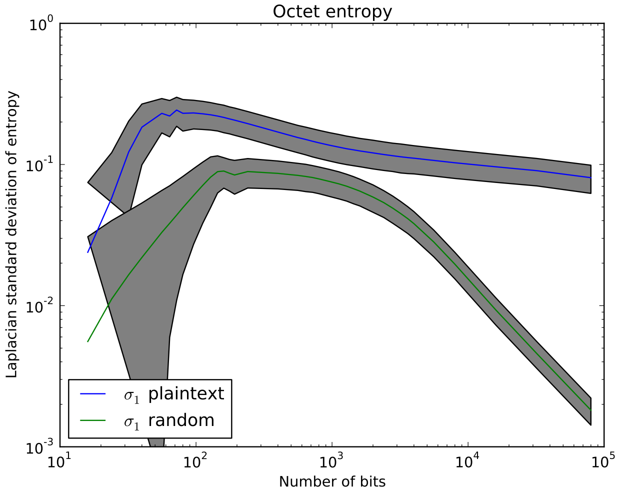


Figure 6.5.4: Difference and 95% confidence band between σ_α for plaintext and random data using normalised Shannon entropy for varying input data length in bits.

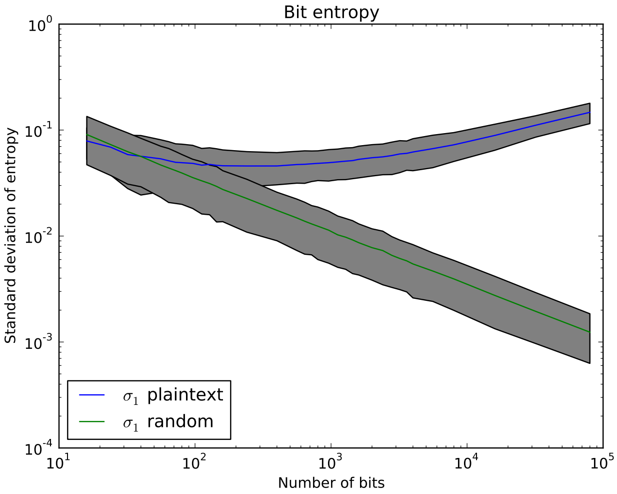


Figure 6.5.5: Payload length corrected Shannon bit-entropy with 95% confidence band as a function of input data length in bits for plaintext and random data.

point where the octet-based metric again improves.

Shannon bit-entropy is more well-behaved than Shannon octet entropy, in that the difference in entropy seems to be a strictly convex function, as opposed to the octet-based entropies. Min-bit-entropy also seems to be well behaved, and has the advantage that the 95% confidence band for Min-bit-entropy is narrower than for Shannon bit-entropy. However it is still overall a much poorer measure of entropy difference than Shannon bit-entropy, since it requires at least 6000 bits (750 octets) to reliably distinguish between plaintext and encrypted data. Octet-based Min-entropy, as shown in Figure 6.5.3, behaves extremely poorly, and is not usable for distinguishing between plaintext and encrypted text.

Overall, this strengthens the conclusion that Shannon entropy is the best metric, regardless of symbol definition since it converges faster than the other alternatives and it distinguishes better between cleartext and encrypted data as long as the payload is longer than the minimum threshold of 5 octets for octet-based entropy or 50 octets for bit-entropy for minimum 50 samples.

6.5.4 Payload Length Correction for Bit-entropy

A deficiency with the entropy standard deviation metrics, is that they decrease as the data length increases. This is the desired behaviour for random uni-

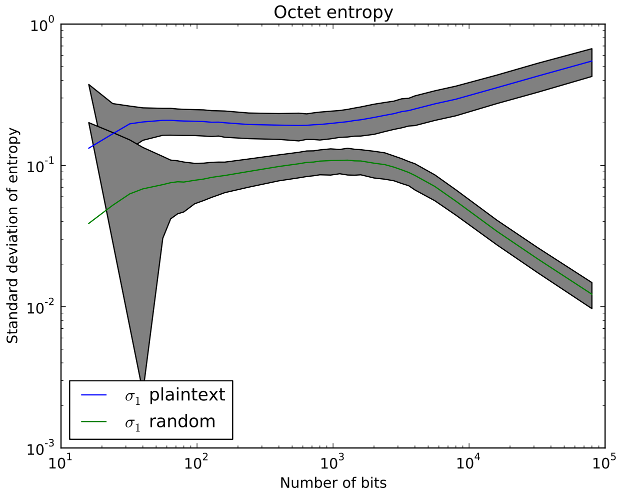


Figure 6.5.6: Payload length corrected Shannon octet-entropy with 95% confidence band as a function of input data length in bits for plaintext and random data.

form data, however it is not necessarily desirable for plaintext data, since this means that the metric can not be considered incentive compatible: it will then pay off for an adversary to match as large plaintext data packets as possible, since this in effect reduces the measured information leakage. An obvious way to mitigate this problem might be to multiply the entropy values with the length $n_i = |y_i|$ of the IDS alarm, i.e. $n_i H_\alpha(y_i)$, and then take the standard deviation of the length corrected entropy values. This correction will however be too strong, since the expected bias for random uniform data of length n_i then would be constant: $\sigma_1^{bias} \approx \frac{\gamma_1 n_i}{n_i} = \gamma_1$. This means that the metric would not converge to zero for encrypted traffic.

This problem can be mitigated by multiplying the entropy values with the *square root* of the payload length n_i. This means that the length corrected entropy values for bit-entropy can be described as $H_\alpha^{'}(y_i) = \sqrt{n_i} H_\alpha(y_i)$.

The length-corrected privacy leakage metric π_R^L, can be expressed as:

$$\pi_R^L = I \cdot \sigma_k^L = I \cdot \sqrt{2} \frac{\sum_{i=1}^{N} w_{i,k} \left| H_1^{'}(y_i) - \tilde{\mu}_k^{'} \right|}{\sum_{i=1}^{N} w_{i,k}}. \tag{6.5.1}$$

where $\tilde{\mu}_k^{'}$ is the median from the LMM.

The payload length corrected Shannon bit-entropy standard deviation function is shown in Figure 6.5.5. It can be observed that the term $\sqrt{n_i}H_\alpha(y_i)$ essentially reduces the convergence speed to detect random uniform traffic for Shannon entropy by a factor of $O(n^{-\frac{1}{2}})$ to $\sigma_1^{bias} \approx \frac{\gamma_1}{\sqrt{n}}$, similar to Min-entropy originally. However random uniform traffic will still converge towards 0, as required, although somewhat more slowly. Furthermore, the measured privacy leakage for plaintext data will now increase exponentially as a function of payload length, instead of decreasing, as long as the payload length is larger than the required 100 bytes (800 bits). This modification avoids the incentive incompatibility for Shannon bit-entropy, since the metric now increases with increasing payload length.

6.5.5 Payload Length Correction for Shannon Octet-based Entropy

Shannon octet-based entropy has the same convergence speed as Min-entropy after an initial transient phase, as shown in Figure 6.5.1. This means that $\sqrt{n_i}$ can be used as a length correction factor also for Shannon octet-entropy to ensure that the measured privacy leakage increases with the payload length for plaintext data, and decreases with the payload length for random data.

This length correction does however not work well below 200 octets, since Shannon octet entropy initially rises quickly until a knee point at 50 bits for plaintext data and 150 bits for random data, and then starts falling, as shown in Figure 6.5.4. It is desirable to reduce the effect of this knee point, in order to have an easier functional relationship between plaintext and random data, so that a fixed threshold can be used to distinguish between cleartext and random traffic. Introducing an additional length correction factor of $\frac{1}{log_2(n_i)}$ where n_i is the length of the payload y_i can be used to reduce the effect of this knee point, as shown in Figure 6.5.6. This means that the payload length correction function for Shannon octet-based entropy is $H_\alpha^{'}(y_i) = \frac{\sqrt{n_i}}{log_2(n_i)}$.

Payload length corrected Shannon octet-entropy standard deviation as a function of payload length is shown in Figure 6.5.6. The initial slightly hourglassed shape of the standard deviation functions means that the octet-based function despite the payload correction still is reduced slightly for plaintext data between 48 and 800 bits (5 and 100 bytes) payload length. This means that the metric is not entirely incentive compatible in this range, since it is

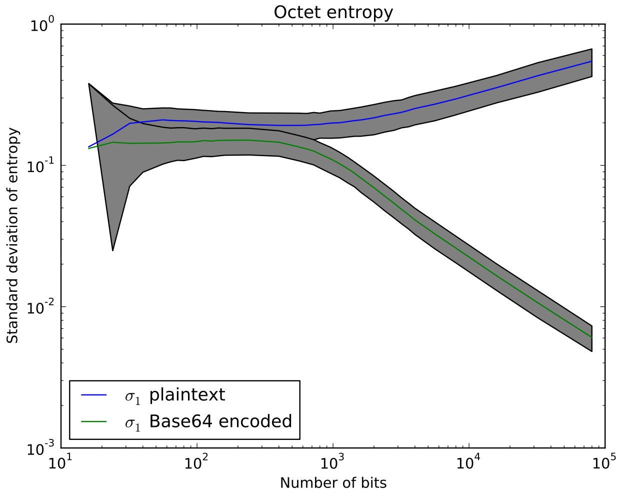


Figure 6.5.7: Payload length corrected Shannon octet-entropy with 95% confidence band as a function of input data length in bits for plaintext and Base64-encoded random data.

slightly decreasing for plaintext instead of increasing, however the deviation is not very large. The octet-based metric is however incentive compatible beyond 100 bytes, since the metric then increases with increasing payload length for plaintext data. An advantage with Shannon octet-entropy, is that it is able to detect whether short strings of data is encrypted or cleartext, for example from pseudonymisation schemes, assuming that the data is at least 5 octets and encrypted using a perfect encryption scheme.

Another advantage with the payload length corrected entropy metrics, is that a *fixed threshold* can be used to distinguish between plaintext and random data, regardless of payload length for a sufficiently large sample (minimum 50 samples). For Shannon bit-entropy this threshold is 0.028, whereas Shannon octet-entropy has a threshold of 0.14 (five times larger).

It must be noted that it is possible to construct data that falls between the two example entropies used here. The first example that comes to mind, is partially encrypted IDS alarms, where for example a header part is nonencrypted and a payload part is encrypted or coded (e.g. compressed). In these cases, some IDS alarms would be interpreted as encrypted, whereas others may be interpreted as nonencrypted. However, an advantage with the octet-based metric, is that relatively few octets are required to calculate it, which means that the header and remaining payload in such cases can be calculated separately.

6.5.6 Standard Deviation of Entropy for Base64-encoded Data

Another interesting case is how σ_α copes with quoting techniques used to transfer binary data on transport protocols that are not 8-bit clean. A common encoding technique is Base64-encoding, which can be used to transfer binary information in SMTP and XML-based formats like HTML or SOAP. Figure 6.5.7 shows the standard deviation of Shannon octet-entropy for plaintext and Base64 encoded random data. The Base64-encoding adds redundancy, which means that the encoded data is closer to plaintext data. This can be seen from the Figure, since the confidence bands now overlap for less than 800 bit (100 bytes). However, for longer input data, the Base64-encoded random data behaves in a similar way like plain random data, since the standard deviation goes towards 0.

This means that at least 100 bytes are required to reliably distinguish between Base64-encoded random data and plaintext data. If it is known that the information is Base64-encoded, then it will be possible to decode the information before the entropy is calculated. This may be useful if the information leakage of shorter Base64-encoded strings are being measured. However this decoding will add additional parsing overhead, which may not be desirable from a performance perspective. This is however avoidable, as long as the payload is larger than 100 bytes as shown above.

6.5.7 Semantic Information of Symbols

The symbol definition for the entropy algorithms will also need to take into account the semantic information that symbols convey. The definition of bytes (or more precisely octets) is in particular important for computer systems, since this is used to define the basic character set used for communicating both text and binary codes. Octet-based symbol definition is also important for many of the attack vectors discussed in the introduction. Buffer overflow attacks for example frequently use the single octet NOP instruction (0x90 on Intel machines) for the NOP sled. There also exist multi-octet NOP variants and other techniques for generating an obfuscated sled [4]. However for now consider single byte based NOP sleds, which are common, not the least because they are easier to exploit. Using this strategy means that the shellcode does not need to be placed on an exact 32- or 64 bits word boundary, as compilers typically enforce for normal programs [4].

The single-byte NOP sled (0x90) is a unique symbol for octet-based entropy, however for bit entropy, this represents the binary string 10010000, which has two out of eight bit set. The problem is that this value is not unique. There will in general be $8!/((8-2)! \cdot 2!) = 28$ different octets, where any combination of these can produce the same two-bit based entropy value as this NOP opcode. In fact, bit-entropy means that 256 different octet values are mapped down to only 9 different bit-entropies. Furthermore, whereas the octet entropy of a list of NOP opcodes will be zero, the bit-entropy will be greater than zero, except if all bits are '1' or '0'. The Shannon bit-entropy of the NOP sled is 0.81, which is very different from the octet-entropy (0). Furthermore, if one octet of information is changed, this means that somewhere between one and eight bits will change. There is in other words a less clear correlation between the change in information and change in entropy for bit-entropy than for octet-based entropy.

This means that octet-based entropy is closer to representing the *meaning* of the information being exchanged, and therefore should be the preferred symbol definition for the privacy metrics. The discussion above has also identified that the standard deviation of Shannon octet-entropy is the metric that overall has the best properties for distinguishing between cleartext and encrypted data, despite its poor convergence properties over part of the usable range. Octet-based entropy is furthermore able to uniquely identify that a sequence of the same octet has zero entropy, something bit-entropy does not identify. This means that Shannon octet-entropy will provide the largest possible difference in entropy between plaintext and strings consisting of sequences of the same character. Shannon octet-entropy is in other words a better privacy leakage metric than Shannon bit-entropy with better distinguishing capability according to our requirements and needs within the operating range. Min-entropy is not usable for our purpose.

6.6 Experimental Results

The experimental results are based on IDS alarms from my own home network between 2009 and 2011. Some of the IDS alarms are also from the KDD-Cup'99 data set. We included the 32 most noisy IDS rules with at least 50 IDS alarms per cluster in the measurements. The threshold of 50 IDS alarms per cluster is chosen to stay within the 95% confidence bands discussed in the simulations in Section 6.5.3. This is a limited data set that will not reflect the privacy leakage measured at a professional MSS provider doing large-scale measurements. The main difference that can be expected from a larger MSS provider, is that there would be a greater selection of IDS alarms with more than 50 alarms per cluster, and that the number of attack clusters would be greater. Furthermore, a larger set of IDS alarms may be enabled by commercial MSS providers to counter for emerging threats that are not yet in the Snort VRT ruleset, which we used. Furthermore, traffic from a commercial MSS provider would not be influenced by the synthetic KDD-Cup'99 data set.

However, despite these deficiencies, there are also some advantages by using our own data. One of the main advantages, is that this allows for discussing the IDS rules that may be leaking private or confidential information in detail, something that it according to our experience would be difficult or impossible to do for a commercial MSS provider due to business confidentiality and repudiation concerns. We have attempted to get agreement for such measurements for commercial MSS providers, however this is only possible if the IDS ruleset is not revealed, which makes it difficult to discuss in a convincing way that the proposed privacy leakage metrics work as intended. More elaborate tests at a commercial MSS provider is therefore left as future work. We decided to use a privacy impact factor $I = 1$ to only show the information leakage part of the privacy leakage metrics.

The experiment includes an IDS rule that we created *(sid:1:1394000)* which tests the worst-case scenario from a privacy perspective. This is a threshold-based IDS rule that essentially samples every 10th packet from the network. This is intended to show the maximum value that the privacy metric typically is able to detect, which is useful to see how far away the IDS rules in the measurements are from a worst-case scenario.

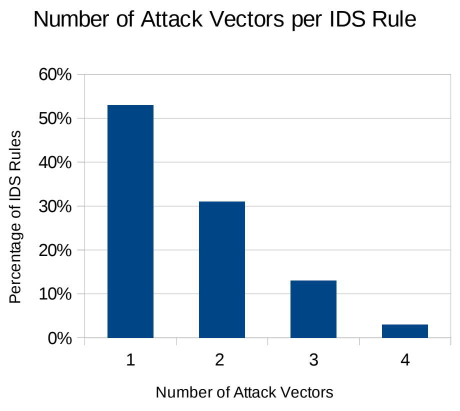


Figure 6.6.1: Number of attack vectors estimated per IDS rule for Shannon octet-entropy.

6.6.1 Number of Attack Vectors

The number of attack vectors per IDS rule for the given experiment is summarised in Figure 6.6.1. For this experiment, 53% (17 rules) have one attack vector, 31% (10 rules) have two clusters identifying attack vectors, 13% (4 rules) have three clusters and 3% (1 rule) have 4 clusters identifying attack vectors. Please note that these numbers are specific to the given experiment. A preliminary experiment at a commercial MSS provider indicates that large-scale operations can expect the distribution to be shifted somewhat towards more attack vectors. It is in other words common that IDS rules may trigger on more than one attack vector, which means that clustering must be used to calculate the entropy of each underlying attack vector.

6.7 Influence by Outliers

Figure 6.7.1 shows the Normal standard deviation σ_1 and Laplacian standard deviation σ_1^L based on the L^1 norm for length corrected normalised Shannon octet-entropy. The Figure shows that the Normal standard deviation σ_1 for some IDS rules indicate a significantly larger privacy leakage than the Laplacian standard deviation σ_1^L. The most extreme cases are SID 119:14 which detects non-standard characters in web requests and SID 1:399 ICMP Host unreachable. The reason for the deviation is in both these cases outliers far out from the main cluster. The Normal standard deviation will give too high weight to the outliers in these cases, since it measures the root of the squared distances. Other IDS rules where the Normal standard deviation of entropy is somewhat influenced by outliers are amongst others SIDs 119:4, 119:15 and 1:1201.

In all these cases, the Laplacian standard deviation will give a more realistic estimate of the privacy leakage than the Normal standard deviation. The Laplacian standard deviation is only significantly larger than the Normal standard deviation for SID 1:402 ICMP Destination Port unreachable. This IDS rule has a left skewed noisy distribution, with several peaks reflecting the servers that were attempted contacted, but did not respond. We interpreted this as one cluster, since the failed services strictly speaking cannot be considered attack vectors. The median for this IDS rule (at 7.5) deviates somewhat from the mean (at 7.2), which gives more weight to the leftmost peaks for the Laplacian standard deviation than the Normal standard deviation does in this case, causing the Laplacian standard deviation to be larger than the Normal standard deviation. This is a pathological case where the normal standard deviation may give a better estimate than the Laplacian standard deviation. However, overall the Laplacian standard deviation σ_1^L should be used to calculate the privacy leakage metric, since this in most cases is the more robust statistic.

6.7.1 Measured Information Leakage

Figure 6.7.1 shows the measured privacy leakage for the experiment using length corrected standard deviation (Normal σ_1 and Laplacian σ_1^L) of normalised Shannon octet-entropy as a function of Snort IDS rule. Further details can be found in Table 6.7.1. This discussion is based on the Laplacian standard deviation, since the previous section shows that the Normal standard deviation has problems with outliers in the dataset. First, it can be observed that the metric works as expected for the extreme cases. The IDS rule that performs random sampling of payload (SID 1:1394000) has the highest privacy leakage. On the other hand, there also exist 5 IDS rules that are very precise at matching the attack vector, and behaves like the perfect model IDS rule R_P with zero privacy leakage. IDS rules that fall into this category are attack vec-

Snort SID	Alarms	K	σ_1	σ_1^L	Description
1:1394000	95096	1	6,71	6,70	Samples random traffic
119:14	3104	1	4,10	3,49	http_inspect non-standard characters in web request
1:402	36224	1	2,34	2,73	ICMP Dest. Port unreachable
1:1201	680	1	1,96	1,77	HTTP 403 Forbidden
119:15	720	1	1,40	1,02	http_inspect over-long URL
1:1394	1384	2	0,90	0,97	Shellcode x86 NOP AAAAAA
119:4	576	1	1,24	0,91	http_inspect preprocessor (IIS decoding attacks)
1:1852	10392	1	0,96	0,75	robots.txt access
1:1463	288	1	0,80	0,72	IRC Chat
119:2	21744	2	0,58	0,61	http_inspect double encoded characters
1:399	631840	1	1,02	0,58	ICMP Host unreachable
119:7	1520	2	0,48	0,43	http_inspect unicode encoded web request
1:12592	312	1	0,33	0,40	SMTP command injection attempt
1:2925	12960	2	0,42	0,35	1x1 GIF attempt (web bug)
1:1560	360	2	0,27	0,30	WEB-MISC /doc access
1:486	368	1	0,37	0,27	ICMP Destination Unreachable
128:4	306616	3	0,25	0,27	spp_ssh
119:18	22760	2	0,32	0,18	http_inspect directory traversal outside web server root.
122:1	576	2	0,08	0,10	sfPortscan preprocessor
122:3	2088	1	0,09	0,09	sfPortscan preprocessor
1:384	566016	4	0,04	0,08	ICMP Ping (general)
1:1437	1056	2	0,08	0,08	MULTIMEDIA Windows Media download
1:408	205904	3	0,04	0,04	ICMP Echo Reply
1:366	202552	1	0,04	0,04	ICMP Ping *NIX
1:368	202552	1	0,04	0,04	ICMP Ping BSD
1:11969	2896	3	0,03	0,03	VOIP-SIP inbound 401 Unauth.
1:385	4392	2	0,04	0,03	ICMP traceroute
1:382	2192	1	0,00	0,00	ICMP Ping Windows (alphabet)
1:2050	32024	1	0,00	0,00	SQL Version Overflow attempt.
1:2003	1777264	1	0,00	0,00	SQL Worm Prop. attempt.
105:2	192	2	0,00	0,00	BO traffic (spp_bo)
106:4	464	3	0,00	0,00	spp_rpc_decode preprocessor

Table 6.7.1: Privacy leakage and number of clusters K measured using length corrected standard deviation based on Shannon octet-entropy for the IDS rules in the experiment.

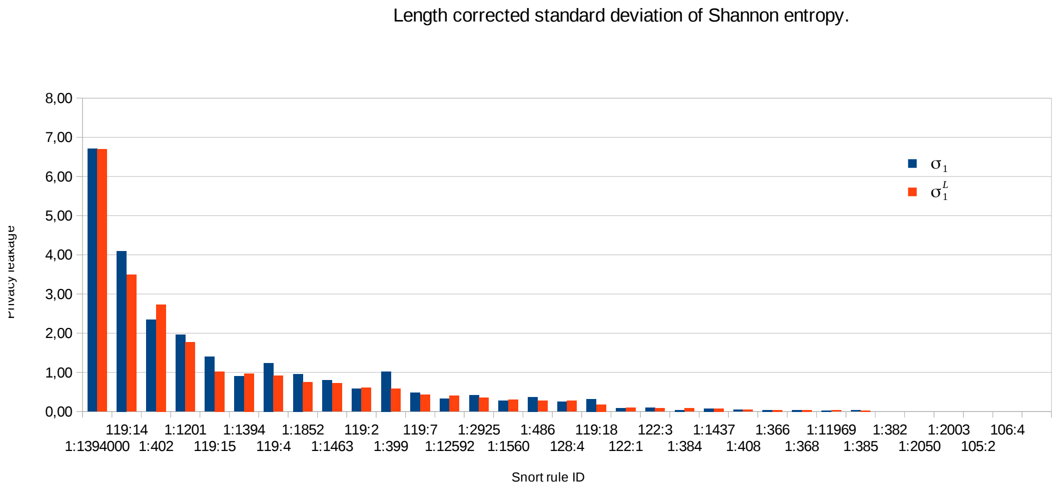


Figure 6.7.1: Privacy leakage measured using length corrected standard deviation (Normal σ_1 and Laplacian σ_1^L) of normalised Shannon octet-entropy as a function of Snort IDS rule.

tors like SID 1:2050 SQL Version Overflow attempt, SID 1:2003 SQL Worm Propagation attempt, SID 105:2 BO traffic and SID 106:4 spp_rpc_decode preprocessor which detect amongst others incomplete RPC segments. All these IDS rules indicate possibly malicious activities, and are precise at detecting the attack. SID 1:382 which detects ICMP Echo requests (Ping) for Windows also behaves like a perfect IDS rule. It typically sends the alphabet in the payload.

There are furthermore 9 additional IDS rules with privacy leakage lower than the threshold of 0.14 for distinguishing between plaintext and encrypted traffic that was identified in Section 6.5.4. Rules in this category can be considered to have insignificant privacy leakage, since it is not distinguishable from encrypted traffic. These include ICMP rules matching ICMP Echo Request and Reply for various platforms (SIDs 1:384, 1:408 and 1:368) and ICMP traceroute (SID 1:385). These ICMP protocols are part of the TCP/IP protocol suite and are benign in themselves, however the Ping protocol is also frequently used for malicious activities like Denial of Service attacks or Ping scans. Furthermore pre-attack activities like portscanning (SIDs 122:1 and 122:3), and unauthorised inbound SIP calls (SID 1:11969) are potentially malicious activities that fall into this category. Last, SID 1:1437 detects download of Windows media files. This would normally be considered a benign activity, and it may also be concerning from a privacy perspective if this

IDS rule is activated, since it could be used to monitor user activities. This rule detects download of Windows media files as two narrow clusters, where the upper cluster at an entropy close to 1 probably indicates download of the compressed media file. This is an example of a pathological case where the entropy standard deviation in itself, as an indirect measure of privacy leakage, does not match the perceived privacy leakage. The data controller may in this case consider whether the privacy impact *I* of this IDS rule should be increased.

The privacy leaking IDS rules can broadly be subdivided into two groups: IDS rules with large privacy leakage ($\sigma_1^L > 1$) and IDS rules with medium privacy leakage ($\sigma_1^L \in < 0.14, 1]$). There are 13 IDS rules with medium privacy leakage. The most privacy leaking of these IDS rules, is SID 1:1394 "SHELLCODE x86 inc ecx NOP" which triggers on any packet that contains a sequence of 31 'A' characters ($\sigma_1^L = 0.97$). The problem is that this sometimes occurs in hex-encoded URLs or hex-encoded data in web pages. It may also occur in non-compressed images, as well as for other protocols. This means that the rule most likely will trigger on a lot of random traffic, which is problematic from a privacy perspective.

Many of the rules with medium privacy leakage may be triggered by normal user behaviour, for example SID 1:486 ICMP Destination Unreachable, SID 1:402 ICMP Destination Port Unreachable and SID 1:399 ICMP Host unreachable. These can be problematic from a privacy perspective, since the ICMP error message often contains the payload of the original request, and these error messages can for example be triggered by high traffic volume (or DoS attacks) towards a server. This means that these ICMP messages essentially sample random user traffic. There are also other IDS rules in this category that will sample random traffic from users, which for example may be used in user profiling. Examples of such rules are SID 1:2925 1x1 GIF attempt that detects web bugs, SID 1:1560 that triggers on access to /doc on the web server root and SIDs 119:2, 119:4, and 119:7 that aim at detecting anomalies in HTTP requests like double encoded requests, IIS decoding attacks and unicode encoded requests. These may indicate attacks, but will in most cases probably be false alarms that essentially sample random user traffic, something that may be problematic from a privacy perspective. 1:1852 robots.txt access, which normally indicates indexing of a web server by a web crawler, also falls into this category.

There are also some other attack rules with medium privacy leakage that do not target ICMP or web traffic. SID 128:4 detects non-SSH traffic on an SSH port, or a protocol mismatch (e.g. SSH1 traffic on an SSH2 port). This rule triggers on the initial key negotiation phase, where some information in the SSH protocol goes in cleartext. This can probably not be considered a significant primary source of privacy leakage, since no sensitive information is transferred in the packets. The data controller may consider reducing the privacy impact for this IDS rule. SID 1:12592 detects SMTP command injection attempts, that aims at exploiting a bug in the ClamAV anti-virus system. The rule definition is a very simple regular expression which is likely to have false alarms. This rule may therefore be concerning from a privacy perspective, although it mostly triggered on spam. Rule 1:1463 triggers on IRC chat traffic, which also may be concerning from a privacy perspective. The reason for implementing this rule, is that IRC bots also often have been used to control botnets of compromised hosts. However, the rule does not check whether the traffic is benign or not.

There are four IDS rules with high privacy leakage, not including the test rule that samples random traffic. Three of these trigger on web traffic: SIDs 119:14, 119:15 and 1:1201. The most privacy leaking ordinary IDS rule (SID 119:14, $\sigma_1^L = 3.49$) triggers on non-standard character encodings in HTTP requests, which are getting increasingly common, especially after IANA allowed non-ASCII domain names. The second most privacy leaking IDS rule is SID 1:402 ICMP Destination Port Unreachable with $\sigma_1^L = 2.73$. This protocol typically copies the failed request in the ICMP message, and therefore samples random traffic requests. On third place is SID 1:1201 HTTP 403 Forbidden ($\sigma_1^L = 1.77$), which also is quite common also for benign traffic, for example on web sites referring to internal material that require subscription. On fourth place is SID 119:15 that tests for over-long URL's ($\sigma_1^L = 1.02$), something that frequently happens for blogs or search engines that use URL referencing. All of these rules may be problematic from a privacy perspective, since they in many cases will trigger on normal user behaviour. It is especially problematic if the IDS rules monitoring web services are set up in an uncritical way, so that these rules trigger for any web server accesses and not only for relevant web servers (e.g. the company's own web servers).

This discussion shows that the privacy leakage metric is able to distinguish between IDS rules that most likely may trigger on ordinary user activities, and therefore may be problematic from a privacy perspective, from the IDS rules that are precise at detecting the underlying attack vector, or that perform a very specific task without leaking any significant amount of data about user behaviour. However there were also two pathological cases where it may make sense to adjust the privacy impact, since using entropy as an indirect measure of privacy leakage not always will gives a true picture of whether the underlying information is sensitive from a privacy/confidentiality perspective or not. Overall, this demonstrates that the privacy leakage metric works as intended. However larger studies involving commercial MSS providers will be needed in the future to confirm these results.

6.7.2 The Effect of Anonymisation

The resulting privacy leakage over all IDS alarms in the experiment, weighted according to number of alarms, is 0.31. However, if the test IDS rule with SID 1:1394000 that samples random data is removed, then the resulting privacy leakage is reduced to 0.16. If all the IDS rules with high privacy leakage are removed, then the resulting leakage is reduced by 0.02 to 0.14.

Surprisingly, it is then more efficient to anonymise all ICMP Destination Host unreachable alarms, since there are many of them (631840) in the data set, and each of them has a significant measured privacy leakage ($\sigma_1^L = 0.58$). Anonymising ICMP Destination Host unreachable alarms would reduce the overall privacy leakage by 0.07 to 0.07. This can probably be done without reducing the usability for the security analyst significantly, since it still would be known which host that was attempted contacted from the IP-address element of the IDS alarm. SID 1:402 ICMP Destination Port unreachable also triggers quite often (32360 times) and has the second highest measured privacy leakage ($\sigma_1^L = 2.73$). Anonymising this rule reduces the privacy leakage by 0.02 to 0.05, and can probably also be done without reducing the possibility to do root cause analysis significantly, since the number of services running on a server normally is limited. Classification based on the EM-clustering can if necessary be used to indicate which server that failed without revealing the original user request. These examples show that the total privacy leakage, calculated as the product of number of IDS alarms N_R for the given rule R and the entropy standard deviation $\sigma^L_{1,R}$, must be used as the optimisation criterion to reduce the overall privacy leakage. The total privacy leakage is calculated as $L_{tot} = N_R \sigma^L_{1,R}$.

Another IDS rule, that either benefits from anonymisation, alternatively by setting the privacy impact to zero, is SID 128:4 which detects ssh anomalies. This rule triggers quite often (306616 times) with $\sigma^L_1 = 0.27$, which means that the overall privacy leakage can be reduced by 0.02 to 0.03 if this rule is anonymised. If the IDS rules with low privacy leakage, that are not relevant from a privacy perspective (all with privacy leakage less than 0.14, except SID 1:1437), are either anonymised or removed by setting the privacy impact to zero, then the resulting privacy leakage index is reduced from 0.03 to 0.011.

If the two IDS rules from the http_inspect preprocessor with largest total leakage (SID 119:14 and 119:2) also are anonymised, then the measured privacy leakage is reduced to 0.005.

This illustrates how a structured method can be used to reduce the privacy leakage of the IDS ruleset based on measured privacy leakage and number of IDS alarms. It is furthermore also clear that many of the IDS rules can be anonymised without significantly reducing the usability for the security analysts. Especially since the clustering model used to identify attack vectors in many cases can be used to help the security analysts in identifying the necessary properties of the underlying data without having to reveal the user payload.

6.8 Related Works

The research area of quantitative information flow based on information theory adds a comprehensive theoretical framework for analysing privacy leakage based on entropies [122, 121]. Our research is based on this, and extends the theory to cover privacy leakage in IDS alarms. There is as far as we are aware of no other research that proposes a comprehensive model of privacy leakage in IDS alarms based on quantitative information flow analysis.

Quantitative information flow analysis that in a similar way uses information entropy has however been proposed used to derive an intrusion detection capability metric in [61]. This metric aims at modelling the uncertainty about the input given the IDS output. The uncertainty as it is termed in this paper is the same as the information leakage defined here based on [122], which in turn is based on the notion of mutual information from [118]. The IDS capability metric is defined as the mutual information between the IDS input and output to the entropy of the input:

$$C_{ID} = \frac{H(X) - H(X|Y)}{H(X)} \tag{6.8.1}$$

The numerator is the same as the information leakage defined in [122], however these data are normalised with respect to the entropy of the input data, something our model does not do. This model assumes that the input data $H(X)$ is the labels (attack or not) from a labelled IDS test set, and the output data $H(X|Y)$ is the classification by the IDS, which also is different from our conceptual model of an IDS rule. It is from this clear that the proposed metric is different from the privacy leakage metric proposed here, since it assumes different input data, a different information model and normalises the indicator to the input data. However an interesting similarity is that the effect of false positives in Figure 3b) in this paper follows a similar falling exponential curve as Figure 6.7.1, as can be expected, since the false alarms here will increase the entropy up to the point where the classifier is not better than random decisions. However this paper does not make the connection to privacy leakage metrics for IDS rules.

There are also some similarities between the proposed approach and the concept of Differential Privacy in statistical databases [40, 41, 42]. Both methods use a Maximum Likelihood (ML) estimate, however the estimate is interpreted differently. Differential Privacy uses the ML estimate to indicate the aggregate value of underlying perturbed data, whereas we use the ML estimate as a measure of underlying attack vectors. Both methods use robust statistics (first norm) for calculating aggregated values. However, Differential Privacy typically adds Laplacian noise to hide individual elements of privacy sensitive information, whereas our privacy leakage metric works in the opposite way - assumed Laplacian noise from an IDS rule is used as an indication of IDS privacy leakage. So although there are similarities, our proposed metric is clearly different to Differential Privacy.

Entropy has previously been proposed as a measure of privacy [29, 17]. Claude Shannon's seminal paper on information theory was the first publication where entropy was proposed to measure the level of ambiguity or equivocation in transferred information [118]. Min-entropy has been proposed as a metric of anonymity that in particular considers local aspects, i.e. the worst case scenario for the user [131]. The more general Rényi entropy has been proposed as a metric of anonymity in [117, 17]. Neither of these have used entropy to measure privacy leakage in IDS alarms.

The chapter is also related to field of privacy-preserving intrusion detection systems [123, 50, 124, 104, 98, 51], however neither of these solutions focus on privacy metrics.

6.9 Conclusion

In this chapter we propose an entropy-based privacy leakage metric founded on quantitative information flow analysis. An advantage is that this metric can be calculated based on already existing information in the IDS alarm database. From a privacy perspective, it provides a structured approach to identify which IDS rules that may be leaking sensitive information and also for handling such privacy leakages.

An advantage with the metric, is that it also is a measure of IDS rule precision. This is clearly desirable, since the objective is to tune the IDS ruleset to reduce the leakage of private or confidential information over time, for example through improving the precision of the IDS rule or by applying anonymisation techniques. This is also an advantage from a security perspective, since more precise IDS rules mean less effort spent on false alarm handling.

We have demonstrated that the proposed approach is feasible based on a set of real IDS alarms. It is furthermore shown that different entropy algorithms and ways to calculate the standard deviation have different strengths and weaknesses. Not surprisingly, the Laplacian standard deviation based on the L^1 norm provides the most robust statistic to avoid problems with outliers, a problem that has been shown to occur in the experimental data. The experiments have shown that Shannon octet-entropy is the best entropy metric with fastest convergence speed for reliably detecting encrypted traffic, and it is also the entropy metric that is is best at distinguishing between plaintext and encrypted traffic. It is also shown how the metric can avoid being incentive incompatible by taking into account the length of the input data.

The Laplacian Mixture Model of the underlying data will in itself be useful for classification purposes. If a given model of the data has been identified, then this can be used for subsequent classification of the underlying samples, for example to anonymise IDS alarms from data clusters that may contain sensitive information about user transactions, or to further classify the attack vectors of the IDS alarms, for example to detect Denial of Service attacks. The clustering can therefore be used as a post-processing step to modify the IDS alarms according to cluster, which means acting as a higher order IDS solution.

A possible attack on the clustering method, is an overfitting attack where a MSS provider decides to shirk by deliberately overfitting the attack vectors. The proposed method to avoid this, is to ensure separation of duties between privacy and security interests and also that third party certification organisations oversee the operation.

The proposed privacy leakage metric only measures the primary privacy leakage sources in IDS alarms. It does not consider secondary sources of information leakage, like correlation of different information sources. However, being able to measure the primary sources of privacy leakage in IDS alarms is at least an initial approach that can and should be considered before more elaborate analyses of the anonymity set are performed. Furthermore, the ability to verify that the anonymisation policies reduce measured information leakages means that policy verification, in the form of a privacy leakage gap analysis, will be possible in order to provide incremental reductions of privacy leakage in IDS alarms over time.

6.10 Future Work

Future work includes doing comparative studies of the performance of different MSS providers from a privacy perspective. Adapting the privacy leakage metric to support anomaly-based IDS is also left as future work. This will amongst others require subdivision of the alarms, for example based on service etc., to avoid that the entropy space becomes too crowded by attack vectors.

Investigating possible secondary privacy leakages that may occur due to inference or cross correlation between different information sources both within the IDS alarm and outside is also left as future research. This would require taking the privacy leakage metrics and evaluation even further in order to evaluate the anonymity set that can be expected for private or sensitive information, using metrics like differential privacy [40, 41, 42], k-anonymity [37], or l-diversity [86].

Chapter 7

Metrics-supported Privacy Enforcement

The dissertation has so far pursued two main goals: Part II covers how to enforce a privacy policy for IDS alarms, and the previous Chapter investigates how to identify and measure leakage of potentially private or confidential information. What is missing, is to show how these two objectives can be combined into a comprehensive solution that allows for implementing privacy-enhanced intrusion detection systems in practice.

This chapter therefore proposes a methodology and tools for metrics-supported privacy enforcement, to measure, identify and reduce the leakage of private or confidential information for a data service. This includes describing a structured approach for identifying how much, where and what information that leaks through a service. One technique developed here, is a Shannon entropy based privacy leakage map that can be used to identify where information leakages occur for different attack vector clusters. The chapter also proposes how a metrics-supported privacy enforcement scheme based on Expectation-Maximisation based clustering of data entropies can be performed under conditions that are verifiable. This can be used as part of a privacy gap analysis to plan, enforce, validate and improve the privacy policy over time.

7.1 Introduction

A privacy policy can be enforced by applying techniques like anonymisation, pseudonymisation or encryption, where private or confidential information in the two latter cases only should be made available to authorised sources, for example doctors at a health institution, plant engineers in a critical infrastructure, security analysts at a managed security service, Computer Emergency Response Team (CERT) or law enforcement during attack investigation.

However a general challenge is knowing what to anonymise, and how much information that is required to efficiently do a given task. This is an area where privacy leakage metrics are useful to support the privacy enforcement policy. Such metrics can for example be used as part of a privacy impact assessment to verify that a privacy policy is working as expected, and trigger revalidation of the privacy policy if a statistical model of the underlying traffic is no longer supported.

Privacy leakage metrics can be used to quantify *how much* private or confidential information that is leaking, as well as detecting *where* such information leakages occur. This can subsequently be used to improve the privacy policy in a structured way, to reduce the overall privacy impact of the service being protected by the anonymiser. The metrics are assumed used together with a privacy enforcement mechanism that allows for ensuring transparency on who have accessed what, in addition to non-repudiation, so that an operator cannot deny having processed certain information. This, together with techniques like key sharing, makes it possible to implement a controlled environment where it will be harder for trusted insiders to abuse the information being processed, or maliciously modify the policies being enforced.

The main contributions of this chapter are:

1. developing the concept of a privacy leakage map, which indicates *where* information leakages occur, and investigating how this can be used to support enforcement of a privacy policy;

2. an analysis and recommendations on how privacy leakage metrics should be used as part of a privacy policy enforcement and validation scheme;

3. describing how the metrics can be used as part of an improvement process to maintain and improve the privacy protection over time;

4. illustrating how the privacy leakage metric and enforcement mechanism can be used for other application areas than privacy-enhanced intrusion detection systems.

The chapter illustrates that there are both privacy and security benefits from the proposed scheme, as well as quite broad application areas. The objective of this chapter is not to describe the method for enforcing privacy by rewriting data but how to combine privacy enforcement techniques (e.g anonymisation, pseudonymisation or encryption) with privacy metrics as part of an information security management process. This can be used to reduce the gap between actual and desired privacy over time.

The remainder of this chapter is organised as follows: Section 7.2 describes the problem that the chapter aims at solving, and gives some motivating examples on how the privacy leakage metrics can be used to support a privacy and security improvement process. This includes giving an introduction to the privacy enforcement scheme that the metrics will be used with. Section 7.3 gives an introduction to the theory behind the privacy leakage metric, which is used as part of the improvement process. Section 7.4 describes how the privacy leakage metric can be extended to a privacy leakage map showing where information leaks in data elements. Section 7.5 analyses how privacy leakage can be detected and mitigated based on case studies. Section 7.6 discusses different anonymisation strategies, and how the privacy leakage metrics can support these. Section 7.7 discusses related works, Section 7.8 concludes the chapter and Section 7.9 outlines future work. Technical details on how metrics-based privacy enforcement scheme has been implemented in the XACML-based reversible anonymiser is described in appendix A.

7.2 Problem Description and Motivation

The problem that this chapter aims at solving, is how to use and extend the Shannon entropy-based privacy leakage metric, described in the previous chapter, to support an improvement process consisting of planning, development, enforcement and verification of privacy policies, where these privacy policies control a reversible anonymisation scheme for XML data. Reversible anonymisation here means that XACML policies can control anonymisation

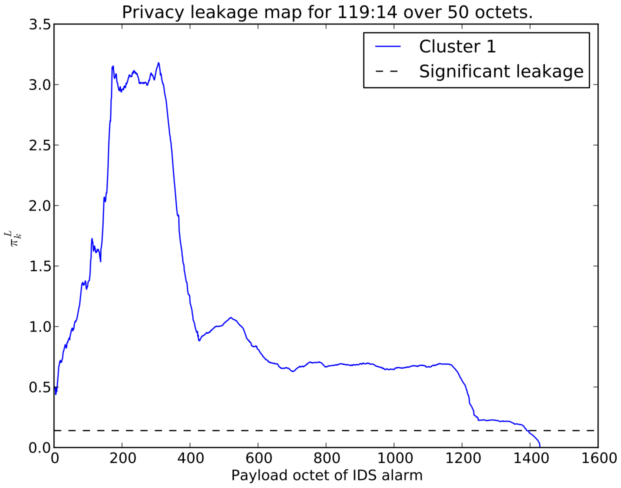


Figure 7.2.1: SID 119:14 Non-standard characters in web requests.

of information down to octet ranges of selected XML elements or attributes, at the same time as the anonymised information can be stored encrypted in a format that specifies how to undo the anonymisation.

The chapter also describes how the privacy leakage metrics and privacy enforcement mechanism can be used in an information security management process to identify gaps between the stated and actual privacy level, as well detecting privacy erosion over time, as the underlying data changes. The chapter furthermore proposes how a detailed privacy leakage map can be calculated based on the standard deviation of Shannon entropy, in order to visualise *where* (in which octets of a data element) information leakages occur. This information can then be used to set up more detailed anonymisation policies that are able to anonymise significant privacy leakages.

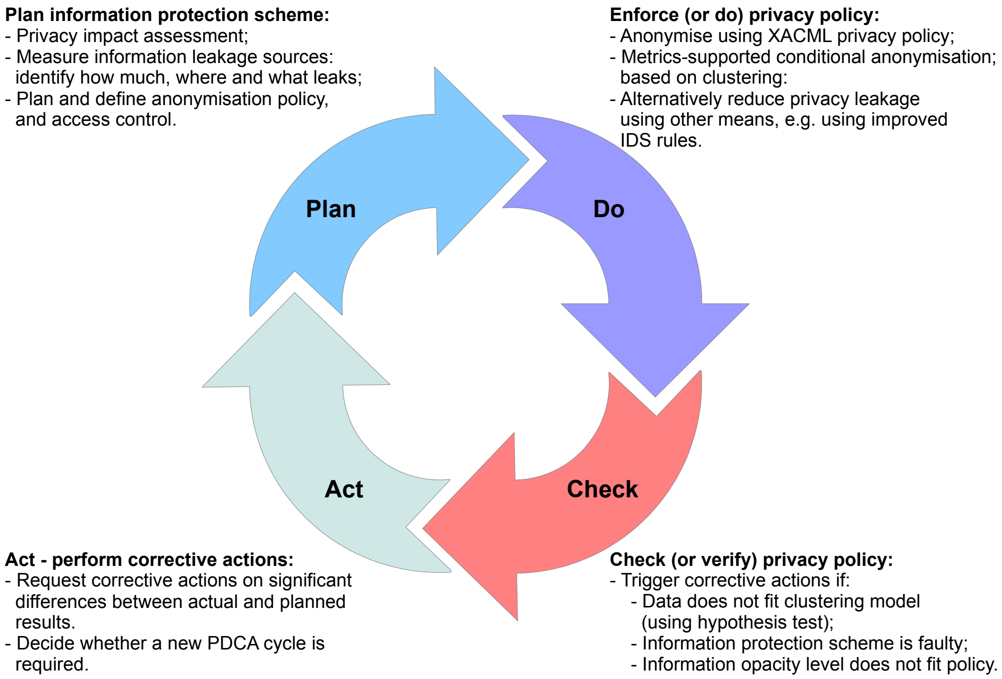
7.2.1 Motivation

Figure 7.2.1 shows a motivating example describing some of what we aim at achieving. This figure illustrates how the Shannon entropy standard deviation varies as a function of payload octet number for the Snort IDS rule with SID 119:14, which triggers an IDS alarm on non-standard characters in web requests (3104 alarms). Figure 7.2.1 illustrates *where* in the IDS alarm information leakages occur. *How much* privacy leakage the IDS rule has can be quantified using Shannon-entropy based privacy leakage metric π^L described in the previous chapter. The privacy leakage map has a stapled line indicating that traffic below this line is considered having insignificant privacy leakage for example being anonymised or encrypted. The figure therefore indicates a significant information leakage ($\pi^L > 0.14$) for payload octets in the range $[0, 1400]$ with a large information leakage peak measuring privacy leakage $\pi^L > 3$ in the octet range $[170, 330]$.

The rule triggers mostly on traffic from Facebook games in the network being monitored, since the games use non-standard characters in the web request when updating game status etc. It also triggers on web requests verifying the status of digital certificates using the Online Certificate Status Protocol (OCSP), which also is related to these games.

This IDS rule has a significant privacy impact since it triggers on benign use of a popular web application like Facebook. It may furthermore reveal privacy sensitive information about user behaviour, user preferences, interests etc., which means that payload samples in alarms from this IDS rule should be anonymised. One technique that may be used, is to anonymise traffic using privacy *blacklisting* based on IP address, so that traffic from Facebook.com and affiliated gaming companies will be anonymised. Privacy blacklisting here means that certain information of private or confidential sources need to be protected against unauthorised disclosure by insiders.

Anonymisation of private or confidential information, for example in IDS alarms or XML messages in web services, is needed to protect such sensitive information from eavesdropping or abuse, both from curious insiders, but also from outsourced services. This is especially the case where services, for example managed security services, are outsourced to third-parties that may not be entrusted to see all information in the messages. Privacy leakage metrics are then required to verify that privacy policy enforcement works as intended, in order to be able to detect and restrict unintended flows of sensitive data. Which information that needs to be restricted will typically come from privacy requirements derived from a privacy impact assessment done by the data controller.



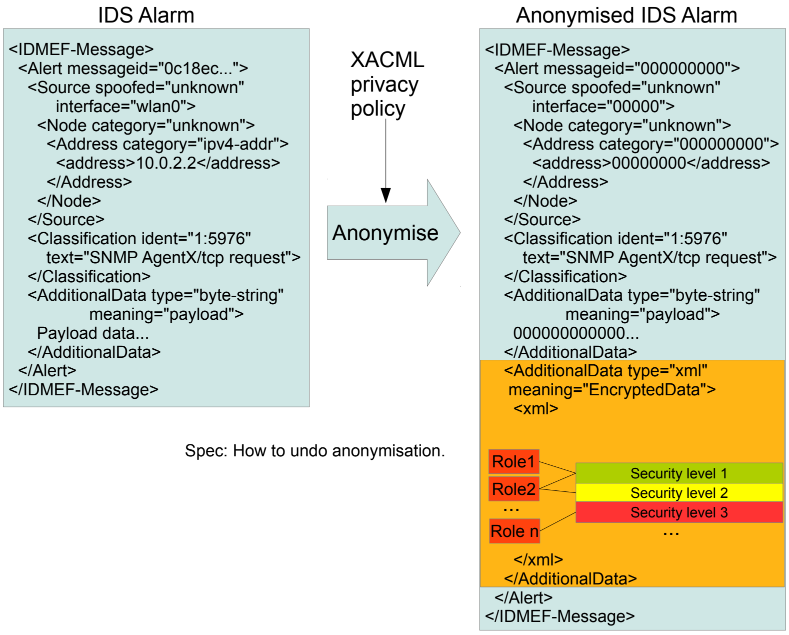
Plan information protection scheme:
- Privacy impact assessment;
- Measure information leakage sources: identify how much, where and what leaks;
- Plan and define anonymisation policy, and access control.

Enforce (or do) privacy policy:
- Anonymise using XACML privacy policy;
- Metrics-supported conditional anonymisation; based on clustering:
- Alternatively reduce privacy leakage using other means, e.g. using improved IDS rules.

Act - perform corrective actions:
- Request corrective actions on significant differences between actual and planned results.
- Decide whether a new PDCA cycle is required.

Check (or verify) privacy policy:
- Trigger corrective actions if:
  - Data does not fit clustering model (using hypothesis test);
  - Information protection scheme is faulty;
  - Information opacity level does not fit policy.

Figure 7.2.2: Privacy Improvement Process using well-known Plan Do Check Act method.



IDS Alarm

```
<IDMEF-Message>
  <Alert messageid="0c18ec...">
    <Source spoofed="unknown"
          interface="wlan0">
      <Node category="unknown">
        <Address category="ipv4-addr">
          <address>10.0.2.2</address>
        </Address>
      </Node>
    </Source>
    <Classification ident="1:5976"
        text="SNMP AgentX/tcp request">
    </Classification>
    <AdditionalData type="byte-string"
              meaning="payload">
      Payload data...
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Anonymised IDS Alarm

```
<IDMEF-Message>
  <Alert messageid="000000000">
    <Source spoofed="unknown"
          interface="00000">
      <Node category="unknown">
        <Address category="000000000">
          <address>00000000</address>
        </Address>
      </Node>
    </Source>
    <Classification ident="1:5976"
        text="SNMP AgentX/tcp request">
    </Classification>
    <AdditionalData type="byte-string"
              meaning="payload">
      000000000000...
    </AdditionalData>
    <AdditionalData type="xml"
     meaning="EncryptedData">
      <xml>

      </xml>
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Figure 7.2.3: Outline of how the reversible anonymisation scheme modifies the data of IDMEF messages.

7.2.2 Introduction to the Privacy Enforcement Scheme

This section gives an overview over how the privacy policy enforcement scheme assumed in this chapter works. Anonymisation enforcement is done using a reversible anonymisation scheme, which is based on and extends the eXtensible Access Control Markup Language (XACML) based decision cache and anonymiser in [135]. Reversible anonymisation means that the anonymised IDS alarm contains an encrypted specification on how to undo the anonymisation, as illustrated in figure 7.2.3. This also means that the enforcement mechanism considers anonymisation as a specialisation of authorisation to information. The scheme supports both traditional irreversible anonymisation and reversible anonymisation. The enforcement mechanism can therefore be used for authorisation, anonymisation or encryption of information in a service oriented architecture.

The figure illustrates anonymisation of an XML-based IDS alarm in the Intrusion Detection Message Exchange Format (IDMEF) [62]. An XACML based privacy policy defines XPath expressions [127], which are used to identify XML resources and attributes that need to be anonymised or permitted. Anonymisation can be performed using either a default PERMIT policy, where private or confidential information explicitly must be anonymised, or using a default DENY policy, where all information by default is anonymised, and selected information that according to a privacy impact assessment has been shown to not be privacy leaking subsequently is permitted. The latter method is in general preferred from a privacy perspective, since it ensures privacy by default, according to the Privacy by Design criteria [25]. The anonymisation protocol uses the IDMEF data extension scheme, which ensures that the anonymiser is compatible with existing Security Information and Event Management systems based on IDMEF, for example PreludeIDS[1]. Furthermore, only authorised stakeholders, possessing the correct private encryption keys, are able to deanonymise information in security levels they are authorised for.

The reversible anonymisation scheme allows several roles or users access to different security levels according to the XACML privacy policy. This essentially means that the scheme supports multilevel security for private or

[1] PreludeIDS: https://www.prelude-ids.org

confidential information in XML messages. It also allows separation of duty constraints to be specified, for example to support key sharing, where several stakeholders need to collaborate to reveal given information (for example the data controller and law enforcement).

The scheme furthermore opens up for transparency on who have seen what information, by logging who have been authorised to see which IDS alarms. Note that the reversible anonymisation scheme is not limited to IDMEF messages, but can with small modifications be adapted to work for other XML-based formats. The only adaptations needed, is to declare where the encrypted specification of anonymisation reversal resides. This makes it possible to perform fine-grained anonymisation of information in any XML message down to octet ranges of individual elements or attributes of a message. This means that the privacy enforcement scheme will be useful for service oriented architectures in general, and in particular for sharing of best practices and attack information when sensitive information needs to be confidential.

XACML was chosen because it is an authorisation language with wide adoption in industry, that works well with other XML-based authentication standards like the Security Assertion Markup Language (SAML) [110] in a service oriented architecture. The solution uses the GeoXACML extension [6], which supports location-aware authorisation where geographical locations can be embedded within the policies, as well as providing rich data types that we use in the reversible anonymisation protocol, amongst others for defining octet ranges and clustering models. An advantage with this approach is that anonymisation and protection of sensitive information is policy-based, configurable and adaptable, instead of being hard-coded.

The Anonymiser and Deanonymiser components will be added as components of an Enterprise Service Bus (ESB) based architecture for protecting critical infrastructures against cyber-attacks. This is being developed as part of the European PRECYSE project[2].

[2] PRECYSE stands for Prevention and Reaction to Cyber Attacks to Critical Infrastructures, http://www.precyse.eu.

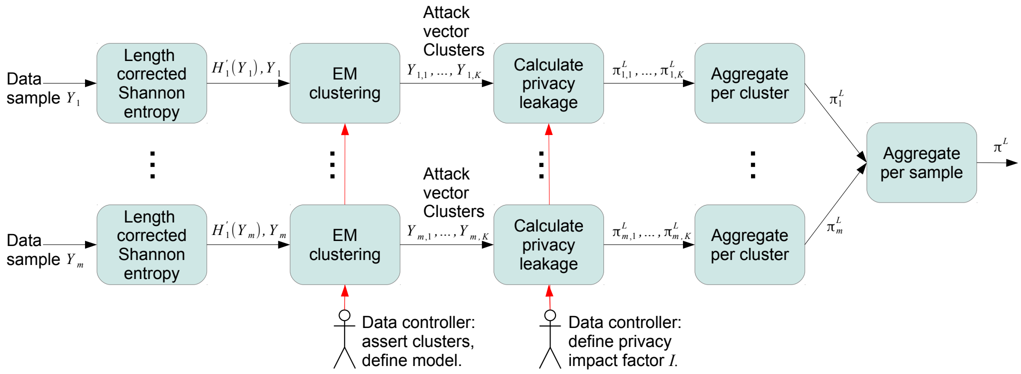

Data sample Y_1
Length corrected Shannon entropy
$H'_1(Y_1), Y_1$
EM clustering
Attack vector Clusters
$Y_{1,1}, \dots, Y_{1,K}$
Calculate privacy leakage
$\pi^L_{1,1}, \dots, \pi^L_{1,K}$
Aggregate per cluster
π^L_1
Aggregate per sample
π^L
Data sample Y_m
Length corrected Shannon entropy
$H'_1(Y_m), Y_m$
EM clustering
Attack vector Clusters
$Y_{m,1}, \dots, Y_{m,K}$
Calculate privacy leakage
$\pi^L_{m,1}, \dots, \pi^L_{m,K}$
Aggregate per cluster
π^L_m
Data controller: assert clusters, define model.
Data controller: define privacy impact factor I.

Figure 7.3.1: Overview over how the privacy leakage metric is calculated.

7.3 Background for Entropy-based Privacy Leakage Metric

This section gives a brief overview over the privacy leakage metric defined in chapter 6. The metric is based on Shannon entropy and the concept of mutual entropy [118], and has been developed based on the theory quantitative information flow analysis in [122, 121]. Figure 7.3.1 gives an overview picture over how the privacy leakage metric is calculated.

Figure 7.3.1 shows that the privacy leakage metric starts with calculating the length-corrected Shannon entropy for a set of data samples. An Expectation-Maximisation based clustering algorithm is then used to derive a set of attack vector clusters from the data sample, for example the underlying attack vectors that an IDS rule triggers on. The clustering is a semi-automatic process, where a data controller can oversee the process, and assert missing clusters or delete clusters where the algorithm overfits the data. The data controller can also investigate the data of the clusters, to determine if a data cluster warrants being defined as a separate attack vector or not. The objective is to define a clustering model that is as simple as possible, but still captures important features of the dataset, to avoid overfitting the model.

The privacy leakage metric, based on the Laplacian standard deviation of Shannon entropy, is subsequently calculated for each cluster. The data controller can in this phase investigate each data cluster, and define a privacy impact factor in order to define the privacy relevance of the given data cluster. The privacy leakage measurements are then aggregated over all clusters for a

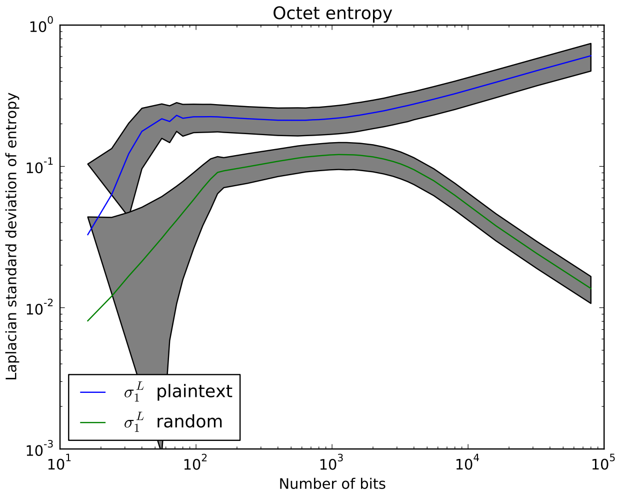


Figure 7.3.2: Laplacian standard deviation of length-corrected Shannon entropy with 95% confidence bands as a function of payload length for plaintext and random data, based on Monte-Carlo simulations.

given IDS rule or event, and can finally be aggregated over all IDS rules or events in the dataset, to provide an overall privacy leakage indicator for the given service.

This makes it possible to calculate a privacy leakage index for a set of IDS rules. The overall privacy leakage risk can be estimated by multiplying the privacy leakage metric with the estimated or measured annual occurrence frequency f of the given IDS alarm or message, i.e: $risk = f \cdot \pi^L$.

It can be observed that the privacy leakage metric will have zero or close to zero entropy in the following three cases:

1. anonymised data;
2. encrypted data, as the length of the data increases;
3. an attack vector cluster that does not change.

It can be observed that an anonymised IDS alarm using the reversible anonymisation scheme normally will have zero or close to zero entropy with zero standard deviation for anonymised document elements in IDS alarms. Furthermore, sensitive data is stored encrypted in the privacy enforcement scheme. Such encrypted data will go asymptotically towards zero measured privacy leakage as a function of payload length as illustrated in figure 7.3.2. This means that π_R^L is a useful metric to verify correct enforcement of a privacy policy, since one can expect lower measured privacy leakage from an improved privacy enforcement scheme.

The privacy leakage metric will in theory work for any type of Intrusion Detection System. The metric work best for signature-based IDS, since the IDS rule signature ID then can be used to subdivide the entropy space, to avoid that one IDS rule matches too many attack vectors, which would make the mixtures of entropies too crowded to be able to distinguish them.

We expect that this limitation can be overcome in order to support anomaly-based IDS. One approach may be to combine clustering of different data elements of the IDS alarm with a decision tree based approach for selecting a subset of the samples. This would limit the number of clusters for elements where the distribution function is too crowded by underlying mixture components, however this is left as future work.

It must be noted that the privacy leakage metric and approach is not limited to measuring privacy leakage in IDS alarms. The metric may also be useful in other cases, for example as part of planning and verifying an information protection or anonymisation scheme for any service, not only IDS based services. The privacy leakage metric is useful for verifying correct implementation and operation of an anonymisation scheme according to a privacy policy. Another example of abnormal data that the approach may be able to successfully identify and classify, is clusters representing buffer overflow attacks. If the attack vector does not contain self-mutating code, then such attacks would stand out as peaks or narrow clusters with little or no variance. This approach also works well for identifying Denial of Service attacks, which often will appear as a peak with no or little variation[3]. The metric is in particular useful if only part of some data is being anonymised, since the entropy standard deviation then can be expected to decrease if the anonymisation scheme is operating correctly, whereas a metric using absolute entropy in principle may take any value, due to the influence of non-anonymised parts of the data.

[3] This is an example of a win-win situation where security also benefits from privacy enhancing technologies, as emphasised by the Privacy by Design principles [25].

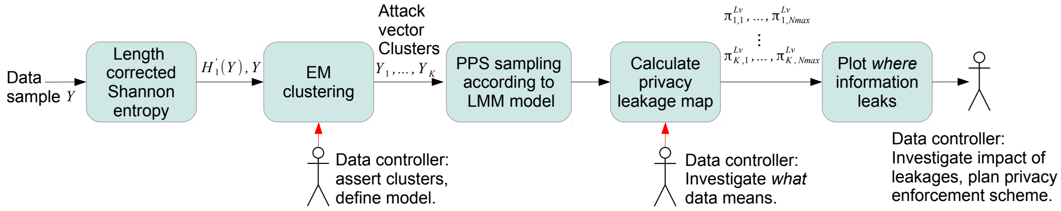

Data sample Y
Length corrected Shannon entropy
$H_1^{'}(Y), Y$
EM clustering
Attack vector Clusters
$Y_1, \ldots, Y_K$
PPS sampling according to LMM model
Calculate privacy leakage map
$\pi_{1,1}^{Lv}, \ldots, \pi_{1,Nmax}^{Lv}$
$\vdots$
$\pi_{K,1}^{Lv}, \ldots, \pi_{K,Nmax}^{Lv}$
Plot *where* information leaks
Data controller: assert clusters, define model.
Data controller: Investigate *what* data means.
Data controller: Investigate impact of leakages, plan privacy enforcement scheme.

Figure 7.4.1: Block diagram showing how the privacy leakage map is being calculated.

7.4 Privacy Leakage Map

This section describes how to extend the privacy leakage metric from measuring one entropy metric per element or attribute of the IDS alarm to a more fine-gained *entropy map* that indicates privacy leakage almost down to individual octet level. This can then be used to identify *hotspots* - octets or protocol parameters where the information leakage is large.

7.4.1 How the Privacy Leakage Map is Calculated

Figure 7.4.1 shows a block diagram of how the privacy leakage map is calculated. The privacy leakage map takes advantage of the classification into attack vectors done by the EM-based clustering of the privacy leakage metric π_R^L, so that the entropy map is calculated for each cluster in the LMM, as shown in figure 7.4.1. This means that the initial two steps: calculating the length-corrected Shannon entropy and EM-clustering, is the same as when calculating π_R^L.

The well known technique of probability proportional to size (PPS) sampling is used to divide the samples between the clusters when calculating the privacy leakage map, in order to get a representative sample of the distribution in each cluster. Probability proportional to size sampling means that a random number $r \leftarrow rnd()$ will be drawn for each sample y_i. This random number is then used to decide proportionally which cluster k the sample belongs to according to the weights $w_{i,k}$ in the Laplacian Mixture Model, as shown below:

$$k = \min_{k \in \{1,\ldots,K\}} \left(k \,\middle|\, r \leq \sum_{j=1}^{k} w_{i,j} \right). \tag{7.4.1}$$

A more detailed entropy map can then be created by calculating the privacy leakage metric over an octet range which essentially cuts out a *slice* of each sample y_i from a set of data samples Y. This is expressed using the notation $y_i[x : x+W]$. This means that the privacy leakage is calculated over all octets from octet $y_{i,x}$ and to octet $y_{i,x+W}$ where x is the starting point of the octet range, and W is the window size the privacy leakage is calculated over. A list slicing operator , which is common in several programming languages, can then be defined as:

$$y_i[x : x+W] = \left\{o_{i,j} | j \in \{x, x+1, ..., x+W\}, j \leq N_i\right\} \tag{7.4.2}$$

This means that the privacy leakage for an octet range, denoted by $\pi_k^L[x_1 : x_2]$, can be calculated using Laplacian standard deviation as:

$$\pi_k^L[x_1 : x_2] = I \cdot \sqrt{2} \sum_{i=1}^{m} \left| H_1^{'}(y_i[x_1 : x_2]) - \overline{H_1^{'}(Y[x_1 : x_2])} \right|. \tag{7.4.3}$$

The length-corrected Shannon entropy $H_1^{'}(y_i)$ should be used when calculating the detailed privacy leakage map. The reason for this is that the length corrected Shannon entropy is nearly constant (difference 0.015) for plaintext data in the interval $[5, 100]$ octets as shown in Figure 7.3.2. This means that the scale of privacy leakage measurements for practical purposes can be considered independent of window size W in this range.

The privacy leakage map can then be visualised by plotting $\pi_k^L[x : x+W]$ for $x \in \{0, 1, ..., Nmax - W\}$ for all clusters k, as illustrated in figure 7.2.1. The minimum window size W is 5 octets for a sample of minimum 50 messages, to ensure a reasonably tight confidence interval, based on the Monte-Carlo simulations in the previous chapter. A small W gives larger precision, but also more noise. A too large window smooths the privacy leakage function too much, and causes slow reaction to changes. We used $W = 50$ octets by default as a compromise when plotting the privacy leakage map from the payload excerpt in IDS alarms. However IDS rules trigging on very short payload may need smaller window sizes. The plots also illustrate the threshold for what is considered a significant information leakage ($\pi^L = 0.14$) which was identified during the Monte-Carlo simulations. Significant here means that the information leakage is distinguishable from random traffic within a 95% confidence interval.

7.5 Case Studies

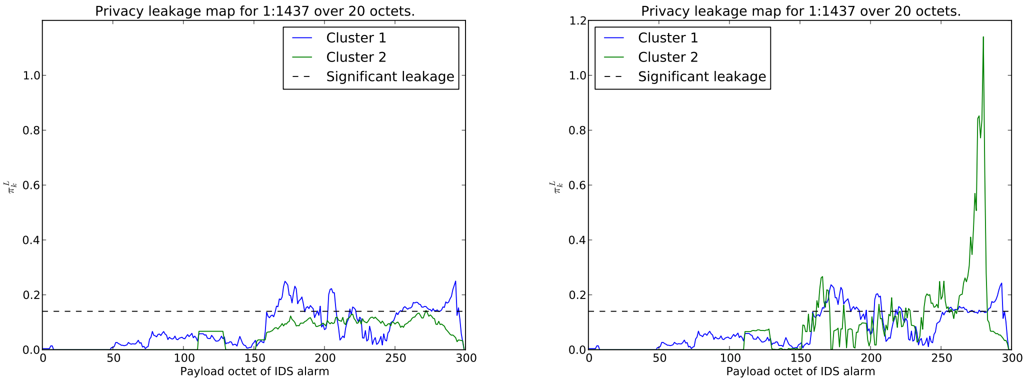
The objective in this section is to investigate the behaviour of the privacy leakage map, including discussing how privacy leakage metrics can be used to support a continuous improvement process, as well as enforcement and validation of a privacy policy. The experimental results are based on 557871 IDS alarms from my own home network between 2009 and 2011, detected using Snort with the publicly available VRT ruleset. 23% of the IDS alarms are from the KDD-Cup'99 data set.

This is a limited data set that will not reflect the privacy leakage measured at a professional MSS provider doing large-scale measurements. This is the same dataset that was used in the previous chapter, and the same limitations as described in section 6.6 therefore apply.

The data set should be sufficient to illustrate that the privacy leakage map works as intended, since the examples clearly illustrate that the metric can be used to identify where entropy sources causing privacy leakages are in the analysed IDS alarms. The next subsections discuss some cases that illustrate how privacy leakage metrics can be used as part of a privacy enforcement scheme. The privacy policy defined in the following examples depends on the IDS alarms triggered by the traffic in a given network, which will differ between networks. This means that a separate privacy impact analysis will be needed for other datasets, and also over time to handle privacy erosion due to changes in the underlying data, introduction of new attack vectors etc. The outcome of this analysis is a set of metrics-supported techniques that can be used to implement a privacy enforcement mechanism that mitigates significant privacy leakages.

7.5.1 Clustering and Octet-Range Based Anonymisation

This case study illustrates a privacy impact analysis of the Snort IDS rule with rule identity (SID) 1:1437, which indicates download of Windows multimedia files. Figure 7.5.1a shows that two clusters are identified for this IDS



(a) Privacy leakage map for gzip compressed data. (b) Privacy leakage map for decompressed data.

Figure 7.5.1: SID 1:1437 Windows Multimedia download before and after decompressing the HTTP header.

rule - cluster 1 in has 560 samples and cluster 2 has 496 samples. Cluster 1 (568 alarms) is in cleartext/XML, and matches the HTTP response and the XML container part of Microsoft's Advanced Streaming Format (x-ms-asf). The payload does not change much between instances for this cluster, which gives a low privacy leakage measurement due to low variance in entropy between samples.

Cluster 2 (488 alarms) contains the start of a gzip compressed data stream. The gzip compressed stream can easily be decompressed since the compression header for these IDS alarms is known. Decompressing it reveals that it mostly contains references to download of advertisements from doubleclick.net. The traffic it triggers on in this experiment is mostly secondary traffic (advertisements etc) caused by web surfing. The payload does not reveal the primary information sources, i.e. which web pages that are being visited. This IDS rule may however still leak a significant amount of privacy sensitive information, for example information about a given user's personal preferences which may be revealed from targeted marketing in the advertisements [26].

This illustrates a pathological example where the privacy leakage metric may give a wrong measure of privacy leakage, since compressed information has similar properties as encrypted traffic, i.e. both have high absolute entropy and low entropy standard deviation. However it is relatively easy to decompress this information. This means that that the algorithm calculating the privacy leakage metric should attempt to decompress information before calculating the entropy values where possible - i.e. if the compression algorithm is known, and the payload excerpt in the IDS alarms contain the start of the compression stream which includes the compression dictionary [137].

Recalculating the privacy leakage measurements with decompression of gzip data switched on, shows that the measured privacy leakage is somewhat larger for decompressed data ($\pi^L = 0.37$), however the detailed entropy map in figure 7.5.1b shows that there is a peak of large information leakage ($\pi_2^L[260:283] = 0.67$), which are the URLs to Doubleclick advertisements.

The EM-based mixture model described in the previous chapter can in this case be used to classify new IDS alarms based on entropy, for example to anonymise targeted marketing in the IDS alarms in cluster 2.

A possible anonymisation scheme for this IDS rule might be to use a threshold-based scheme, which anonymises any octets with entropy standard deviation larger than 0.01. This would block out all varying parts of the IDS alarms with significant variance. However, a problem with such a naive threshold-based strategy, is that if an external attacker adds entropy to the data (e.g. by modulating parameters that can be changed in the payload while still being able to match the given IDS rule), then the attack would also risk being anonymised which is not desirable from a security perspective.

It is therefore safer and better to anonymise *given data ranges* for a given attack vector than to base the anonymisation strategy on thresholds in measured privacy leakage. A possible strategy for this IDS rule is to anonymise octet 260 and 283 of the payload in IDS alarms matching cluster 2 for decompressed data, to anonymise the Doubleclick advertisements and avoid leaking information about personal preferences. Such an anonymisation strategy can be expected to be reasonably robust due to the small overlap between the clusters, as long as the clustering model is being monitored to detect that the model is being supported by the data over time. This strategy means that the enforcement mechanism must be able to decompress compressed data before calculating the length-corrected Shannon entropy.

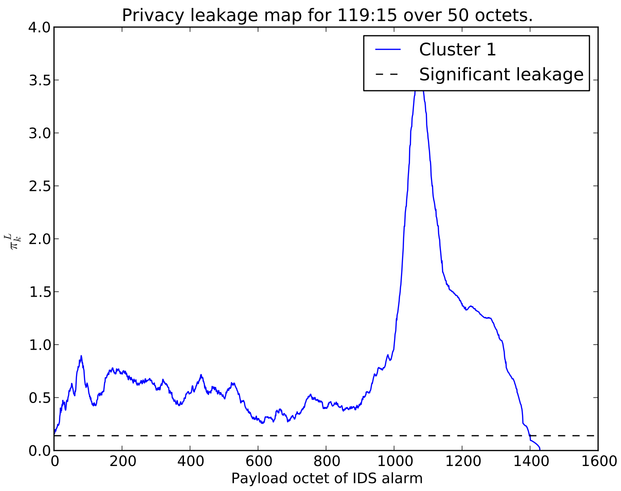


Figure 7.5.2: SID 119:15 http_inspect over-long URL

7.5.2 Unconditional Reversible Anonymisation

Figure 7.5.2 shows the detailed privacy leakage map for SID 119:15 (616 samples), which triggers on over-long URLs. The historic reason for this IDS rule is a buffer overflow exploit that could occur in very old versions of Microsoft's IIS browser. Nowadays long URLs are quite common, for example if a web site refers and URL to another site. The privacy leakage for this rule is $\pi^L = 1.02$, and it can be observed that the first 1000 octets, which contain the referrer part of the URL, has relatively low privacy leakage, whereas the URL or parameters being referred to (>1000 octets) has larger information leakage. If this rule is enabled, then the entire payload should be anonymised, to avoid unnecessarily revealing information about user behaviour. Authorised stakeholders can however still be given access with the reversible anonymisation enforcement scheme, however access to such information should be logged.

7.5.3 Pattern Matching/Data Mining Based Anonymisation

The privacy leakage map for the IDS rule with SID 1:402 (36224 alarms), which triggers on ICMP Destination Port unreachable, is shown in figure 7.5.3. The measured privacy leakage is relatively large ($\pi = 2.73$). This IDS rule has significant information leakage in the privacy leakage map between 0 and 350 octets, where it drops to nearly zero.

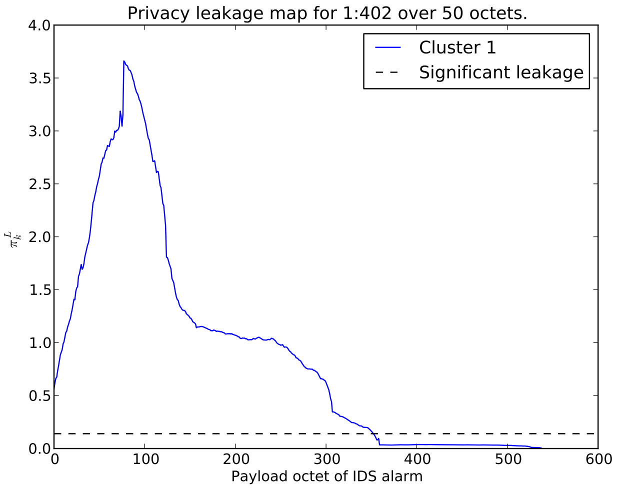


Figure 7.5.3: SID 1:402 ICMP Destination Port unreachable.

The ICMP Destination Port unreachable protocol copies the request that failed as part of the payload. This means that the rule essentially samples random attempted user sessions, which may be considered a significant privacy problem. The entropy distribution is quite noisy, and there are no clear clusters that can be identified. Part of the information seems to be Netbios TCP/IP protocol statistics requests from *nbtstat*. This could be the result of malware probing activities against our network. The rule triggers on a mix of benign and suspicious traffic. Some domains are according to the Web Of Trust [4] benign (e.g. update.microsoft.com), others are suspicious (e.g. spreading spam, viruses, Trojans etc.). The rule seems to trigger on a significant amount of traffic with malicious origin, so it is clearly useful from a security perspective. It can be used to identify suspicious malware activities, and to identify problems with own services. The privacy impact of IDS alarms from this rule is for our data set limited to revealing information in failing SIP messages, which identifies users. The utility from a security perspective is probably high, so the payload from this rule should be available to security analysts.

A privacy policy may want to test for, and anonymise specific messages (e.g. the SIP messages) from this IDS rule, for example by testing for given patterns in the payload of the IDS alarms using regular expressions and anonymise these.

[4] http://www.mywot.com

7.6 Anonymisation Strategies

The previous examples show that the privacy leakage metric π^L is useful to indicate how much private or confidential information an IDS rule is leaking. The EM-based clustering algorithm can furthermore be used to identify what the underlying attack vectors of an IDS rule represents. It can also be used to detect changes caused by introduction of new attack vector variants detected by the IDS rule, which would show up as new emerging clusters.

The privacy leakage map gives an overview over where the underlying entropy sources are in each cluster. It may also be useful as a unique signature of a given attack vector, for example as part of alarm correlation analysis to identify different attack vectors recognised by the IDS rule. Based on the case studies, there are several different anonymisation strategies that may be considered. Some of the possible strategies are directly related to the privacy leakage metrics:

- Cluster-based anonymisation;
- Privacy leakage threshold based anonymisation based on π^L metric;
- Privacy leakage map threshold based anonymisation.

Other anonymisation strategies may indirectly benefit from the privacy leakage metrics:

- Anonymisation of octet ranges within IDS alarms of given clusters/attack vectors;
- Anonymisation of information identified using pattern matching or data mining techniques;
- Anonymisation of information based on privacy blacklisting or whitelisting;

We will discuss each of these strategies more in detail below.

7.6.1 Anonymisation Strategies Directly Related to Privacy Leakage Metrics

Cluster-based Anonymisation

Cluster-based anonymisation enforcement means that the IDS alarms are classified according to the cluster number of the LMM the length-corrected Shannon entropy $H_1^{'}(y_i)$ of an IDS alarm belongs to, assuming a hard clustering strategy. This means that the IDS alarm is assigned to the cluster of the LMM that $H_1^{'}(y_i)$ has the largest probability of belonging to. The mixture model can in this way be used for authorisation/anonymisation of each incoming IDS alarm depending on attack vector, for example to leave information in clusters corresponding to attack vectors with no or little privacy leakage as they are, and anonymise the information in other clusters that by investigation have been shown to leak private or confidential information.

Overall, a cluster-based anonymisation scheme can be considered robust and predictable for a given LMM model, given that the clusters do not overlap significantly, and also that the underlying distribution of IDS alarms does not change significantly. This means that it will require a supporting management process to monitor the relevance of the mixture model, and update the model if the data changes sufficiently to warrant a model update (for example due to introduction of new attack vectors).

The LMM can can for example be validated using statistical hypothesis testing to verify if the components of the mixture model are still supported by the data for a sufficiently large sample of IDS alarms [31]. The amount of overlap between the distributions in the LMM can be calculated using the overlapping coefficient denoted by O [147, 116]:

$$O(f,g) = \int_{-\infty}^{\infty} min\{f(x), g(x)\} dx \qquad (7.6.1)$$

The overlapping coefficient between samples belonging to a given attack vector θ_k in the LMM and the sum of the samples belonging to the other $K-1$ components in the mixture model can be expressed in terms of the length-corrected Shannon entropy as:

$$O(\theta_k, \Theta \backslash \theta_k) = \sum_{i=1}^{N} min\{w_{i,k} H_1^{'}(y_i), \sum_{j \in 1,\ldots,K, j \neq k} w_{i,j} H_1^{'}(y_i)\} \tag{7.6.2}$$

Where $\Theta \backslash \theta_k$ denotes the relative complement between all components in the mixture Θ and the mixture component (or attack vector) θ_k. A requirement can for example be that the overlapping coefficient must be less than a given percentage of the effective number of samples in the cluster, if the model is to be used for authorisation or anonymisation purposes. The overlapping coefficient can also be calculated at regular intervals, to trigger a revalidation of the privacy policy if the overlapping between the mixture components exceeds the required threshold.

Attacks and Attack Mitigation for Cluster-based Anonymisation

The standard deviation of entropy metric may open up for shirking attacks from a privacy perspective, meaning that an internal adversary can avoid doing his duties by doing technical adaptations that affects the privacy metrics. This can be done by deliberately overfitting the clustering model, or operating with unrealistic privacy impact factors, since this would reduce the overall privacy leakage measurements. Another risk that must be considered, is the risk of malicious insider attacks, where for example the data controller sets a too restrictive privacy policy, essentially rendering attack detection ineffective. One way to handle this, is to use separation of duty constraints where the security manager and data controller need to collaborate on deploying new privacy policies. This can be implemented using the key sharing functionality of the reversible anonymisation scheme.

It is therefore important to have *external quality assurance* of the privacy policy by both the data controller and also by external quality certification agencies to reduce the risk of such privacy attacks.

Privacy Leakage Threshold Based Anonymisation

Another possibility may be to use a threshold value on the privacy leakage metric π^L over the sample of IDS alarms Y as basis for authorisation decisions. A possible attack on such a threshold-based authorisation scheme, is an entropy sliding attack. This is an attack where the external attacker over time changes the average entropy slowly by forging a given entropy, however being careful not to exceed the threshold for anonymisation. In this way, the external attacker may be able to change a plaintext attribute to an encrypted attribute over time (or vice versa) by shifting the average entropy. This means that the underlying attack vector definition, which is defined by the average entropy $\overline{H_\alpha}$, can change significantly, without the data controller being notified by the change.

It is possible to mitigate this attack by adding hypothesis tests that verify whether the LMM is supported by the data [31]. If the hypothesis tests indicate that the distribution has changed significantly, then the tool should trigger a revalidation of the policy by the data controller.

Another, and more serious problem with such an approach is that the authorisation decisions over time may change depending on the sample of IDS alarms. This means that threshold-based authorisation decisions only could be cached for a relatively short time interval, depending on the size of the IDS alarm sample window and the flow of incoming IDS alarms. This may also cause unintended privacy leakages during phases when the measured information leakage changes in magnitude.

Such behaviour is not acceptable in use cases where zero information leakage is tolerated (e.g. for monitoring of health institutions). It is also problematic from a security perspective, since an external attacker can exploit such a dynamic access control scheme to force anonymisation of the payload the IDS rule triggers on, by deliberately increasing the entropy of the traffic matched by the IDS rule, to hide attacks.

It is important for a managed security service provider that the IDS behaves consistently, which means that such a dynamic authorisation policy is not a good idea. It would confuse the security analysts if an IDS rule that used to be in cleartext suddenly became anonymised. Other strategies than using a privacy leakage threshold for anonymisation of information should therefore be used to ensure that the IDS shows consistent behaviour over time.

The overall conclusion is therefore that a privacy leakage threshold based authorisation/anonymisation scheme is not recommended, since it easily can be exploited by a determined adversary, it may also be confusing for the security analysts and it may cause privacy leakages which in most use cases will be considered not acceptable. It must however be noted that a threshold based approach still may be useful as part of a model validation strategy, to trigger model revalidation if the privacy leakage exceeds a given threshold. This means that the data controller could get a warning that the anonymisation scheme may not be effective, for example due to introduction of new attack vectors, however this would not impact the operation of the service.

Privacy Leakage Map Threshold Based Anonymisation

The detailed privacy leakage map basically has the same deficiencies as using the privacy leakage threshold based approach, and is therefore not recommended for dynamically enforcing authorisation/anonymisation of information.

The privacy leakage map is in other words useful for identifying which octets that should be anonymised, however an entropy threshold based anonymisation approach is not recommended. It is then better and safer to use anonymisation of octet ranges, which by investigation of the privacy leakage map have been found to be problematic from a privacy or confidentiality perspective.

It must however be noted that also a threshold based approach may be useful as part of a model validation strategy, to trigger model revalidation if entropies in the privacy leakage map exceed a given threshold. This means that the data controller could get a warning that the anonymisation scheme may not be effective, however this would not directly impact the operation of the service, as a dynamic access control scheme would.

7.6.2 Other Anonymisation Strategies

These are anonymisation strategies that indirectly may benefit from using information from the privacy leakage metrics.

Anonymisation of Octet-ranges

The case studies show that it may be desirable to anonymise entropy sources within each cluster representing attack vectors. This means that a *fixed range* *of octets,* or a set of such octet ranges, representing entropy sources in the element being considered, should be anonymised.

This can be considered a robust and predictable anonymisation scheme, as long as the underlying assumptions hold (i.e. the probability distribution does not change significantly and the clusters being considered do not start overlapping significantly). This means that monitoring of other entropy sources, that are not covered by the anonymisation scheme, will still need to be performed to ensure that these entropy sources do not start leaking a significant amount of private or confidential information.

This can be monitored by checking the privacy leakage of the anonymised IDS alarms, and trigger a model revalidation procedure if the entropy distribution of the privacy leakage map changes significantly, for example using hypothesis testing, or by visual inspection by rerunning the clustering algorithm. The privacy leakage map can be used to identify where entropy sources are in the data samples.

Using a default DENY authorisation and anonymisation policy, and declassifying only octet-ranges that are assumed unproblematic, is considered the safest approach from a privacy perspective to reduce the risk of privacy leakages, and at the same time ensure that necessary information for attack investigation is visible. This allows for revealing only the information known not to not leak private or confidential information. However, such an approach also means that relevant stakeholders, for example security analysts, must be given conditional access to information in the IDS alarms in order to do attack analysis where this is needed. This is however still better than most existing approaches, since access to sensitive information can be accounted for, so that there is transparency on who have had access to what information.

Anonymisation of information based on privacy blacklisting or whitelisting

Anonymisation based on privacy blacklisting or whitelisting can be implemented in XACML-based privacy policies using the proposed reversible anonymisation scheme. XACML-based authorisation based on blacklisting and whitelisting was first demonstrated in [77], and the lists can be implemented in a similar way by extending the XACML Policy Information Point to reflect blacklist or whitelist data from a data repository.

The clustering tool used for privacy leakage calculations can also be used to investigate the underlying data in the matching clusters. This makes it easier to identify information within IDS alarm clusters that are candidates for privacy blacklisting or whitelisting.

A synergy from a security perspective, is that this functionality also can be used to implement traditional blacklisting or whitelisting from a security perspective, by annotating or filtering IDS alarms that either are known suspicious (blacklist) or are known false alarms from trusted sources (whitelist). This makes it possible to improve efficiency and reduce the alarm processing load for the Security Operations Centre.

Anonymisation of information identified using pattern matching or data mining techniques

Anonymisation can also be based on pattern matching, for example by testing for given patterns in the payload of the IDS alarms using regular expressions and anonymise these as we previously identified in one of the case studies. XACML already supports such functionality. In more complex scenarios, data mining techniques may be useful to detect and anonymise information that is private or confidential, however investigating such approaches is left as future research.

7.7 Related Works

This chapter elaborates on how privacy leakage metrics can be integrated as part of an information security management process. An objective of this chapter is to describe how the privacy metrics can be combined with a privacy enforcement mechanism based on the proposed reversible anonymiser to support planning and development of privacy policies, validation of the privacy enforcement scheme and also demonstrate metrics-supported authorisation and anonymisation policies.

This chapter is based on and extends the privacy leakage metrics for IDS alarms described in [138], which in turn is based on and extends the theory of quantitative information flow [122, 121], as well as building on the

EM-based clustering method for finite mixture models in [31, 49]. Privacy metrics based on entropy have also been investigated by several other authors. Rényi-entropy was investigated as a generalised privacy metric that generalises Shannon, Min and Max-entropy in [29]. This chapter builds on the comparative analysis of entropy-based metrics in [138], which concluded that Shannon octet-based entropy [118] was best suited for measuring information leakage from IDS alarms based on theoretical considerations, Monte-Carlo simulations and analysis of IDS alarms. It does therefore not consider a generalised entropy, like Rényi entropy [107].

There are some similarities with the theory of differential privacy [40, 41, 42, 113], which is used to evaluate the privacy leakage of databases where sensitive information has been perturbed. These methods are founded in information theory, however these techniques use perturbation (adds Laplacian noise) to hide sensitive parameters, whereas we model Laplacian information around a mean value as attack vectors detected by IDS rules. Another difference is that the metric proposed here has a length correction that optimises the metric for distinguishing between text-based and encrypted/coded data sources. The length correction also ensures that the measured information leakage increases with length for longer payloads (>100 octets). Our theoretical model and interpretation of privacy leakage is therefore clearly different from differential privacy, and is perhaps more closely related to the concept of mutual information in information theory [118], than obfuscation of information in databases.

There also exist some examples of other methodologies that incorporate privacy requirements into the design process. One example is the PRIS method, which addresses privacy requirements during system design [71]. Another example is the eTVRA methodology, which is a threat, vulnerability and risk assessment method developed by ETSI [109]. Both of these are high-level methodologies that are useful in an early phase of a privacy enhanced system, but they are less well suited for managing and reducing privacy leakage gaps during operation. These high-level methodologies do however not describe in detail how quantitative privacy leakage metrics and privacy enforcement mechanisms can be combined, to support manage and reduce privacy gaps during operation. These methodologies can therefore be considered complementary rather than competing to the method proposed here.

The proposed scheme is also related to privacy-enhanced IDS solutions, e.g. [51, 22, 50, 123, 82]. Most existing privacy-enhanced IDS solutions are based on a pseudonymisation scheme, whereas our scheme is based on a reversible anonymisation scheme for XML documents. All of these are solutions based on cryptography and are not being supported by privacy leakage metrics to verify correct operation of the privacy policy as the solution proposed here does.

7.8 Conclusion

This chapter shows how privacy leakage metrics supported privacy enforcement can be used as a structured methodology to reduce leakage of private or confidential information. The proposed method may both have privacy and security benefits, for example by protecting private or confidential information against disclosure and also classifying attack vector clusters that by inspection have been shown to consist of false alarms. Avoiding these would improve the efficiency of security operations.

The privacy leakage map allows identifying where information is leaking in the data representing a given attack vector cluster. This information can be used to plan a privacy enforcement scheme using a range of enforcement mechanisms that have been discussed here. The efficacy of anonymisation policies can be verified by using entropy-based privacy metrics to compare the entropy before and after anonymisation.

The proposed method supports the *Plan Do Check Act* (PDCA) model for improvement [92], that amongst others is adopted by the ISO27k set of security management standards, as illustrated in figure 7.2.2. The privacy metric and privacy leakage map can be used in the *Planning* phase to identify privacy leakages and propose methods for reducing the privacy leakage. The planned actions can then be enforced in the *Do* phase by defining a privacy policy for the privacy enforcement scheme. When the updated privacy policies and privacy enforcement scheme have been deployed, the privacy metric can *Check* that the planned privacy enhancements and anonymisation scheme work as intended based on techniques derived from the privacy leakage metrics. Significant deviations can then be identified and used to trigger an *Action* to reassess the privacy policy and perform a new PDCA cycle.

This approach adds value to traditional privacy impact assessments based on qualitative indicators and questionnaires, by supporting a structured methodology for identifying what, how much and where information is leaking in services based on quantitative privacy leakage metrics.

7.9 Future Work

The privacy metrics discussed here will need to be verified in a larger study involving commercial MSS providers. Hopefully, this research based on a limited dataset will encourage security organisations and managed security service providers to collaborate on performing larger studies to confirm that these metrics work as intended. It would also be useful to extend the privacy leakage map to show changes over time, for example by plotting it as a 3D map. A larger sample, monitored over time, is however needed to do this.

Investigating how the privacy leakage metrics can be extended to efficiently support anomaly-based IDS is left as future work. We believe that this can be achieved by combining clustering of different features of the IDS alarm or message with a decision tree based approach to limit the number of matching clusters for elements where the distribution function is too crowded by underlying mixture components.

This study only scratches the surface of privacy related problems services like MSS by investigating primary sources of information leakage. Future work is to perform a more comprehensive analysis of such privacy leakages, for example using metrics and techniques like k-anonymity, l-diversity or similar [126, 86, 27], which consider the risk of privacy leakages from cross correlating data. It may also be possible to use more complex AI-based techniques for privacy preservation, for example using relevance metrics like term frequency/inverse document frequency to aid the data controller in detecting information that may violate privacy or confidentiality [148].

Part IV

The Way Ahead

Part IV contains the general discussion and conclusion of the dissertation, including discussing to what extent the problem has been solved, as well as the expected impact of the proposed solution. It furthermore outlines starting points for future research and development.

Chapter 8

Discussion, Conclusion and Future Work

This chapter concludes the dissertation and discusses future research and development. The objective is discussing to what degree the proposed privacy-enhanced network monitoring solution solves the problem statement and other privacy objectives stated in the introduction. The chapter is structured as follows: The next section discusses whether the seven foundational principles of Privacy by Design are covered. Section 8.2 subsequently discusses the expected impact of the solution, and in particular whether it can be considered usable, easy to deploy, effective and robust according to the technological objectives in Section 1.4.3. Section 8.3 concludes the dissertation with a discussion on whether the solution supports the initial problem statement. Finally, Section 8.4 discusses further development of the solution and Section 8.5 outlines possible starting points for subsequent research.

8.1 Is Privacy by Design Supported?

The proposed approach for privacy-enhanced network monitoring is based on a reversible anonymisation scheme and metrics-supported enforcement of privacy and confidentiality. This approach can be used to design network monitoring systems that are characterised by proactive rather than reactive measures for enhancing privacy. The proposed approach is clearly better than existing IDS schemes, since it proactively can provide protection of private

PbD principle	Supported?	Comments
1. Proactive	Yes	
2. By default	Yes	
3. Embedded	Yes	Easy to integrate into existing IDS technologies.
4. Positive sum	Yes	Win-win between security and privacy.
5. Full life cycle protection	Yes	
6. Transparency	Outlined	Detailed implementation left as future work.
7. Respect for users	Yes	Ensured by data controller.

Table 8.1.1: How the scheme supports the Privacy by Design Principles.

or confidential information by supporting a continuous improvement process. The privacy metrics can furthermore verify whether protective privacy enhancing techniques work as expected, and trigger actions if privacy leakage changes significantly compared to expected values. This means that the Privacy by Design (PbD) principle 1 in Table 8.1.1 is supported.

For some use cases, for example for health institutions, it may not be acceptable that person sensitive information leaves the hospital perimeter. The proposed privacy-enhanced IDS scheme can support such a use case using a default DENY policy in accordance with the Privacy by Default principle (PbD principle 2), so that the privacy policies ensure that *any* sensitive information in the IDS alarms sent to an outsourced MSS by default is anonymised. Information that is required to notify security analysts about the attack, for example the IDS alarm identity and type of attack it triggered on, can subsequently be permitted by using the *declassify* operation, introduced in Section 5.4. The IDS alarms may additionally be copied to an internal alarm database, where all potentially sensitive information in the IDS alarms is stored in an encrypted security level, so that only authorised entities can access this data on a needs basis, for example to investigate suspected attacks. This means that an outsourced MSS operator will be able to do basic attack surveillance. More detailed attack analysis is possible by accessing further information from the alarm database inside the hospital perimeter.

It is assumed that the existing IDS infrastructure shall be used as far as possible to capitalise on existing security investments and knowledge. This is a more realistic assumption than assuming that a completely new and perfect privacy enhanced IDS scheme shall be built from scratch. The scheme uses the extension facilities of IDMEF to provide seamless integration with existing IDS alarm databases. Backwards compatibility with existing applications, both intrusion detection systems, alert databases and more comprehensive Security Information and Event Management Systems (SIEMs), is a prerequisite for market uptake of privacy-enhanced IDS services. The conclusion from this discussion is that the proposed approach is well embedded into existing systems, and therefore supports PbD design principle 3.

The proposed privacy metrics allow measurement of IDS rule precision, which can be used to improve the IDS rules to reduce the amount of unnecessary, possibly privacy leaking IDS alarms. This also means that IDS rules with large information leakage, which may have a significant risk of leaking private or confidential information, can be identified and proper actions (e.g. tune the IDS rule or anonymise sensitive data) can be enforced by the data controller. However, the privacy metrics can also be used to prove the effectiveness of preventive actions taken to improve the privacy, for example to verify whether an improvement causes a network monitoring service to leak less sensitive information, or whether an anonymisation scheme remains effective. This allows for detecting faulty privacy configurations and unauthorised security leakages - for example non-encrypted traffic going on links that only are allowed to convey encrypted traffic. This can also be used to detect anomalous traffic, for example from malware. There is furthermore synergy between the proposed security metrics and Denial of Service attack detection, since such attacks typically will skew the distribution of entropies significantly, which can be detected as an anomaly. The proposed multilevel security based reversible anonymisation scheme means that access to private or confidential information in the IDS alarms can be controlled. This means that there is a win-win situation by implementing the proposed privacy metrics and enforcement mechanism both from a security and privacy perspective, as required by the 4. Privacy by Design principle.

Sensitive information can for all practical purposes be protected from inception by integrating the anonymiser in an appliance that enforces controlled access and encrypts the link to the anonymiser. Furthermore, the reversible anonymisation scheme in Chapter 5 enforces protection of sensitive data during its lifetime, and the time-based data expiry scheme in Section 5.6 can be used to ensure safe destruction of sensitive data after a given retention time. This means that the proposed scheme can support full protection of data from inception and until destruction, as required by the PbD principle 5.

The reversible anonymisation scheme intrinsically supports some degree of transparency and accountability for access to sensitive information, since it indicates who have access to this information. However, the details on how to enforce transparency is considered beyond the core scope of the dissertation, since this typically will be implemented as part of a larger system architecture supporting the security operations, and not only as part of the anonymiser. The dissertation has therefore not discussed in detail on how transparency can be enforced. It has only outlined that access to sensitive data should be logged, in order to support transparency (PbD principle 6), and that this can be enforced based on XACML obligations when authorising access to sensitive data. Full support the PbD design principle 6 is therefore left as future research and development.

The proposed scheme can be said to respect the users (PbD design principle 7), by allowing design of privacy enhanced network monitoring systems that prioritise the interests of the individual by supporting strong privacy defaults (the default DENY scheme), and also by supporting separation of duties where privacy interests can be enforced by a separate entity, the data controller. Network monitoring systems should not need to be aware of who the underlying users are, so there is no need to provide user-controlled access to own data. Appropriate notice that network monitoring is being performed should be given, however this is considered outside the scope of the technical solution proposed here. This will need to be enforced by an overarching methodology that amongst others checks that companies using managed security services provide an appropriate warning to their customers.

Table 8.1.1 sums up how the proposed scheme supports the Privacy by Design principles. The only main principle that is not explained in detail in the dissertation, is logging to ensure that operations on sensitive data are transparent and accountable. The section on future work gives a brief outline of the research needed to support this.

8.2 Impact

The dissertation provides new insight on how to measure privacy leakage for IDS rules and it also describes and implements a privacy enforcement scheme that supports reversible anonymisation of XML messages according to a given privacy policy.

Considering the technological objectives in Section 1.4.3, the solution proposed in the dissertation can be considered *usable* for MSS providers, even if IDS alarms to a greater extent than today are being anonymised. The reason is that the metrics can be used to optimise the IDS rule set, so that rules with a significant risk of privacy leakage can be identified and subsequently either be improved from a privacy perspective or anonymised. Furthermore, the anonymisation scheme can be set up to provide multi-level security with reversible anonymisation, so that authorised parties or services will be able to see the same data as they do today if needed. The usability may in some areas even be increased compared to current IDS solutions, since the EM-based clustering and privacy leakage metrics facilitates easier visualisation and data mining of suspicious events than current IDS alarm databases and SIEM solutions do.

Controlling access to this sensitive information also opens up for supporting transparency/accountability on who accesses sensitive information and why. This supports far stricter enforcement of the need-to-know principle than IDSs typically support today with more detailed control over access to sensitive information. Handling more than one security level can in addition be useful for graded systems used for critical infrastructures or military systems.

It is furthermore possible to improve the attack detection precision of IDS rules to reduce the measured privacy impact, also in those cases where anonymisation is not considered an option. This means that the MSS provider will get a rich toolbox that can be used to optimise the security operation and also to implement privacy policies that anonymise private or confidential information. The overall usability is therefore not necessarily significantly reduced for the MSS provider if the privacy-enhancing metrics and technologies are applied, at the same time as the accountability and transparency regarding investigation of private or confidential data is significantly improved.

The proposed methods are easy to *deploy* on top of existing IDS infrastructures, since the anonymiser acts as a proxy between the IDS and the alarm database using the standardised IDMEF protocol. Furthermore, only small modifications are needed to support other XML-based protocols. One such example may be to monitor privacy leakage in SOAP-based web services, where the anonymiser and deanonymiser can be integrated as components in an Enterprise Service Bus.

The proposed metrics have been shown to be *effective*, by being able to detect IDS rules with a significant risk of leaking private or confidential information. Furthermore the metrics can be used to verify that an anonymisation scheme is properly enforced. The experiments show that the solution should have sufficient performance to be used in small to medium scale IDS deployments.

The technical means for privacy enforcement in the anonymiser/proxy can be considered *robust* for authorisation or anonymisation decisions, based on the discussion in chapter 7.

To sum it up, this means that the proposed solution should have significant impact towards making privacy-enhanced IDS services a reality. This is an area of research that so far mostly has seen theoretical solutions founded on cryptography, e.g [123, 22, 51]. This can in particular be useful to provide controlled access to private or confidential information for outsourced Managed Security Service providers, where the provider is not being fully trusted to see certain information in the network being monitored. The solution may also be useful for exchanging anonymised alarm data or data forensic information between semi-trusted managed security service providers or CERT teams to provide more efficient exchange of anonymised attack related information during cyber-attacks.

The privacy enforcement methods and privacy metrics make privacy-enhanced IDS a viable alternative for managed security service providers, since it allows a continuous improvement process of both the privacy and security objectives of a network monitoring service. Furthermore, the proposed solution is based on standards, integrates well with existing IDS technologies and can be implemented on top of an existing IDS infrastructure. Other proposed approaches for privacy-enhanced IDS, for example [51], may require a non-standard protocol for conveying the IDS alerts, and it may require a totally different IDS rule set, something that significantly reduces the practical applicability of such methods.

8.3 Conclusion

The dissertation shows how entropy-based metrics can be combined with a reversible anonymisation scheme to provide a structured methodology for reducing the privacy leakage from IDS alarms and other services. EM-based clustering is used to identify the individual attack vectors that an IDS rule triggers on, and entropy-based privacy leakage metrics can be used to identify which attack vectors that leak a significant amount of private or confidential information and also where these privacy leakages occur within the IDS rule. The XACML-based reversible anonymiser can subsequently be used to enforce a privacy policy which reduces the privacy leakages.

The proposed method supports the Plan Do Check Act (PDCA) model for improvement, where entropy metrics are used in the planning phase to identify privacy leakages and to propose methods for reducing these leakages. Planned actions can subsequently be enforced using the XACML-based reversible anonymiser or by tuning the IDS rule set to be less privacy leaking. Privacy metrics can then be used to check that the planned privacy enhancements work as expected, and significant deviations from the expected privacy leakage can be used to trigger new actions to reassess the privacy policy and perform a new PDCA cycle.

This is a flexible solution that has been integrated into existing signature-based intrusion detection systems, and that furthermore can support the Privacy by Design principles [25]. The only principle that is not specified in detail, is secure logging, which can be added based on existing schemes, e.g. [74]. The solution furthermore fulfills the technological objectives in Section 1.4.3, by being usable, easy to deploy and effective. Overall, this means that the proposed approach solves the research problem in section 1.2, since it:

- proposes suitable privacy leakage metrics which can be used to detect leakage of private or confidential information in IDS alarms;
- implements a fine-grained reversible authorisation and anonymisation

of information in IDS alarms using the eXtensible Access Control Markup Language (XACML), which is supported by the proposed privacy leakage metrics;

- and proposes a methodology that shows how the privacy leakage metrics can be connected to the technical solution for privacy enforcement, to support a continuous improvement process.

The solution can be seamlessly integrated into existing IDS and SIEM tools that support IDMEF, and can with small modifications be extended to support other protocols than IDMEF. The solution has been implemented and tested, and the performance should be sufficient for small to medium scale IDS deployments. Finally, it is expected that the privacy enforcement mechanism, privacy metrics and methodology for combining these will be useful for controlling leakage of private or confidential information also in other use cases than privacy enhanced network monitoring.

8.4 Future Development

The reversible anonymiser, privacy leakage metrics and methodology will be further developed in the PRECYSE EU-project[1]. The general idea is to integrate the anonymiser and deanonymiser into an Enterprise Service Bus (ESB), as illustrated in Figure 8.4.1, so that services which need multi-level security or anonymisation policies can use these components [132]. The initial part of an ESB integration, an IDMEF publishAlert interface, has already been implemented and works. We are able to store anonymised IDS alarms in PreludeIDS. A planned extension of the solution is to implement a secure logging scheme as a separate ESB component.

The PRECYSE architecture consists of an Information Security Management (ISM) module that performs risk and vulnerability analysis, a Control module that runs a Security Operations Centre which amongst others performs IDS alarm handling using a SIEM tool, IDS alarm correlation and also system and policy configuration management. The Control module manages a set of one or more Domains, which contain a set of existing and PRECYSE

[1] PRECYSE http://www.precyse.eu

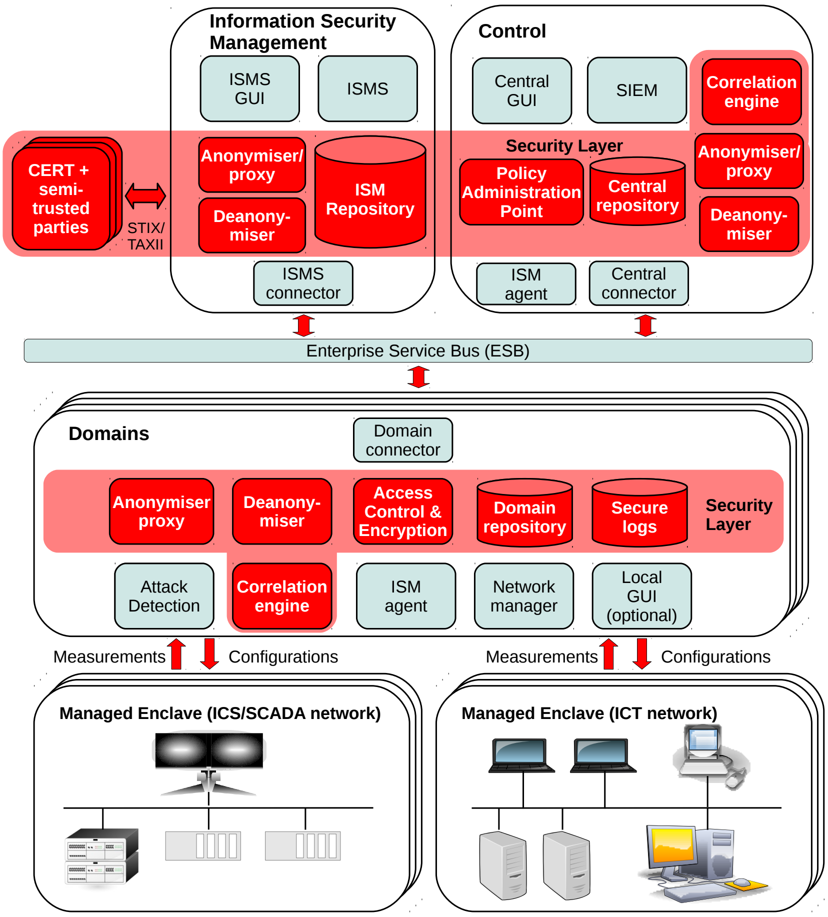


Figure 8.4.1: PRECYSE Privacy and Security Architecture [132].

specific security tools, like Snort, OpenNMS, OpenSCAP etc. for performing threat and vulnerability detection on target Enclaves [132]. An Enclave is here considered a part of the network or critical infrastructure that is managed using a common set of security policies. Privacy enhancing techniques will be implemented in several parts of the architecture, as illustrated in Figure 8.4.1. These are discussed more in detail below.

The privacy leakage metrics and methodology will be integrated into the PRECYSE information security management and risk assessment methodology which is based on the open risk assessment standard Magerit [33]. This allows for supporting quantitative risk analysis to detect and mitigate privacy and confidentiality gaps in the security configuration.

An advantage by using the reversible anonymiser, is that the information protection scheme is separated from the underlying functionality of the protocols or services being protected. This allows for implementing XACML-based information protection schemes for any XML-based protocol, and adds flexibility in adapting the information protection scheme according to business requirements compared to using a hard-coded XML-Encryption based solution.

The reversible anonymisation scheme may for example be used for cryptographic protection of the XACML policies, to ensure trustworthy deployment of privacy and security policies. It can also be used to protect vulnerability information from vulnerability assessments in Open Vulnerability and Assessment Language (OVAL) format[2], since information about system vulnerabilities for a given critical infrastructure should be considered confidential information [132].

Reversible anonymisation may also be used to support sharing of attack related information between organisations, for example by adding a security layer to information conveyed in the Structured Threat Information Expression (STIX) format. This is an XML format for exchanging cyber-threat information or information about countermeasures between organisations [130]. Using the reversible anonymisation scheme means that only authorised stakeholders can access information considered sensitive by the originating organisation.

[2] OVAL https://oval.mitre.org/

The reversible anonymiser may in addition be useful for protecting the confidentiality and integrity of system configurations. It can for example be combined with the scheme proposed in [136], to support location-aware role-based deployment of IDS rules. The reversible anonymiser can furthermore be used to cryptographically enforce that a given workflow of operations and authorisations must be followed when updating these. It supports key sharing, which can be used to reduce the risk that corrupt or malicious insiders are able to destabilise a critical infrastructure by faulty or malicious configuration deployments. The methods and tools developed in this dissertation can in other words be useful in a range of different use scenarios beyond building privacy-enhanced intrusion detection systems.

8.5 Future Research Directions

This dissertation only scratches the surface of privacy and confidentiality related problems for network monitoring systems. It does for example not cover how information leakage through inference or cross-correlation between different elements of an IDS alarm can be detected and avoided. This would require taking the privacy leakage metrics and evaluation even further to evaluate the anonymity set that can be expected for private or sensitive information, using metrics like k-anonymity [37], l-diversity [86] or differential privacy [40, 41, 42]. This is a large research area that is left as future work. The proposed privacy leakage metrics only measure the primary sources that may leak private or sensitive information. However, being able to do this is an initial approach that can and should be considered before more elaborate analyses of the anonymity set are performed. It would furthermore be interesting in the future to do comparative analyses on how well different MSS providers perform using the proposed privacy leakage metrics.

A limitation with the current implementation of the reversible anonymisation scheme is that there is no efficient way to perform on-demand data mining that searches the encrypted information (e.g. for attack investigation), apart from letting an authorised/trusted party or application decrypt the sensitive information. It would be useful to do further research on efficient privacy-enhanced database schemes to support certain queries over the encrypted information. This can for example be implemented by extending the anonymisation scheme to store a cryptographic pseudonym that would allow certain calculations or tests to be done on the encrypted value, in a similar way as CryptDB does [103]. Examples of such cryptographic schemes is order preserving or homomorphic encryption [103]. The solution could then be taken one step further to implement privacy-enhanced operations and queries for XML databases.

Future work furthermore involves investigating the details of how transparency of the proposed solution can be enforced, for example based on a combination of a secure logging scheme and using XACML obligations to do logging of authorisation decisions to sensitive data in the policies, in a similar way as [74]. The reversible anonymisation scheme may be used as a building block for implementing secure logging schemes.

Appendix A: Integrating Privacy Leakage Metrics in Anonymiser

Appendix A describes how the fine-grained XACML-based authorisation and anonymisation scheme for XML documents in Part II has been extended to implement the metrics supported privacy enforcement scheme described in chapter 7. This material has been split out from chapter 7, since it focuses on the technical details needed to implement the metrics supported privacy enforcement scheme. This information did not fit into the more high-level discussion in chapter 7, however it has been included as an appendix in the dissertation for completeness.

A.1 Integrating Entropy Metrics into the Anonymiser

Knowing exactly where sensitive information leakages are makes it possible to propose mitigation strategies to avoid such leakages as discussed in chapter 7. This section describes how the XACML-based anonymisation policies can be modified to detect and anonymise parameters, attributes or octets in XML messages which appear to leak a significant amount of private or confidential information. This can be implemented using the reversible anonymiser in Chapter 5. The data controller then needs to go through the octets of elements or attributes in IDS alarms with significant entropy variation and evaluate whether they cause significant privacy concern. If so, then XACML anonymisation policies can be instantiated to anonymise the data. Alternatively, privacy by default can be used by applying a default DENY policy where the security manager needs to argue for why certain fields need to be declassified.

```
<AttributeValue DataType="urn:ogc:def:dataType:geoxacml:1.0:geometry">
 <gml:MultiPoint srsName="&prile;ByteRanges">
   <gml:pointMember>
     <gml:Point>
       <gml:coordinates>20, 100</gml:coordinates>
     </gml:Point>
    </gml:pointMember>
    <gml:pointMember>
     <gml:Point>
       <gml:coordinates>200, -1</gml:coordinates>
     </gml:Point>
    </gml:pointMember>
   </gml:MultiPoint>
</AttributeValue>
```

Figure 8.5.1: Example GML MultiPoint format used for entropy data ranges.

A.1.1 Anonymisation Functions

In Section 7.6 we identified two primary privacy enforcement mechanisms that are considered useful:

1. Use the Laplacian Mixture Model from the EM-algorithm as classifier for IDS rules in order to support cluster based anonymisation policies.

2. Anonymise octet ranges that are considered problematic from analysis of the detailed vertical entropy map.

This section discusses more in detail how these anonymisation functions can be implemented in XACML. In the following, the XACML namespace is denoted as *&xacml*; and the XML Schema namespace is denoted as *&xs;*. The W3C namespace *http://www.w3.org/* is denoted *&w3c;* and our own extensions are defined in the namespace *http://www.prile.org/,* denoted by *&prile*;. It is assumed that the reader has a basic understanding of XACML. The next section elaborates on how such an XACML policy can be implemented. It is also assumed that the IDS alarms use the IDMEF XML format for IDS alarms [62].

Anonymise or Declassify XML Element based on Octet Range

This section gives a high-level functional description of how to anonymise or declassify XML elements or attributes based on a set of one or more octet ranges. The following anonymisation functions are executed by the Policy Enforcement Point (PEP) as XACML response obligations:

1. *anonymise(e, $octetlist$)*: obligation to anonymise the element or attribute referenced by *e*. This is an obligation policy function in the PEP that is used for default PERMIT policies. It is included in the XACML Response as an Obligation containing an <*AttributeAssignment*> with the *AttributeID* of the element being referenced. The optional $octetlist$ can be used to specify octet ranges to anonymise, as shown in Figure 8.5.1. If it is not present, then the entire element being authorised is anonymised.

2. *declassify(e, $octetlist$)*: obligation to declassify the element or attribute referenced by *e*. This is an obligation policy function in the PEP that is used for default DENY policies. It is included in the XACML Response as an Obligation containing an <*AttributeAssignment*> with the *AttributeID* of the element being referenced. The optional $octetlist$ can be used to specify octet ranges to declassify, as shown in Figure 8.5.1. If $octetlist$ is not specified, then the default behaviour is to declassify the entire element.

The *octetlist* is an optional list of $(start, end)$ points defining a data range. It is implemented as a GeoXACML *MultiPoint* data type [6], although the interpretation here will be data ranges and not 2D coordinates. Figure 8.5.1 illustrates how the MultiPoint data type can be used. Octet 0 is the start of the string, and the number -1 is used to denote the end of the string. An octet range of <*gml:coordinates*>*0, -1*</*gml:coordinates*> therefore matches the entire element. <*gml:coordinates*>*2, 2*</*gml:coordinates*> matches octet number 2 and <*gml:coordinates*>*10, 100*</*gml:coordinates*> matches the octet range $[10, 100]$.

Anonymisation of an IDS Alarm Element based on Laplacian Mixture Model

The Laplacian Mixture Model, identified by the EM-algorithm [31, 138], can be used as a classifier to identify which cluster the entropy of a given element or attribute of an IDS rule belongs to. This can be used to perform conditional anonymisation of IDS alarms depending on which attack vector cluster the IDS alarm element belongs to. The function for identifying the given cluster, based on the entropy of the given element, is shown below:

***int cluster(*$H_\alpha(e), \Theta_R$*)*:** Returns the cluster index the entropy $H_\alpha(e)$ belongs to, according to the Laplacian Mixture Model Θ_R for the given IDS rule R and IDS alarm element/attribute e [138].

This function can be used in an XACML Condition to anonymise the IDS rule, depending on which cluster it belongs to. It will typically be used as part of an XACML policy that returns an Obligation to *anonymise()* or *declassify()* information in the element or attribute e being authorised, based on the cluster it belongs to, as illustrated in Figure 8.5.5. This function takes the entropy $H_\alpha(e)$ as argument instead of the element e, to avoid leaking more sensitive information about e than necessary to the Policy Decision Point (PDP).

The cluster classifier uses a Laplacian Mixture Model, consisting of several Laplacian distributions. The Laplacian Mixture Model denoted by Θ_R is defined using the GeoXACML *MultiPoint* data type as a list of triplets consisting of the cluster model parameters: median $\tilde{\mu}_k$, scale parameter λ_k and mixing probabilities β_k, for a model k so that $\Theta_R = [(\tilde{\mu}_1, \lambda_1, \beta_1), ..., (\tilde{\mu}_K, \lambda_K, \beta_K)]$ for a Laplacian Mixture Model consisting of K elements, as illustrated in Equation 8.5.2. The model is initially created by clustering IDS alarm entropies in the alarm database using the Expectation Maximisation algorithm, as described in chapter 6. Furthermore, the data controller and Security Analysts must monitor the goodness of fit of the given model and overlap between mixture models, as described in section 7.6.1. The model will then need to be updated by redoing the clustering, if necessary, as pointed out in Section 7.6.1. We have earlier shown that the Laplacian distribution, denoted as $\mathscr{L}(H_\alpha|\tilde{\mu}_k, \lambda_k)$, is defined as [31]:

$$\mathscr{L}(H_\alpha|\tilde{\mu}_k, \lambda_k) = \frac{1}{2\lambda_k} exp\left(-\frac{|H_\alpha - \tilde{\mu}_k|}{\lambda_k}\right). \tag{8.5.1}$$

The cluster that the entropy of the element belongs to can then be calculated as:

$$cluster(H_\alpha, \Theta_R) = \underset{k \in 1,...,K}{\arg\max} \mathscr{L}(H_\alpha|\tilde{\mu}_k, \lambda_k) \cdot \beta_k. \tag{8.5.2}$$

This classification function is implemented as an XACML extension function denoted *&prile;function:cluster* in the XACML PDP. It takes the horizontal entropy of the element being authorised and the Laplacian mixture model Θ_R as arguments, and returns the cluster the entropy belongs to. It is assumed that the mixture model Θ_R is made available to the XACML Policy via the XACML Policy Information Point (PIP), alternatively it may be directly embedded into the policy, if the model is not expected to change frequently.

A.1.2 Trustworthy Authorisation Policy Updates

It is assumed that the data controller and security manager need to authorise the LMM model and privacy policy updates before they can be deployed to the production systems. Trustworthy authorisation policy updates can be implemented by adapting the multi-level security solution in Chapter 5 to encrypt XACML policies, using threshold cryptography to split the encryption key into shares ensuring that both the security manager, data controller and the trusted Policy Administration Point (PAP) application need to agree to decrypt and deploy the updated XACML policy during deployment. The details on how to do this is however left as future work. The PAP is assumed to not accept deployment of unsigned and unencrypted policies.

A.2 Use Case Example

Consider a use case based on SID 1:1437 Windows Multimedia download, which was analysed in section 7.5.1. Assume that it is desirable to anonymise cluster 2 of the decompressed data which contains targeted advertisements from Doubleclick, as shown in figure 7.5.1b. It is furthermore assumed that a default DENY protocol is in effect to avoid the risk of unknown private or confidential information being revealed. Information from octets 0-230 is considered unproblematic from a privacy perspective and is also useful from a security perspective, so this octet range will not be anonymised. Cluster 1 is not considered in this example, to keep the XACML example brief. It is trivial to extend the example to also cover the second cluster by adding another similar policy to it.

This can be implemented using the policy pseudocode in Figure 8.5.2. The declassification policy takes as arguments the IDS rule identity R, the length-corrected Shannon entropy $H_1'(e)$ and a reference to the element e

```
function DECLASSIFICATIONPOLICY($R, e, entropy$)
    if $R =' 1 : 1437'$ then
        $\Theta_R \leftarrow [(11.7, 2.9, 0.46), (13.7, 2.5, 0.54)]$
        if $cluster(entropy, \Theta_R) = 2$ then
            return $Permit, Obligation(declassify(e, [0, 230]))$
end function
```

Figure 8.5.2: Declassification Policy Pseudocode for SID 1:1437.

```
<Response>
  <Result ResourceID="PEP">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="&xacml;status:ok"/>
    </Status>
    <Obligations>
      <Obligation ObligationId="&prile;authorize-elements" FulfillOn="Permit">
        <AttributeAssignment AttributeId="&prile;default-policy:DENY"
          DataType="&xs;string">
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:1:id"
          DataType="&xs;string">/Alert/AdditionalData[@meaning='payload']
        </AttributeAssignment>
        <AttributeAssignment AttributeId="&prile;resource:1:assertion:1:scope"
          DataType="&xs;string">/Alert/Classification/@ident
        </AttributeAssignment>
        <AttributeAssignment
          AttributeId="&prile;resource:1:assertion:2:lcorr-shannon-entropy"
          DataType="&xs;string">/Alert/AdditionalData[@meaning='payload']
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Figure 8.5.3: XACML response to initial authorisation of the IDS-PEP.

being declassified. The declassification policy first checks if the IDS rule identity matches the intended IDS rule with SID 1 : 1437, and if it does, then the alarm cluster is calculated from the entropy and the Laplacian mixture model Θ_R. If the entropy matches alarm cluster 2, then the policy adds the obligation to declassify the octet range $[0, 230]$ of the element e, and returns a Permit XACML Response with these Obligations. The XACML policy that implements this pseudocode is shown in Figures 8.5.5 and 8.5.6. The detailed solution of how the XACML authorisation scheme is implemented is described in the following subsections.

A.2.1 Initial XACML Authorisation

The response to the initial XACML Request is shown in Figure 8.5.3. The response contains an XACML Obligation that defines the initial configuration of the XACML Decision Cache based anonymiser, similar to what is proposed in [135]. The first XACML attribute of the obligation is *&prile;default-policy:DENY*, which specifies that this is a default DENY protocol.

The XACML attribute *&prile;resource:1:id* contains an XPath expression */Alert/AdditionalData[@meaning='payload']* which refers to the IDMEF element being authorised. This element refers to the payload sample of the IDS rule. The next XACML attribute *&prile;resource:1:assertion:1:scope* defines the XPath scope expression */Alert/Classification/@ident*, which contains the unique identifier of the IDS rule R.

Finally, *&prile;resource:1:assertion:2:lcorr-shannon-entropy* is an instruction to the PEP to calculate the length corrected Shannon entropy $H_1'(e)$ of the given payload element e before the element is sent as parameter in an XACML Request. This strategy avoids revealing the element being authorised to the PDP.

A.2.2 XML Element Authorisation Request

The PEP will first parse the IDS alarm using an XML parser. It will then perform XPath searches to retrieve the XML elements and attributes from the IDS alarm that are required to perform the authorisation decisions. The XPath expression for *&prile;resource:1:id* refers to the payload element that is asked to be declassified, *&prile;resource:1:assertion:1:value* refers to the rule ID of the IDS rule R that is being considered (here SID 1:5976), and *&prile;resource:1:assertion:2:value* refers to the length corrected Shannon entropy of the payload element of the IDS alarm.

The PEP will in this case retrieve the payload using the XPath expression */Alert/AdditionalData[@meaning='payload']* and will subsequently calculate the length-corrected Shannon octet-entropy on the result of this XPath expression. Since this is a default DENY policy, then all elements and attributes of the IDS alarm will subsequently be anonymised, using the reversible anonymiser in Chapter 5.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="&xacml;context:schema:os"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="&xacml;context:schema:os
         http://docs.oasis-open.org/xacml/\
         access_control-xacml-1.0-context-schema-os.xsd">
  <Subject>
    <Attribute AttributeId="&xacml;subject:subject-id"
               DataType="&xs;string">
      <AttributeValue>soc1@outsourced.example.com</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId="&xacml;resource:resource-id"
          DataType="&xs;string">
      <AttributeValue>&prile;resource:1:id</AttributeValue>
    </Attribute>
    <Attribute AttributeId="&prile;resource:1:assertion:1:value"
          DataType="&xs;string">
      <AttributeValue>1:5976</AttributeValue>
    </Attribute>
    <Attribute AttributeId="&prile;resource:1:assertion:2:value"
          DataType="&xs;string">
      <AttributeValue>11.26</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="&xacml;action:action-id"
               DataType="&xs;string">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
</Request>
```

Figure 8.5.4: XACML request to declassify XML element.

The PEP will then go through all the resources defined in the initial XACML policy, and perform XACML Requests asking for permission to read elements that need authorisation as shown in Figure 8.5.4. For the given example, this means that one XACML Request will be sent that asks for permission to read resource *&prile;resource:1:id* for the Snort IDS rule with SID *1:5976* having payload entropy $H_1^{'}(payload) = 11.26$.

On receipt of the XACML Request, the PDP will evaluate the XACML policy described in Figures 8.5.5 and 8.5.6. The policy verifies that this is resource 1, that the IDS rule R is SID 1:1437 and that the desired action is *read*. If all the resources match the policy *Target* section and the clustering function evaluates to $cluster(entropy, \Theta_R) = 2$, then the policy will return an XACML *Permit* response, with an Obligation to declassify the identity of the IDS rule and the octet range $[0, 230]$ of resource 1, which refers to the payload. In addition, the decision has a cache timeout of one day.

On receipt of the XACML Response, the PEP will fulfill the Obligation as illustrated in Figure 8.5.6, and declassify the octet range $[0, 230]$ of the payload (*/Alert/AdditionalData[@meaning='payload']*), which means removing the default anonymisation for this octet range. The anonymised IDS alarm with this element declassified will then be sent to the central IDS alarm database. This is a simplified example for illustration purposes. There would normally be more elements that need declassification, for example the IDS alarm ID, time when the IDS alarm occurred, possibly IP addresses and port numbers etc. However, these can easily be added using the proposed scheme. If only one authorisation decision is desired per IDS alarm, then additional elements which are authorised in the same decision can be added to the Obligation in a similar way as for the IDS rule identity.

An advantage with this scheme, is that only one XACML authorisation decision normally would be required per IDS alarm, assuming that clustering was done on the payload of the IDS alarm. This means that the proposed scheme will be quite efficient, especially for alarm clusters with a narrow or discrete probability distribution.

```
<Policy PolicyId="idmef:anonymisation:policy"
        RuleCombiningAlgId="&xacml;rule-combining-algorithm:deny-overrides">
  <Target>
    <Resources>
      <Resource>
        <ResourceMatch MatchId="&xacml;function:string-equal">
          <AttributeValue DataType="&xs;string">urn:prile:org:resource:1:id
          </AttributeValue>
          <ResourceAttributeDesignator AttributeId="&xacml;resource:resource-id"
            DataType="&xs;string"/>
        </ResourceMatch>
      </Resource>
      <Resource>
      <ResourceMatch MatchId="&xacml;function:string-equal">
        <AttributeValue DataType="&xs;string">1:1437</AttributeValue>
        <ResourceAttributeDesignator DataType="&xs;string"
          AttributeId="&prile;resource:1:assertion:1:value"/>
      </ResourceMatch>
    </Resource>
    </Resources>
    <Actions>
      <Action>
        <ActionMatch MatchId="&xacml;function:string-equal">
          <AttributeValue DataType="&xs;string">read</AttributeValue>
          <ActionAttributeDesignator AttributeId="&xacml;action:action-id"
            DataType="&xs;string"/>
        </ActionMatch>
      </Action>
    </Actions>
  </Target>
  <Rule Effect="Permit" RuleId="lmm:cluster:access">
    <Target/>
    <Condition>
      <Apply FunctionId="&xacml;function:integer-equal">
        <Apply FunctionId="&prile;function:cluster">
          <Apply FunctionId="&xacml;function:double-one-and-only">
            <ResourceAttributeDesignator
              AttributeId="&prile;resource:1:assertion:1:value"
              DataType="&xs;double"/>
          </Apply>
          <AttributeValue DataType="urn:ogc:def:dataType:geoxacml:1.0:geometry">
            <gml:MultiPoint srsName="&prile;LaplacianMixtureModel">
              <gml:pointMember>
                <gml:Point>
                  <gml:coordinates>11.7,2.9,0.46</gml:coordinates>
                </gml:Point>
              </gml:pointMember>
              <gml:pointMember>
                <gml:Point>
                  <gml:coordinates>13.7,2.5,0.54</gml:coordinates>
                </gml:Point>
              </gml:pointMember>
            </gml:MultiPoint>
          </AttributeValue>
        </Apply>
        <AttributeValue DataType="&xs;integer">2</AttributeValue>
      </Apply>
    </Condition>
  </Rule>
```

Figure 8.5.5: Example XACML Policy for IDS rule with SID 1:1437 (part 1 of 2).

```
  <Obligations>
    <Obligation FulfillOn="Permit"
     ObligationId="&prile;resource:1:restrictions">
      <AttributeAssignment AttributeId="&prile;resource:1:policy:declassify"
        DataType="urn:ogc:def:dataType:geoxacml:1.0:geometry">
        <gml:MultiPoint srsName="http://www.prile.org/ByteRanges">
          <gml:pointMember>
            <gml:Point>
              <gml:coordinates>0,230</gml:coordinates>
            </gml:Point>
          </gml:pointMember>
        </gml:MultiPoint>
      </AttributeAssignment>
      <AttributeAssignment AttributeId="&prile;resource:1:cache-timeout"
        DataType="&w3c;TR/2002/WD-xquery-operators-20020816#dayTimeDuration">
        P1D</AttributeAssignment>
    </Obligation>
  </Obligations>
</Policy>
```

Figure 8.5.6: Example XACML Policy for IDS rule with SID 1:1437 (part 2 of 2).

Bibliography

[1] NIST Special Publication 800-56A. Recommendation for pairwise key establishment schemes using discrete logarithm cryptography. `http://csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf`, 2007.

[2] Alessandro Acquisti, Leslie John, and George Loewenstein. What is privacy worth? `http://www.futureofprivacy.org/wp-content/uploads/2010/07/privacy-worth-acquisti-FPF.pdf`, 2010.

[3] Alessandro Acquisti, Sabrina di Vimercati, Stefanos Gritzalis, and Costos Lambrinoudakis. *Digital Privacy*. Auerbach Publications, Boston, MA, USA, 2007.

[4] P. Akritidis, E. P. Markatos, M. Polychronakis, and K. Anagnostakis. STRIDE: polymorphic sled detection through instruction sequence analysis. In *In 20th IFIP International Information Security Conference*, 2005.

[5] A. Alharby and H. Imai. IDS false alarm reduction using continuous and discontinuous patterns. *Lecture Notes in Computer Science*, 3531:192–205, 2005.

[6] Andreas Matheus (ed). OGC 07-026r2 Geospatial eXtensible Access Control Markup Language (GeoXACML) version 1.0. `http://portal.opengeospatial.org/files/?artifact_id=25218`, 2007.

[7] C.A. Ardagna, M. Cremonini, S. De Capitani di Vimercati, and P. Samarati. A privacy-aware access control system. *Journal of Computer Security*, 16(4):369–397, January 2008.

[8] Claudio Agostino Ardagna, Sabrina De Capitani di Vimercati, Stefano Paraboschi, Eros Pedrini, and Pierangela Samarati. An XACML-based privacy-centered access control system. In *Proceedings of the first ACM workshop on Information security governance*, pages 49–58, Chicago, Illinois, USA, 2009. ACM.

[9] Ashmore, W. C. et al. Impact of alleged russian cyber attacks. *Baltic Security & Defence Review*, 11, 2009.

[10] B. Claise (ed). Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information. `http://www.ietf.org/rfc/rfc5101.txt`, 2008.

[11] B. Feinstein, G. Matthews. The intrusion detection exchange protocol (IDXP). `http://www.ietf.org/rfc/rfc4767.txt`, 2007.

[12] T L Bailey and C Elkan. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, 2:28–36, 1994. PMID: 7584402.

[13] Michael Baker, David Turnbull, and Gerald Kaszuba. Finding needles in haystacks (the size of countries). `http://media.blackhat.com/bh-eu-12/Baker/bh-eu-12-Baker-Needles_Haystacks-WP.pdf`, 2012.

[14] J. Bentham. Panopticon. or, the inspection-house, &c. *Criminological perspectives: essential readings*, page 25, 2003.

[15] Jeremy Bentham. *The Works of Jeremy Bentham.* Edinburgh: W. Tait; London, Simpkin, Marshall, 1843.

[16] Stefan Berthold and Rainer Böhme. Valuating privacy with option pricing theory. In Tyler Moore, David Pym, and Christos Ioannidis, editors, *Economics of Information Security and Privacy*, pages 187–209. Springer US, January 2010.

[17] M. Bezzi. An entropy based method for measuring anonymity. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 28–32, 2008.

[18] Paul Bicknell and Hung Jean. National information assurance partnership common criteria evaluation and validation

scheme, validation report hp tippingpoint intrusion prevention systems. http://www.commoncriteriaportal.org/files/epfiles/st_vid10345-vr.pdf, 2011.

[19] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. Content and popularity analysis of Tor hidden services. arXiv e-print 1308.6768, University of Luxembourg, August 2013.

[20] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Clustering via concave minimization. In *Advances in Neural Information Processing Systems -9*, page 368–374. MIT Press, 1997.

[21] Ian Brown and Douwe Korff. Terrorism and the proportionality of internet surveillance. *European Journal of Criminology*, 6(2):119–134, January 2009.

[22] Roland Büsckes and Dogan Kesdogan. Privacy enhanced intrusion detection. In *Multilateral Security for Global Communication - Technology, Application, Business*. Addison-Wesley-Longman, 1999.

[23] C. Powers, M. Schunter (ed). Enterprise privacy authorization language (EPAL 1.2). http://www.zurich.ibm.com/security/enterprise-privacy/epal/Specification/index.html, 2003.

[24] A. Cavoukian. Whole Body Imaging in Airport Scanners: Activate Privacy Filters to Achieve Security and Privacy. http://www.ipc.on.ca/images/Resources/wholebodyimaging.pdf, 2009.

[25] Ann Cavoukian, Scott Taylor, and Martin E. Abrams. Privacy by design - essential for organizational accountability and strong business practices. *Identity in the Information Society*, 3(2):405–413, 2010.

[26] Darren Charters. Electronic monitoring and privacy issues in Business-Marketing: the ethics of the DoubleClick experience. *Journal of Business Ethics*, 35(4):243–254, February 2002.

[27] V. Ciriani, S. Capitani di Vimercati, S. Foresti, and P. Samarati. k-Anonymity. In *Secure Data Management in Decentralized Systems*, pages 323–353. Springer, 2007.

[28] Cisco. Writing custom signatures for the cisco intrusion prevention system. http://www.cisco.com/web/about/security/intelligence/ips_custom_sigs_pdf.pdf, Accessed 2013.

[29] Sebastian Clauß and Stefan Schiffner. Structuring anonymity metrics. In *Proceedings of the second ACM workshop on Digital identity management*, pages 55–62, Alexandria, Virginia, USA, 2006. ACM.

[30] Julie Connolly, Mark Davidson, Matt Richard, and Clem Skorupka. The trusted automated eXchange of indicator information (TAXII). http://taxii.mitre.org/about/documents/Introduction_to_TAXII_White_Paper_November_2012.pdf, 2012.

[31] Aurélien Cord, Christophe Ambroise, and Jean-Pierre Cocquerez. Feature selection in robust clustering based on laplace mixture. *Pattern Recognition Letters*, 27(6):627–635, April 2006.

[32] Intel Corporation. *Intel SSE4 Programming Reference*. Intel, 2007.

[33] Cresbo, Gómez, Candau, and Mañas. MAGERIT - version 2 methodology for information systems risk analysis and management book i - the method. Technical report, Ministerio de administraciones públicas, Madrid, 2006.

[34] CrySys lab. Duqu: A stuxnet-like malware found in the wild, technical report. http://www.crysys.hu/publications/files/bencsathPBF11duqu.pdf, 2011.

[35] Ernesto Damiani, Pierangela Samarati, Sabrina De Capitani di Vimercati, and Stefano Paraboschi. Controlling access to XML documents. *IEEE Internet Computing*, 5:18–28, November 2001.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, page 1–38, 1977.

[37] Sabrina De Capitani di Vimercati and Pierangela Samarati. Privacy in the electronic society. In *Information Systems Security*, pages 1–21. Springer, 2006.

[38] Wen Ding, William Yurcik, and Xiaoxin Yin. Outsourcing internet security: Economic analysis of incentives for managed security service providers. In *Internet and Network Economics*, volume 3828 of *LNCS*, pages 947–958. Springer, 2005.

[39] D. Dolev and A. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198 – 208, March 1983.

[40] C. Dwork. Differential privacy. *Automata, languages and programming*, page 1–12, 2006.

[41] C. Dwork. Differential privacy: A survey of results. *Theory and Applications of Models of Computation*, page 1–19, 2008.

[42] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, page 371–380, 2009.

[43] J. Greene E. Karnin and M. Hellman. On secret sharing system. *IEEE Trans. on Info. Theory*, IT-29:35–41, January 1983.

[44] Donald Eastlake et al. XML Encryption Syntax and Processing. `http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/Overview.html`, 2002.

[45] Donald Eastlake et al. XML Signature Syntax and Processing Version 1.1, W3C Candidate Recommendation. `http://www.w3.org/TR/2011/CR-xmldsig-core1-20110303/#sec-XML-1`, 2011.

[46] European Commission. Directive 2002/58/EC of the European Parliament and of the Council of 12 july 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32002L0058:en:NOT`, 2002.

[47] European Commission. Directive 2006/24/EC of the European Parliament and of the Council of 15 march 2006 on the retention of data generated or processed in connection with the provision of publicly available electronic communications services

or of public communications networks and amending Directive 2002/58/EC. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:105:0054:01:EN:HTML, 2006.

[48] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO' 93*, volume 773, page 480–491. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.

[49] Mario A. T. Figueiredo and Anil K. Jain. Unsupervised learning of finite mixture models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(3):381–396, March 2002.

[50] Simone Fischer-Hübner. *IDA - An Intrusion Detection and Avoidance System (in German).* Aachen, Shaker, 1997.

[51] Ulrich Flegel. *Privacy-Respecting Intrusion Detection.* Springer, 1 edition, October 2007.

[52] Marc Fossi, Eric Johnson, Trevor Mack, Dean Turner, Joseph Blackbird, Mo King Low, Teo Adams, et al. *Symantec Global Internet Security Threat Report. Trends for 2008*, volume XIV. Symantec, 2008.

[53] W. N. Francis and H. Kučera. A standard corpus of present-day edited American English, for use with digital computers (Brown). Brown university. providence, Rhode Island, 1964, 1971, 1979.

[54] Felix C. Freiling, Thorsten Holz, and Georg Wicherski. Botnet tracking: exploring a root-cause methodology to prevent distributed denial-of-service attacks. In *Proceedings of the 10th European conference on Research in Computer Security*, ESORICS'05, pages 319–335, Berlin, Heidelberg, 2005. Springer-Verlag.

[55] G. Bianchi et al. Towards privacy-preserving network monitoring: Issues and challenges. In *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, pages 1–5, 2007.

[56] Gartner. Magic quadrant for network intrusion prevention systems, 2010.

[57] Lawrence A. Gordon and Martin P. Loeb. The economics of information security investment. *ACM Trans. Inf. Syst. Secur.*, 5(4):438–457, 2002.

[58] Jean Goubault-larrecq and Julien Olivain. *Detecting Subverted Cryptographic Protocols by Entropy Checking*. Laboratoire Spécification et Vérification, ENS Cachan, France, 2006.

[59] Andy Greenberg. Shopping for zero-days: A price list for hacker's secret software. `http://www.forbes.com/sites/andygreenberg/2012/03/23/shopping-for-zero-days-an-price-list-for-hackers-secret-software-exploits/`, 2012.

[60] S. Gritzalis, A. Yannacopoulos, C. Lambrinoudakis, P. Hatzopoulos, and S. Katsikas. A probabilistic model for optimal insurance contracts against security risks and privacy violation in IT outsourcing environments. *International Journal of Information Security*, 6(4):197–211, July 2007.

[61] Guofei Gu, Prahlad Fogla, David Dagon, Wenke Lee, and Boris Skorić. Measuring intrusion detection capability: an information-theoretic approach. In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, ASIACCS '06, page 90–101, New York, NY, USA, 2006. ACM.

[62] H. Debar, D. Curry, B. Feinstein. The intrusion detection message exchange format (IDMEF). `http://www.ietf.org/rfc/rfc4765.txt`, 2007.

[63] Cheng-Yuan Ho, Yuan-Cheng Lai, I.-Wei Chen, Fu-Yu Wang, and Wei-Hsuan Tai. Statistical analysis of false positives and false negatives from real traffic with intrusion detection/prevention systems. *IEEE Communications Magazine*, 50(3):146–154, 2012.

[64] Thomas Holz. An efficient distributed intrusion detection scheme. In *COMPSAC Workshops*, pages 39–40, 2004.

[65] G. Hsieh, R. Meeks, and L. Marvel. Supporting secure embedded access control policy with XACML+XML security. In *2010 5th Inter-*

national Conference on Future Information Technology (FutureTech), page 1 –6, May 2010.

[66] INET Research Group, TU-Berlin. Time machine. `http://www.net.t-labs.tu-berlin.de/research/tm/`.

[67] Yücel Karabulut Joachim Biskup. A hybrid PKI model with an application for secure mediation. *In 16th Annual IFIP WG 11.3 Working Conference on Data and Application Security*, pages 271–282, 2002. Published: Journal.

[68] C. Jordan, J. ROYES, and J. WHYTE. Writing detection signatures. *USENIX; login*, 30(6):55–61, 2005.

[69] K. M. Moriarty, B. H. Trammell. IODEF/RID over SOAP. `http://www.ietf.org/internet-drafts/draft-moriarty-post-inch-rid-soap-05.txt`, 2008.

[70] Sudhanshu Kairab. *A practical guide to security assessments*. Auerbach Publications, Boca Raton, Fla., 2005.

[71] Christos Kalloniatis, Evangelia Kavakli, and Stefanos Gritzalis. Addressing privacy requirements in system design: the PriS method. *Requir. Eng.*, 13(3):241–255, August 2008.

[72] Neil M. Richards & Jonathan H. King. Three paradoxes of big data. *Stanford Law Review Online*, 66:41, September 2013.

[73] O. Kolesnikov and W. Lee. Advanced polymorphic worms: Evading IDS by blending in with normal traffic. Technical report, Georgia Institute of Technology, 2005.

[74] Stefan Köpsell and Petr Švenda. Secure logging of retained data for an anonymity service. In Michele Bezzi, Penny Duquenoy, Simone Fischer-Hübner, Marit Hansen, and Ge Zhang, editors, *Privacy and Identity Management for Life*, volume 320, pages 284–298. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[75] Nir Kshetri. The simple economics of cybercrimes. *IEEE Security and Privacy*, 4(1):33–39, 2006.

[76] Romain Laborde and Thierry Desprats. An extension of XACML to improve the performance of decision making processes when dealing with stable conditions. In Latifa Boursas, Mark Carlson, Wolfgang Hommel, Michelle Sibilla, and Kes Wold, editors, *Systems and Virtualization Management. Standards and New Technologies*, volume 18 of *Communications in Computer and Information Science*, pages 13–24. Springer Berlin Heidelberg, 2008.

[77] Bo Lang, I. Foster, F. Siebenlist, R. Ananthakrishnan, and T. Freeman. A multipolicy authorization framework for grid security. In *Fifth IEEE International Symposium on Network Computing and Applications, 2006. NCA 2006*, pages 269–272, 2006.

[78] Lawrence Berkeley National Laboratory. Bro intrusion detection system. `http://bro-ids.org`.

[79] John O. Ledyard. incentive compatibility. In Steven N. Durlauf and Lawrence E. Blume, editors, *The New Palgrave Dictionary of Economics*, pages 158–164. Nature Publishing Group, Basingstoke, 2 edition, 2008.

[80] Alex X Liu, Fei Chen, Jeehyun Hwang, and Tao Xie. T.: XEngine: a fast and scalable XACML policy evaluation engine. *Conference on Measurement and Modeling of Computer Systems*, 2008.

[81] LOBSTER Consortium. Large-scale monitoring of broadband internet infrastructures. `http://www.ist-lobster.org`, 2007.

[82] Emilie Lundin and Erland Jonsson. Anomaly-based intrusion detection: privacy concerns and other problems. *Comput. Netw.*, 34(4):623–640, 2000.

[83] M. Marchiori (ed). The platform for privacy preferences 1.0 specification. `http://www.w3.org/TR/P3P`, 2002.

[84] Sourcefire, inc. M. Roesch, C. Green. Snort, 2009.

[85] J. Giffin M. Sharif, A. Lanzi and W. Lee. Impeding malware analysis using conditional code obfuscation. *NDSS'08*, 2008.

[86] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *Cornell University*, page 52, March 2007.

[87] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14, 1967.

[88] Gregor Maier, Robin Sommer, Holger Dreger, Anja Feldmann, Vern Paxson, and Fabian S. Schneider. Enriching network security analysis with time travel. *SIGCOMM Comput. Commun. Rev.*, 38(4):183–194, 2008.

[89] McAfee. Mcafee intrushield IPS, user-defined signature creation version 4.1. `https://kc.mcafee.com/resources/sites/MCAFEE/content/live/PRODUCT_DOCUMENTATION/20000/PD20345/en_US/INTR_User-Defined_Signatures_4.1.pdf`, 2007.

[90] John Stuart Mill. *Utilitarianism.* London: Parker, Son, and Bourn, 1863.

[91] MIT Kerberos Team. Kerberos: The network authentication protocol. `http://web.mit.edu/Kerberos`, accessed 2009.

[92] R. D. Moen, T. W. Nolan, and L. P. Provost. *Quality improvement through planned experimentation.* McGraw-Hill New York, 1999.

[93] N. Ulltveit-Moe. Two-tier XACML policy for IDS. `http://u-moe.org/twotierids/policy.xml`, 2009.

[94] Gunter Ollmann. The evolution of commercial malware development kits and colour-by-numbers custom malware. *Computer Fraud & Security*, 2008(9):4–7, September 2008.

[95] Michael Onken, Jörg Riesmeier, Marcel Engel, Adem Yabanci, Bernhard Zabel, and Stefan Després. Reversible anonymization of DICOM images using automatically generated policies. *Studies in health technology and informatics*, 150:861–865, 2009. PMID: 19745435.

[96] G. Orwell. *Nineteen eighty-four*. Secker and Warburg, 1949.

[97] Lasse Øverlier, Tønnes Brekne, and André Årnes. Non-expanding transaction specific pseudonymization for IP traffic monitoring. In *Cryptology and Network Security*, number 3810 in LNCS, pages 261–273. Springer, CANS, 2005.

[98] Ruoming Pang and Vern Paxson. A high-level programming environment for packet trace anonymization and transformation. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 339–351, Karlsruhe, Germany, 2003. ACM.

[99] Hyun-A Park, Dong Hoon Lee, Jongin Lim, and Sang Hyun Cho. PPIDS: privacy preserving intrusion detection system. In Christopher C. Yang, Daniel Zeng, Michael Chau, Kuiyu Chang, Qing Yang, Xueqi Cheng, Jue Wang, Fei-Yue Wang, and Hsinchun Chen, editors, *Intelligence and Security Informatics*, volume 4430, page 269–274. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[100] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[101] Panos Periorellis. *Securing Web Services*. Idi Global, October 2007.

[102] Michalis Polychronakis, Kostas G. Anagnostakis, and Evangelos P. Markatos. An empirical study of real-world polymorphic code injection attacks. In *Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more*, LEET'09, pages 9–9, Berkeley, CA, USA, 2009. USENIX Association.

[103] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. CryptDB: protecting confidentiality with encrypted

query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, page 85–100, New York, NY, USA, 2011. ACM.

[104] R. Büschkes, D. Kesdogan. Privacy enhanced intrusion detection. In G. Müller and K. Rannenberg, editors, *Multilateral Security in Communications, Information Security*, pages 187–204. Addison Wesley, 1999.

[105] R. Danyliw, J. Meijer, Y. Demchenko. The incident object description exchange format. `http://www.rfc-editor.org/rfc/rfc5070.txt`, 2007.

[106] Jaziar Radianti and Nils Ulltveit-Moe. Classification of malicious tools in underground markets for vulnerabilities. *NISK 2008*, pages 19–31, 2008.

[107] A. Rényi. On measures of information and entropy. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960*, pages 547–561, 1961.

[108] Miriam Ronzoni. Teleology, deontology, and the priority of the right: On some unappreciated distinctions. *Ethical Theory and Moral Practice*, 2009.

[109] Judith E. Y. Rossebø, Scott Cadzow, and Paul Sijben. eTVRA, a threat, vulnerability and risk assessment method and tool for eEurope. In *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*, pages 925 –933, April 2007.

[110] S. Cantor et al. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, 2005.

[111] Pierangela Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13:1010–1027, 2001.

[112] Pamela Samuelson. Privacy as intellectual property? *Stanford Law Review*, 52(5):1125–1173, 2000.

[113] L. Sankar, S.R. Rajagopalan, and H.V. Poor. Utility and privacy of data sources: Can shannon help conceal and reveal information? *Information Theory and Applications Workshop (ITA), 2010*, pages 1 –7, 2010.

[114] Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (IDPS). `http://csrc.nist.gov/publications/nistir/ir7628/nistir-7628_vol1.pdf`, 2007.

[115] Stuart E Schechter and Michael D Smith. How much security is enough to stop a thief? the economics of outsider theft via computer systems and networks. *Financial Cryptography*, 2742:122–137, 2003.

[116] Friedrich Schmid and Axel Schmidt. Nonparametric estimation of the coefficient of overlapping—theory and empirical application. *Computational Statistics & Data Analysis*, 50(6):1583–1596, March 2006.

[117] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Privacy Enhancing Technologies*, page 259–263, 2003.

[118] Claude Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[119] Klaus Brunnstein Simone Fischer-Hübner. Combining verified and adaptive system components towards more secure computer architectures security and persistence. 1990.

[120] A. Slagell and W. Yurcik. Sharing computer network logs for security and privacy: A motivation for new methodologies of anonymization. In *Security and Privacy for Emerging Areas in Communication Networks, 2005. Workshop of the 1st International Conference on*, page 80–89, 2005.

[121] G. Smith. Quantifying information flow using min-entropy. In *Quantitative Evaluation of Systems (QEST), 2011 Eighth International Conference on*, page 159 –167, September 2011.

[122] Geoffrey Smith. On the foundations of quantitative information flow. In Luca de Alfaro, editor, *Foundations of Software Science and Computational Structures*, number 5504 in Lecture Notes in Computer Science, pages 288–302. Springer Berlin Heidelberg, January 2009.

[123] Michael Sobirey, Simone Fischer-Hübner, and Kai Rannenberg. Pseudonymous audit for privacy enhanced intrusion detection. In *Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC'97)*, pages 151–163, May 1997.

[124] Michael Sobirey, Birk Richter, and Hartmut König. The intrusion detection system AID - architecture and experiences in automated audit trail analysis. In *Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security*, pages 278–290, 1996.

[125] Y. Song, M. E Locasto, A. Stavrou, A. D Keromytis, and S. J Stolfo. On the infeasibility of modeling polymorphic shellcode. In *Proceedings of the 14th ACM conference on Computer and communications security*, page 541–551, 2007.

[126] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10:557–570, 2002.

[127] T. Moses. OASIS eXtensible Access Control Markup Language (XACML) Version 2.0. `http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf`, 2005.

[128] PreludeIDS Technologies. Prelude ids. `http://www.prelude-ids.com`, accessed 2009.

[129] Tenable Network Security. Nessus. `http://www.nessus.org`, accessed 2009.

[130] The MITRE Corporation. Standardising cyber threat intelligence information with the structured threat information eXpression (STIX). `http://makingsecuritymeasurable.mitre.org/docs/STIX-Whitepaper.pdf`, 2012.

[131] G. Tóth, Z. Hornák, and F. Vajda. Measuring anonymity revisited. In *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, 2004.

[132] Nils Ulltveit-Moe, Terje Gjøsæter, Sigurd M. Assev, Geir M. Køien, and Vladimir Oleshchuk. Privacy handling for critical information infrastructures. In *Industrial Informatics (INDIN), 2013 11th IEEE International Conference on*, pages 688–694, 2013.

[133] Nils Ulltveit-Moe and Vladimir Oleshchuk. Two tiered privacy enhanced intrusion detection system architecture. In *2009 IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pages 8–14, Rende, Italy, 2009.

[134] Nils Ulltveit-Moe and Vladimir Oleshchuk. Privacy leakage methodology (PRILE) for IDS rules. In Michele Bezzi, Penny Duquenoy, Simone Fischer-Hübner, Marit Hansen, and Ge Zhang, editors, *Privacy and Identity Management for Life*, volume 320 of *IFIP Advances in Information and Communication Technology*, pages 213–225. Springer Berlin Heidelberg, 2010.

[135] Nils Ulltveit-Moe and Vladimir Oleshchuk. Decision-cache based XACML authorisation and anonymisation for XML documents. *Comput. Stand. Interfaces*, 34(6):527–534, November 2012.

[136] Nils Ulltveit-Moe and Vladimir Oleshchuk. Mobile security with location-aware role-based access control. In Ramjee Prasad, Károly Farkas, Andreas U. Schmidt, Antonio Lioy, Giovanni Russello, Flaminia L. Luccio, Ozgur Akan, Paolo Bellavista, Jiannong Cao, Falko Dressler, Domenico Ferrari, Mario Gerla, Hisashi Kobayashi, Sergio Palazzo, Sartaj Sahni, Xuemin (Sherman) Shen, Mircea Stan, Jia Xiaohua, Albert Zomaya, and Geoffrey Coulson, editors, *Security and Privacy in Mobile Information and Communication Systems*, volume 94 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, page 172–183. Springer Berlin Heidelberg, 2012.

[137] Nils Ulltveit-Moe and Vladimir A. Oleshchuk. A composite privacy leakage indicator. *Wireless Personal Communications*, 61(3):511–526, August 2011.

[138] Nils Ulltveit-Moe and Vladimir A. Oleshchuk. Measuring privacy leakage for IDS rules. *CoRR*, abs/1308.5421, 2013.

[139] United Nations. The universal declaration of human rights. `http://www.un.org/en/documents/udhr/`, 1948.

[140] M. Vallentin. On the evolution of buffer overflows. `http://matthias.vallentin.net/course-work/buffer_overflows.pdf`, 2007.

[141] Dawid Nowak Vanessa Ayala-Rivera. Protecting organizational data confidentiality in the cloud using a high-performance anonymization engine. *In press*, 2013.

[142] C. S. Wallace and D. M. Boulton. An information measure for classification. *The Computer Journal*, 11(2):185–194, January 1968.

[143] Greg Walton. China's golden shield - corporations and the development of surveillance technology in the people's republic of China. `http://www.ichrdd.ca/site/_PDF/publications/globalization/CGS_ENG.PDF`, 2001.

[144] Samuel Warren and Louis D. Brandeis. The right to privacy. *Harvard Law Review*, 4(5), 1890.

[145] Stefan Weber and Max Mühlhäuser. Multilaterally secure ubiquitous auditing. In Santi Caballé, Fatos Xhafa, and Ajith Abraham, editors, *Intelligent Networking, Collaborative Systems and Applications*, volume 329 of *Studies in Computational Intelligence*, page 207–233. Springer Berlin / Heidelberg, 2011.

[146] Stefan G. Weber. Harnessing pseudonyms with implicit attributes for privacy-respecting mission log analysis. In *Intelligent Networking and Collaborative Systems, 2009. INCOS '09. International Conference on*, page 119–126. IEEE, November 2009.

[147] Murray S. Weitzman. *Measures of Overlap of Income Distributions of White and Negro Families in the United States*. U.S. Bureau of the Census, 1970.

[148] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting TF-IDF term weights as making relevance decisions. *ACM Trans. Inf. Syst.*, 26(3):13:1–13:37, June 2008.

[149] IBM Security Systems X-Force. Signature author's guide, IBM security systems opensignature. `http://www-01.ibm.com/support/docview.wss?uid=swg21570487&aid=3`, 2011.

[150] Athanassios N. Yannacopoulos, Costas Lambrinoudakis, Stefanos Gritzalis, Stylianos Z. Xanthopoulos, and Sokratis N. Katsikas. Modeling privacy insurance contracts and their utilization in risk management for ICT firms. In *Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security*, pages 207–222, Málaga, Spain, 2008. Springer-Verlag.

[151] A. Ziviani, A.T.A. Gomes, M.L. Monsores, and P.S.S. Rodrigues. Network anomaly detection using nonextensive entropy. *IEEE Communications Letters*, 11(12):1034 –1036, December 2007.

Index