Sikkert valgsystem over Internett

Hovedoppgave
ved
sivilingeniørutdanning i
informasjons- og kommunikasjonsteknologi

av
Magne Hopland
Rune Jensen

Grimstad, mai 2001

Sammendrag

Det har i de siste årene vært en økende interesse for å utvikle sikre valgsystem over Internett. Allerede i 1869 fikk Thomas Edison patent på et elektronisk valg apparat, men det var derimot ingen den gang som ville kjøpe oppfinnelsen hans.

Et elektronisk valgsystem må oppfylle en del sikkerhets krav før det kan implementeres i en valgsammenheng.

Hovedoppgaven ser på en del grunnleggende ideer for valgprotokoller, og på valgprotokoller som er brukt i eksisterende system i dag.

Et ideelt valgsystem inneholder følgende karakteristiske trekk: Anonymitet, Demokrati, Fleksibilitet, Kontrollerbarhet, Mobilitet, Nøyaktighet, Tilretteleggelse.

Det er utviklet en prototyp som har som oppgave å opprettholder velger sin anonymitet, fra han stemmer, til opptelling.

Prototypen opprettholder velgeren sin anonymitet, forutsatt at de forskjellige valgautoriteter er adskilt fra hverandre under valgprosessen, ved hjelp av RSA krypteringsalgoritmen. En del begrensninger gjør at systemet må forbedres for å implementere det i en valgsammenheng. Det vil blant annet måtte stilles krav til bedre nøyaktighet ved opptelling av stemmer for å kontrollere at resultatet blir talt opp riktig.

Flere forslag er satt opp for å bedre prototypen slik at den kan brukes i et valgsystem.

Forord

Som et ledd i den avsluttende delen av sivilingeniørstudiet ved Høgskolen i Agder, ble hovedoppgaven *Sikkert valgsystem over Internett* utarbeidet.

Oppgaven gir en analyse og oversikt over noen eksisterende elektroniske valgsystem og metoder som allerede finnes, og ser på sikkerhetsaspekter som blant annet sikrer brukers anonymitet. Det er på dette grunnlag utarbeidet en prototyp.

Arbeidet med hovedoppgaven begynte i februar og avsluttet i mai 2001.

Vi vil takke Vladimir Oleshchuk og Magne Arild Haglund som var veilederne våre ved Høgskolen i Agder.

Grimstad, 28.mai 2001

Magne Hopland
Rune Jensen

Innholdsfortegnelse

Figuroversikt

1. Innledning

Ettersom flere får tilgang til Internett, vil det være en økende interesse for å foreta valg prosedyrer elektronisk. Elektroniske valg er billigere og mindre tidkrevende å administrere enn manuelle valg. Det at det blir enklere å velge kan også være en bidraende faktor til å få økt velger oppslutning ved valg. Det er mye enklere å si din mening med et par klikk med musetasten enn å følge de tradisjonelle prosedyrene ved et manuelt valg, der en ofte må reise over lengre strekninger for å avgi sin stemme ved valglokalet.

Sikkerhets aspektene omkring elektronisk valg over Internett er mange. Avstemninger og undersøkelser, er system som har hatt en ganske lang levetid på Internett på grunn av at sikkerhetskravene for slike system er forholdsvis ordinære. Disse systemene er imidlertid ikke laget for de strenge sikkerhetskrav som er nødvendig for valg i offentlig sektor.

Dersom et valgsystem ikke er ordentlig designet så kan det lett manipuleres på forskjellige måter, som er med på å forfalske resultat eller bryte velger sin anonymitet.

1.1 Definisjon av oppgaven og avgrensning

En del av oppgaven besto i å gjøre en undersøkelse av protokoller og eventuelle algoritmer som er i bruk i eksisterende system i dag eller tilgjengelige for bruk.
Det skulle gjøres en analyse av forskjellige applikasjonsområdene som benyttet seg av aktuelle sikkerhetskrav, som autentisering og kryptering, innenfor et valgt område og hvilke krav som blir stilt til et slikt system. Til slutt skulle det utarbeides et design av protokoller og algoritmer som skulle brukes til fremstilling av en prototyp.

Avgrensning:
Det er valgt å se nærmere på protokoller og krypteringsalgoritmer som tilbyr anonymitet for valgsystem over Internett. En analyse av forskjellige protokoller vil bli gjort.

1.2 Fremdrift av oppgaven

Arbeidet med hovedoppgaven har vært delt inn i flere trinn.
Det ble først gjort et litteraturstudie som gikk fra februar, da oppgaven ble satt, til slutten april. Dette innebar en undersøkelse av forskjellige sikkerhetsprotokoller, kryptografiske algoritmer og krav som stilles til et valgsystem og for Internett generelt.
Det ble underveis gjort en avgrensning og skriveprosessen begynte i slutten av april.

2 Sikkert valgsystem

2.1 Utgangspunkt

Det ble bestemt at det skulle lages en prototyp av et sikkert valgsystem for presentasjon under framvisningen av hovedoppgaven. For å få et sikkert valgsystem over Internett må det settes en del krav som bør oppfylles. Et strukturert design bidrar også til at systemet blir mindre utsatt for manipulasjon og feil.

2.2 Krav

Det viktigste kravet i et elektronisk valgsystem, er at bruker sin identitet ikke blir koblet opp mot hans stemmeseddel slik at han er anonym under valgprosessen. Velgeren må være autorisert til å velge, og kun få lov til å avlevere én stemme. Etter valget må man kunne kontrollere at stemmene blir telt opp på en riktig måte. Det må ikke være noen begrensning hvor velgeren avgir sin stemme.

2.3 Forklaring av struktur til et sikkert valgsystem

Et tenkt valgsystem inneholder fire aktører:

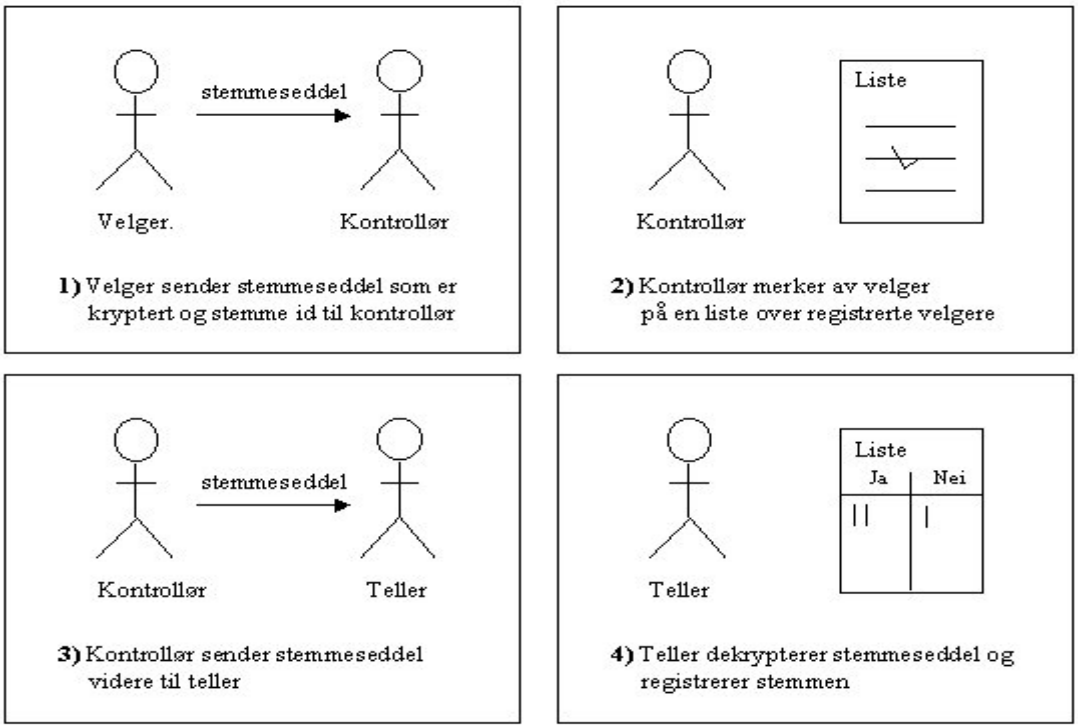
- Velger
- Registrator
- Kontrollør
- Teller

Velger:
For å få tilgang til et valg må bruker registrere seg som godkjent velger i god tid før valget tar til. Dette skjer ved å sende personlig informasjon i form av for eksempel navn og personnummer. Dersom en person oppfyller kravene til å få velge, får han tilsendt en stemmeId og personlig passord for valget. Når velgeren avgir sin stemme sendes en kryptert stemmeseddel sammen med stemmeId til *kontrolløren*.

Registrator:
Registrator står ansvarlig for å gi autorisasjon til de som ønsker å delta i valget. Dersom brukeren oppfyller kravene til å stemme, lager *Registrator* en stemmeId sammen med et personlig passord som sendes til bruker sin adresse. Når registreringsperioden er over, vil *Registrator* sende en liste over alle stemmeId til *kontrolløren*.

Kontrollør:
Kontrolløren er ansvarlig for å sjekke om velgeren er autorisert av *Registrator*. Når *kontrolløren* mottar stemmen fra en velger i form av stemmeId og kryptert stemmeseddel, vil han sjekke mot listen han mottok fra *Registrator* om velgeren finnes på listen over godkjente stemmeId. Han vil da se om velgeren har stemt tidligere eller ikke. Hvis en velger finnes på listen og ikke har valgt tidligere, vil *kontrolløren* signere den krypterte stemmeseddelen og sende den til teller uten stemmeId.

Teller:
Når *telleren* mottar en signert kryptert stemmeseddel fra *kontrolløren*, vil signeringen verifisere at stemmen er gyldig. *Telleren* dekrypterer stemmeseddelen og oppdaterer valgresultatet.



Figur 1 – Sikkert valgsystem

3 Litt generelt om sikkerhet

3.1 Hva legger vi i ordet datasikkerhet ?

Sikkerhet er definert som en tilstand utenfor fare, eller det å være sikker, føle seg trygg. I data sammenheng kan man si det er et begrep på hvordan man kan sikre at dataressursene ikke blir misbrukt eller ulovlig satt ut av funksjon.

På en annen måte kan vi si at datasikkerhet vil bestå av forskjellige tiltak, og en del sikkerhetsregler for å beskytte verdiene våre i et datasystem.

Man kan dele datasikkerhet inn i hovedsakelig tre deler; konfidensialitet, integritet, og tilgjengelighet.

Konfidensialitet: Vil si at informasjonen ikke skal være tilgjengelig for uautoriserte personer. Konfidensiell informasjon er hemmelig informasjon, noe som utenforstående personer ikke skal ha innsyn til. Eksempler inkluderer forsvarshemmeligheter, personopplysninger, krypteringsnøkler, bedriftshemmeligheter og industrihemmeligheter.

Integritet: Vil si at informasjonen ikke skal kunne ødelegges eller forandres av personer som ikke er autorisert til det.

Tilgjengelighet: Vil si at en tjeneste oppfyller bestemte krav til stabilitet slik at den informasjonen du er ute etter er tilgjengelig ved behov.

For å oppnå datasikkerhet benytter man seg gjerne av, autorisasjon, autentisering, kryptering, noe vi skal se litt nærmere på senere.

3.2 Autorisering

Autorisering er en avgjørelse om en person skal ha tilgang på ressurser eller informasjon, sikkerhetsreglene skal gi de nødvendige retningslinjene for hvorvidt en person skal få tilgang på de aktuelle ressursene eller informasjonen.

3.3 Autentisering

Autentisering er prosessen med å bekrefter en oppgitt identitet. Det finnes mange måter å få bekreftet en oppgitt identitet på, og vi kan dele opp disse i tre typer av autentisering av personer:

- Man benytter noe man vet; for eksempel passord, PIN, personlig informasjon.
- Token basert, man benytter noe man har; for eksempel id kort, dåps attest, smartkort.
- Biometri, man benytter noe man er; for eksempel fingeravtrykk, stemmegjenkjenning, iris, ansiktstrekk.

Videre så kan man dele inn autentiseringen med hvordan den blir gjennomført:

- Manuell; for eksempel over disk i banken.
- Elektronisk; for eksempel innlogging, minibank.
- Helautomatisk; for eksempel digitale signaturer.

Når man benytter seg av elektronisk autentisering regnes biometri som sterkere enn token basert som igjen er sterkere enn passord typen.

3.4 Kryptering

Moderne krypteringsteknikker har utviklet seg fra tidenes morgen. Kryptering er en samling av teknikker som forandrer data på måter som gjør de vanskelige å tyde.

Informasjon som du sender i fra deg kan ende opp i hendene på personer med skumle hensikter. Dersom de ønsker det kan de endre eller forfalske din informasjon, enten for fornøyelse eller slik at det gagner de bedre. Kryptografi kan omforme din informasjon slik at den ikke lenger er leselig for andre en de som informasjonen er beregnet på, og på den måten er sikrere i sin ferd i mellom datamaskinene.

Det finnes to hoved krypterings metoder. Den ene benytter seg av symmetriske krypteringsnøkler; den bruker samme nøkler til kryptering og dekryptering. Den andre metoden benytter seg av asymmetriske krypteringsnøkler. Den har en offentlig og en privat nøkkel, der den ene blir brukt til kryptering og den andre til dekryptering. Sender og mottaker må finne en måte å få etablert og delt en krypterings nøkkel. Dette kan gjøres på flere måter. Enten ved fysisk kontakt mellom partene der overlevering av nøklene skjer. En annen måte er å bruke en såkalt *tiltrodd tredjepart.* Dette vil være en uavhengig instans som alle parter stoler på og kan verifisere at brukeren er den han påstår han er. En type *tiltrodd tredjepart* er PKI; Public Key Infrastructure. Dette er en instans som kan bevise identiteten til personer som har krypteringsnøkler. PKI vil typisk bestå av en sertifiserings autoritet som er en instans som kan utstede offentlig nøkkel sertifikat.

Når Sender og mottaker bruker et krypterings system for å kryptere en melding, stoler de på at krypterings teknikken sikrer dem at meldingen deres er:

- De stoler på at meldingen deres er konfidensiell. At ingen andre kan lese deres melding, siden ingen andre enn de har krypteringsnøkkelen.
- De stoler på at meldingen kommer fra den personen en tror den kommer fra, siden ingen andre har krypteringsnøkkelen. En slipper å bekymre seg så masse for forfalskninger fordi en forfalsket melding skulle dekrypteres ufullstendig.
- De stoler på integriteten til meldingen. En forfalsker kan ikke vite hva meldingen sier og kan derfor ikke forandre innholdet i meldingen på en fornuftig måte. Tilfeldig forandring i meldingen vil til all formodning gjøre meldingen ulesbar.

Dersom dine data er viktige og det er en reel risk for at noen har interesse for å tukle med dem, trenger du å beskytte dine data ved hjelp av kryptering.

Digital signatur:
Digital signatur er en teknikk for å signere digital informasjon, som bygger på avanserte krypteringsteknikker. Bruker av digital signatur får tildelt to elektroniske krypteringsnøkler, en privat og en offentlig. Når avsender ønsker å sende et elektronisk dokument, sendes den sammen med dennes private nøkkel. Når mottaker får en slik melding, vil han i utgangspunktet kunne lese selve meldingen. For å verifisere at det er en ekte melding fra avsender, vil mottaker verifisere den mot avsenders offentlige nøkkel. Resultatet av denne prosessen vil være et godkjent/ikke godkjent utfall. Ved godkjente utfall vil mottaker være sikret meldingens integritet, autentisitet og ikke benekting (at sender ikke kan benekte å ha sendt dokumentet).

Hash funksjoner:
Hash funksjoner tillater et digitalt "fingeravtrykk" eller "hash verdi" å bli tatt fra en vilkårlig bitblokk. Denne blokken kan være en hvilken som helst elektronisk dokument. Når man så har et "fingeravtrykk" av et elektronisk dokument, vil "fingeravtrykket" forandres dersom dokumentet forandres. Forfatteren av dokumentet kan ta den genererte "hash verdien" og kryptere den med sin private nøkkel og på den måten generere en digital signatur. Denne digitale signaturen kan sendes sammen med dokumentet slik at mottakeren kan sjekke at dokumentet ikke har blitt forandret på; mottaker kan dekryptere signaturen med forfatterens offentlige nøkkel, og dermed avsløre den originale "hash verdien" til dokumentet. Så sammenligner han bare de to "hash verdiene", og dersom de er like vet han at dokumentet ikke er blitt forandret på.

Blind Signatur:
Blind signatur [18] gir muligheten til å få en melding signert uten å avsløre hva som signeres. David Chaum [12] som introduserte denne metoden demonstrerte implementasjonen ved å bruke RSA signaturer på denne måten: Sett at Alice har en melding m som hun ønsker å signert av Bob, og hun ikke ønsker at Bob skal vite hva m inneholder. La (n,e) være Bob sin offentlige nøkkel og (n,d) hans private nøkkel. Alice genererer en tilfeldig verdi r slik at $gcd(r,n) = 1$ og sender

$$m' = r^e\, m \bmod n$$

til Bob. Verdien m' er "blenda" av den tilfeldige verdien r, og dermed kan ikke Bob hente noen nyttig informasjon fra meldingen. Bob returnerer så den signerte verdien

$$s' = (m')^d = (r^e m)^d \bmod n$$

til Alice. Siden $s' = rmd \bmod n$, kan Alice få frem den virkelige signaturen s av m ved å rekne ut

$$s = s'\, r^{-1} \bmod n$$

Nå har Alice en signatur som hun ikke kunne ha skaffet seg på egenhånd.

3.5 Sikringstiltak

For å sikre våres ressurser og informasjon for utenforstående, må man ha visse sikrings tiltak. Foruten fysisk sikring som hindrer uautoriserte personer å komme i kontakt med informasjonen, må man også ha tilgangskontroll på nettverksdelen av systemet vårt. Passord er det mest grunnleggende sikringstiltak også i forbindelse med nettverkssikkerhet; det sies at 80% av alle hackere aldri ville fått tilgang til dataanleggene dersom passords rutinene hadde vert bedre[3].

Det er viktig at brukerne identifiserer seg med brukernavn og passord. Det bør ikke være mulig for andre å få tilgang på brukerens passord, dette krever at passordene blir lagret krypterte. Da sikrer man seg om at andre som er pålogget kan hente ut en annen brukers passord.

Tilgangskontroll til et system vil si å styre tilgangen til ressursene på en mest mulig fornuftig måte. Alle brukere av et system vil ikke trenge full tilgang til alle ressursene, man får tilgang etter ”need-to-know” prinsippet, man får tilgang til de ressursene som er relevante for at man kan få utført sine oppgaver på en mest hensynsmessig måte.

Formålet med et tilgangs kontroll system (TKS) er todelt. Det ene er å gjøre det umulig for uautoriserte personer å endre , lese eller fjerne data, det andre er å gjøre det lettere for autoriserte personer å få utført arbeidet sitt i et kontrollert miljø.
TKS består generelt av fire funksjoner:

- identitetskontroll(LOGONID)
- autentisering(pålitelighet)
- autorisasjon
- overvåkning og rapportering(logging)

Når vi skal sende informasjon i nettverket er kryptering det beste hjelpemiddelet for å hindre utenforstående å få tilgang på informasjonen som sendes. Det er også viktig at den enkelte bruker av systemet er klar over hvor viktig det er med sikkerhet. For uansett hvilke tiltak som iverksettes, vil resultatene av tiltakene i stor grad avhenge av den enkelte brukers sikkerhets forståelse. Det hjelper ikke at sikringstiltakene er gode dersom brukerne ikke bruker de riktig.

Sikker kanal:
Med sikker kanal menes en kanal som sikrer data ved overføring fra en klient til server eller omvendt. En slik kanal tilbyr en høyere grad av nettverkssikkerhet ved å tilby endepunkts autentisering, meldings kryptering og meldings autentisering. En sikker kanal implementeres over TCP/IP protokollene og under applikasjonsprotokoller som HTTP og FTP. Sikre kanaler tilbyr :

- anonymitet: data kan ikke bli undersøkt
- integritet: data kan ikke bli forfalsket.
- autentisering: gjør at klient og server krever at motpart identifiserer seg.

Secure socket layer (SSL) [6] og Transport layer security (TLS) er to standarder tilgjengelig i dag for å oppnå en sikker kanal.

4 Manuelt valgsystem

For å forklare gangen i et manuelt valgsystem er det tatt utgangspunkt i det norske systemet.

4.1 Valgprosessen

Prosessen ved valglokalet:

- autorisering
- stemmegivning
- avlevering av stemme

Autorisering:
Det første som skjer ved et valglokale er at man henvender seg til en valgfunksjonær som sitter med en manntallsliste og sjekker om du har lov til å velge.
Valgfunksjonæren krysser av i listen dersom det ikke er gjort fra før. Dersom dette er gjort betyr det at velgeren allerede har stemt, enten før på dagen eller ved forhåndsstemming.

Stemmegivning:
Dersom du er blitt autorisert til å stemme får du utlevert en konvolutt og blir vist til et stemmeavlukke. Her ligger stemmesedler for hvert parti, man velger det parti man ønsker å stemme på og legger stemmeseddelen i konvolutten.

Avlevering av stemme:
Etter å ha lagt stemmeseddelen i konvolutten, legges konvolutten i en valgurne.
Når valglokalet har stengt foretas en manuell opptelling av stemmene.

Forhåndsstemming:
Personer som ikke kan møte på valg dagen har mulighet til å forhåndsstemme fra 15.juli til og med siste fredag før valg dagen.
Ved forhåndsstemming kan en person avgi sin stemme via posten. Man møter opp hos fungerende valg funksjonær og får utdelt en konvolutt som stemmen legges i. Deretter legger valgfunksjonæren konvolutten med stemmen i en ny konvolutt der fullt navn og adresse blir påført. Forhåndstemmer blir krysset av på manntallsliste etter hvert som de kommer inn til valgstyret.

5 Fra manuelt til elektronisk valg

Målet med å foreta valg elektronisk i stedet for manuelt er at velgeren kan på en enklere måte avgi sin stemme. Fordelen med et elektronisk valg:

- billigere og mindre tidkrevende
- enklere prosedyrer kan medføre økt velger oppslutning
- slipper reisetid til valglokale
- mer presis
- hindrer valgjuks
- automatisk opptelling

Når et elektronisk valgsystem skal lages må de samme sikkerhetsrutinene opprettholdes som ved et manuelt valg. I tillegg må egne sikkerhetsrutiner for sikring av dataoverføring implementeres.

Det vil være naturlig å overføre de forskjellige funksjonalitetene ved et manuelt valg til elektroniske valg.

Valgfunksjonæren sin oppgave under manuelle valg er å sjekke om personer er autorisert til å velge. En måte å overføre denne funksjonaliteten til elektroniske valg på er å innføre en kontrollør som sjekker mot en liste over godkjente velgere og som ser om velgeren har valgt tidligere eller ikke.
For å opprettholde sin anonymitet ved manuelle valg legger velgeren sin stemmeseddel i en konvolutt. Ved elektroniske valg kan dette gjøres ved hjelp av kryptering. En automatisk tellefunksjon vil naturlig overføre den manuelle tellingen.

6 Elektronisk valgsystem

Flere og flere hverdagslige prosedyrer blir integrerte i dagens elektroniske verden. Det har i flere år blitt foretatt elektroniske transaksjoner, avstemninger og undersøkelser over Internett. Grunnen til at disse systemene er såpass utbredt er at det ikke stilles større sikkerhetskrav til dem.

Et valgsystem over Internett som skal operere på et seriøst og offentlig grunnlag, må ta hensyn til flere sikkerhets krav for å ivareta velgerens anonymitet og for å hindre valgjuks.

Før et valg kan gjennomføres må den enkelte velger registrere seg for å få tilgang til valget. Dette gjøres i en bestemt periode før valget og resulterer i at brukeren får tilsendt en bruker id og et passord.

Når en velger ønsker å avgi sin stemme, må administrator av valget kunne verifisere at velgeren har rett til å stemme ved å spørre om bruker id og eventuelt passord. Administrator må også til enhver tid kunne holde styr på om pålogget bruker har stemt tidligere eller ikke.
Etter at velgeren har avlagt sin stemme, må administrator samle inn stemmen for deretter sende den videre til en tellemekanisme som holder oversikt over resultatet.

Når man skal lage et elektronisk valgsystem over Internett, er det viktig å se på hvordan valget kan gjennomføres uten å offentliggjøre velgeren sin identitet eller muliggjør valgjuks. For å avgjøre om et system ivaretar disse oppgavene godt nok er det nyttig å fastsette en del kriterier for å avgjøre systemets utførelse:
Noen ønskelige karakteristiske trekk ved et valgsystem:

Anonymitet:
Et valgsystem støtter anonymitet dersom

- verken administrator eller noen andre kan knytte en stemmeseddel til velgeren som avla stemmen

Demokrati:
Et valgsystem er demokratisk dersom

- det tillater kun personer som er autorisert til å velge
- hver velger kun kan stemme en gang

Fleksibelt:
Et valgsystem er fleksibelt dersom det tillater forskjellige formater på spørsmålsformuleringen, ikke bare ja/nei spørsmål.

Kontrollerbart:
Et valgsystem er kontrollerbart dersom man kan kontrollere at alle stemmen er telt riktig.

Mobilitet:
Et valgsystem er mobilt dersom det er ingen restriksjoner på hvor velgeren avgir sin stemme. Et stort problem ved valg er hjemmesitterne. Dersom folk kan stemme fra sin PC eller en mobilterminal (WAP-telefon, PDA) ville valgdeltakelsen økt.

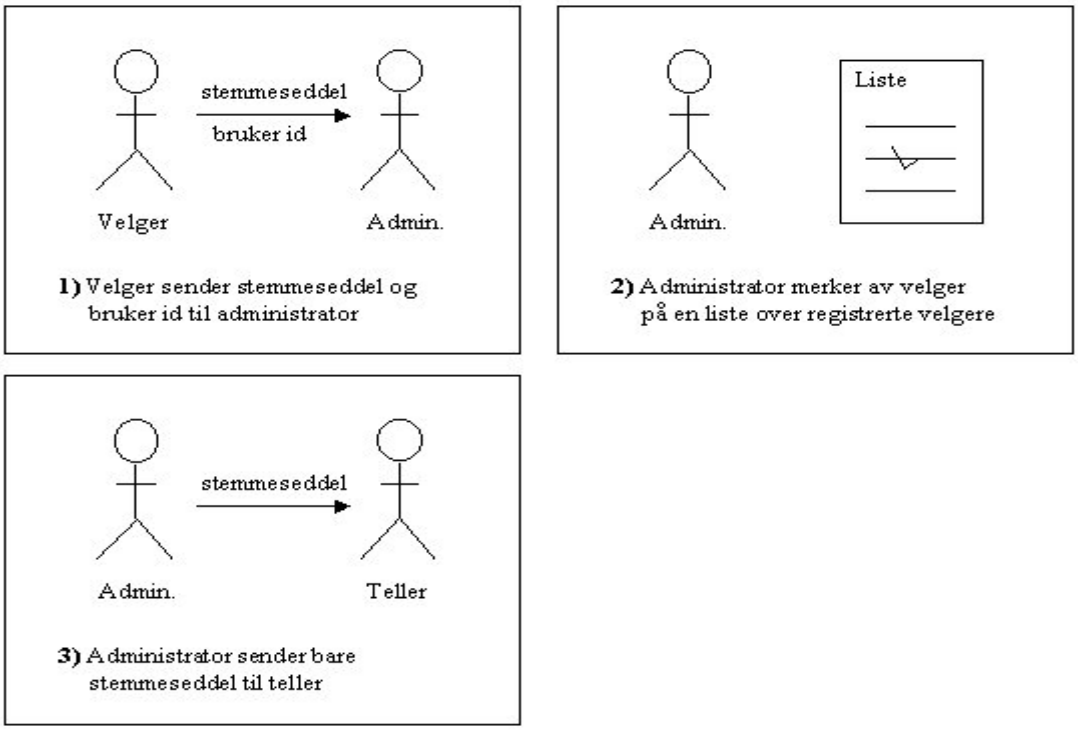
Nøyaktighet:
Et valgsystem er nøyaktig dersom det ikke er mulighet for å

- forandre på en stemme
- slette en stemme fra opptellingen
- telle en ugyldig stemme i opptellingen uten at det blir oppdaget.

Tilretteleggelse:
Et valgsystem er godt tilrettelagt dersom det tillater velgeren å gjennomføre valget raskt og enkelt uten noen spesielle ferdigheter eller utstyr.

6.1 Valgprotokoller

The simple protocol:
For å implementere de ovenstående karakteristiske trekk, kan man tenke seg å lage en enkel protokoll som krever at velgeren sender en elektronisk stemmeseddel til en administrator sammen med en bruker id. Deretter fjerner administrator bruker id og sender stemmeseddelen videre til en tellemekanisme.

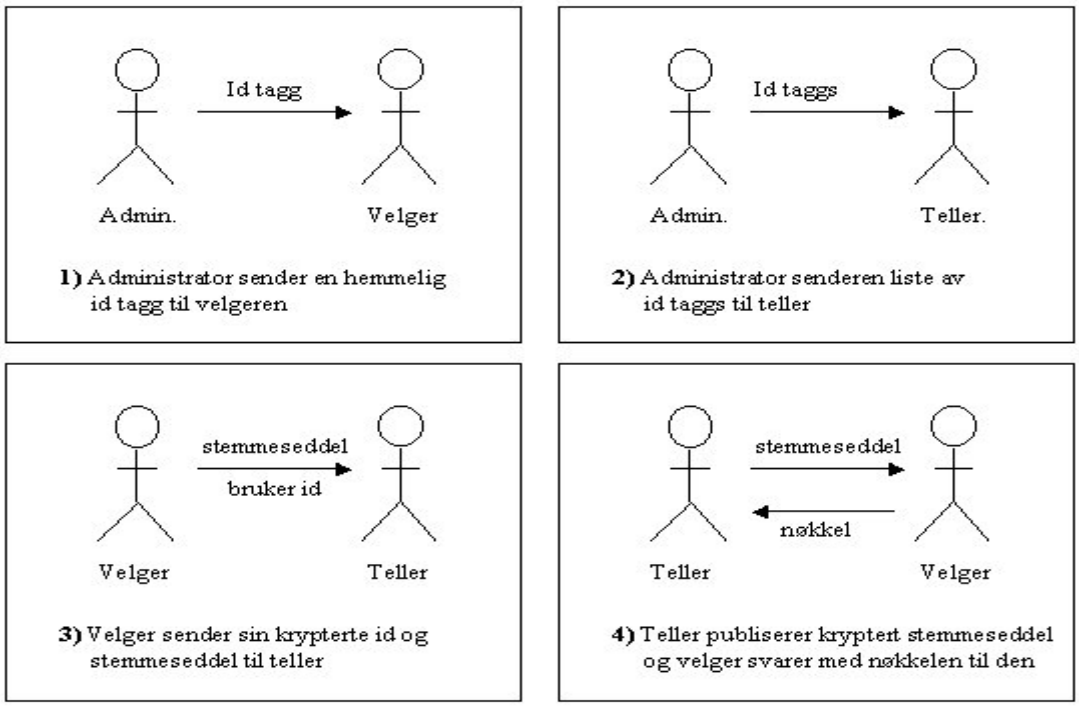


Figur 2 – The Simple Protocol

Denne protokolloppbygningen inneholder trekkene fleksibilitet, mobilitet og tilretteleggelse, men har flere store svakheter.

Velgere kan bruke andres bruker id. Velgere kan ikke være sikker på at administrator ikke leser stemmeseddelen: Anonymiteten til velgeren kan dermed brytes.
Man kan ikke være sikker på at administrator ikke endrer stemmeseddelen før han sender den videre til tellemekanismen eller at administrator lager stemmesedler som aldri har blitt innsendt av velgerne. Det er ingen måte å forsikre seg om at tellemekanismen registrerer stemmene på en riktig måte.
Problemet ved at velgere bruker andre sin bruker id kan løses ved at velgeren signerer sin stemmeseddel ved hjelp av digital signatur.
For å hindre at administrator bryter med velgerens identitet, ved at man krypterer stemmeseddelen med tellemekanismens offentlige nøkkel.

One and two agency protocols:
Tre personer; Nurmi, Salomaa og Santean [14] foreslo en måte å løse en del av problemene ved The Simple protocol.
Two agency protokollen virker slik at valg administrator distribuerer en hemmelig id tag til hver velger før valget tar til. Administrator sender så en liste over alle id tags, uten å identifisere velgerne, til en tellemekanisme. Velgerne sender sin id tag og en kryptert fil som inneholder en kopi av tagen og stemmeseddelen til tellemekanismen. Tellemekanismen kan nå sjekke om id tagen er gyldig, men det er ingen mulighet for programmet å sjekke innholdet av stemmeseddelen. Tellemekanismen publiserer den krypterte fila, og velgeren sender så en nøkkel som tellemekanismen trenger for å dekryptere den. Når valget er over vil tellemekanismen utgi en liste med alle stemmesedler og korresponderende krypterte filer. Dette gjør at velgeren kan bekrefte at stemmen hans ble telt riktig.

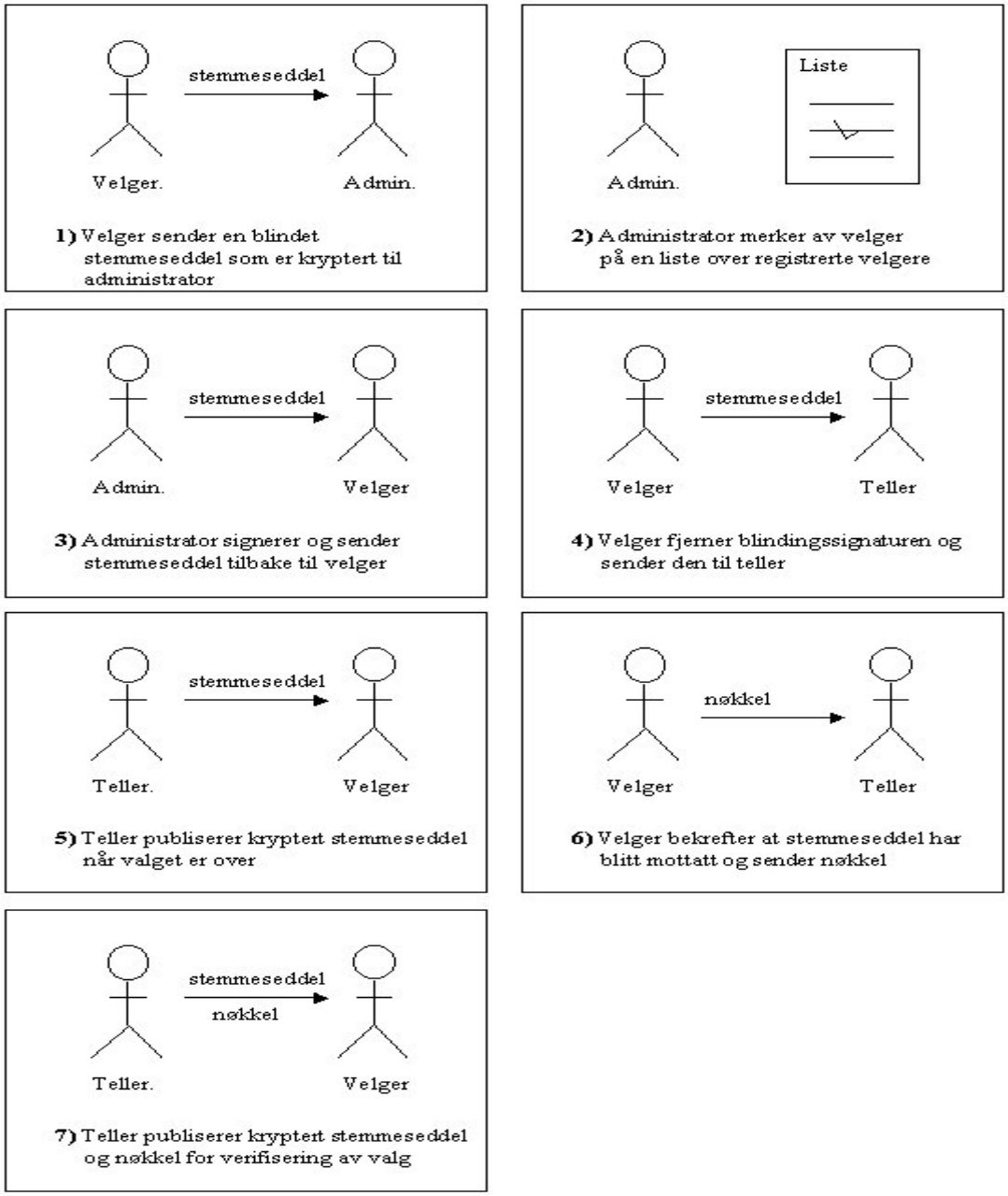


Figur 3 – Two Agency Protocol

Det viktigste her er at two agency protokollen ikke beskytter en velgers anonymitet dersom administrator og tellemekanismen er slått sammen, eller samarbeider. Det vil altså si at hvis begge de to autoritetene jobber sammen, så kan en i stedet se på de to som en autoritet; One agency protocol. En slik protokollstruktur er lik two agency protokollen bortsett fra hvordan id tags blir distribuert. I denne strukturen er det tellemekanismen som distribuerer id tags til velgeren. Denne strukturen har ingen administrator.

Den måten velgerne kan verifisere at stemmene deres ble telt opp riktig, tillater dem også å bevise at de stemte på en bestemt måte. En tellemekanisme kan telle stemmene til alle velgerne som har blitt tildelt id tags men som ikke bruker sin rett til å stemme. Velgerne som blir manipulert på denne måten kan rapportere det, men de kan ikke bevise at det ikke var dem selv som stemte.

Blind signature protocols:
Konseptet om blind signatur ble utviklet av David Chaum i 1982 [12]. Det gikk ut på at såkalte blinde signaturer kunne brukes for valg som hemmeliggjør stemmeseddelen.
Ti år senere utviklet Fujioka, Okamoto og Ohta [13] en valgmetode som tar bruk av blind signatur for å løse anonymitets problem i protokoller som Two agency protocol. Metoden ble utformet slik at kompleksiteten til protokollen ikke ble vesentlig større. Blinde signaturer er en form av digitale signaturer som gjør at man kan underskrive et dokument uten å røpe innholdet av det.



Figur 4 – Blind signature protocols

Strukturen er slik at en velger klargjør en stemmeseddel, krypterer den med en hemmelig nøkkel og *blinder* den. Velgeren underskriver på stemmeseddelen og sender den til administrator. Administrator sjekker og verifiserer at underskriften tilhører en registrert velger som ikke har stemt ennå. Administrator signerer og sender stemmeseddelen tilbake til velgeren hvis den er gyldig. Velgeren fjerner så *blindings* signaturen og sitter igjen med en kryptert stemmeseddel underskrevet av administrator. Denne stemmeseddelen sender velgeren så til tellemekanismen. Tellemekanismen sjekker om signaturen på stemmeseddelen er gyldig. Dersom den er det plasserer tellemekanismen stemmeseddelen på en liste som blir publisert etter at

alle velgerne har avlagt sin stemme. Etter at listen har blitt publisert sjekker velgerne at deres stemmeseddel finnes på listen og sender tellemekanismen krypteringsnøkkelen som er nødvendig for å dekryptere deres stemmeseddel. Tellemekanismen bruker disse nøklene til å dekryptere stemmesedlene og for deretter å summere valgresultatet. Etter valget publiserer tellemekanismen krypteringsnøklene sammen med de krypterte stemmesedlene slik at velgerne kan uavhengig sjekke valgresultatet.

Sensus polling protocol:
Lorrie Faith Cranor og Ron K. Cytron [17] sitt Sensus systemstruktur er løselig basert på Fujioka, Okamoto og Ohta [13] sin struktur blind signatur i valgsystem. Hovedforskjellen mellom disse to fremgangsmåtene skjer etter at velgeren har sendt den krypterte stemmeseddelen til tellemekanismen. I Sensus protokollen sender tellemekanismen en kvittering til velgeren. Velgeren kan da sende krypteringsnøkkelen umiddelbart etter å ha mottatt denne krypteringen og fullfører dermed valget.
Systemet bruker et program som gjennomfører all kryptering funksjoner og transaksjoner mot valgprogrammet på velgerens vegne.
Sensus protokollen har de fleste av de ønskelige karakteristikkene nevnt tidligere, men har ikke greid å rette på problemene den har arvet fra One/Two agency protocols.
Det største problemet er at administrator kan stemme på velgernes vegne. Disse ugyldige stemmene kan bli oppdaget av velgerne selv. Det er heller ingen måte å identifisere disse ugyldige stemmesedlene og fjerne dem fra opptellingen. Hvis velgere som ikke ønsker å stemme sender inn en blank stemmeseddel kan dette problemet unngås.

Ingen av de nevnte protokoller over tilfredsstiller ønske om å kontrollere at valgresultatet er talt opp riktig.

7 Noen eksisterende valgsystem

7.1 VoteHere Inc

Det amerikanske selskapet VoteHere [21], tilbyr et elektronisk valgsystem som har vært i bruk i USA siden oktober 1999. Det ble blant annet brukt i Alaska som et prøvevalg på USAs president. Systemet er også blitt brukt som prøvevalg i deler av California og Arizona.

VoteHere har utviklet to valgsystem:

- *VoteHere Gold*: et valgsystem som retter seg mot organisasjoner på privat sektor som universiteter, fagforeninger, politiske partier og spesielle interesse grupper.
- *VoteHere Platinum*: et valgsystem som er laget etter krav som retter seg mot offentlig sektor.

For å opprettholde anonymiteten til hver velger bygger VoteHere sitt elektroniske valgsystem seg på tre konstruksjoner:

- digitale signaturer, forklart i kapittel 3.4
- homomorphic kryptering og
- Zero knowledge proof som sikrer anonymitet i systemet

Homomorphic kryptering:
VoteHere har utviklet sine valgsystem med bruk av et krypteringssystem kalt homomorphic kryptering. Denne krypteringen er en metode som har den egenskap at summen av to krypterte nummer alltid er lik krypteringen av summen til de to. Det vil si at hvem som helst kan anslå og verifisere den krypterte sum av en samling av krypterte verdier. Fordi data er kryptert vil ikke en samme person vite hvilke numre som er kryptert, enten i form av originale verdier eller slutt summen, men han/hun vil vite at hva enn de krypterte verdiene er så opprettholder de summen/det totale forhold.

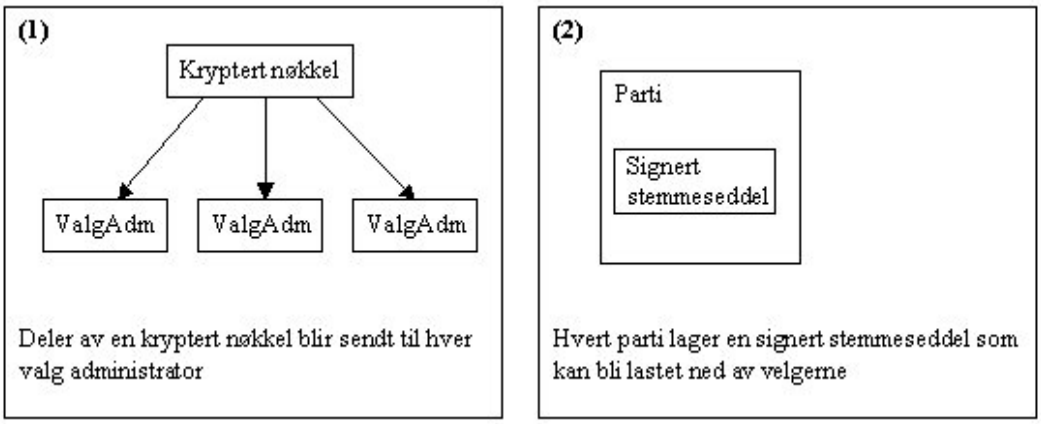
Zero knowledge proof:
Zero knowledge proof er en spesiell konstruksjon som tillater deltakere i en valgprosess, både velgerne og Telleautoritetene, å gjøre noe i hemmelighet (ivareta sin anonymitet), men de må også kunne bevise ærlighet i det som ble gjort.

VoteHere har følgende stegvis gjennomgang av et storskala valgsystem:

- før valget
- valg
- opptelling

Før valget:

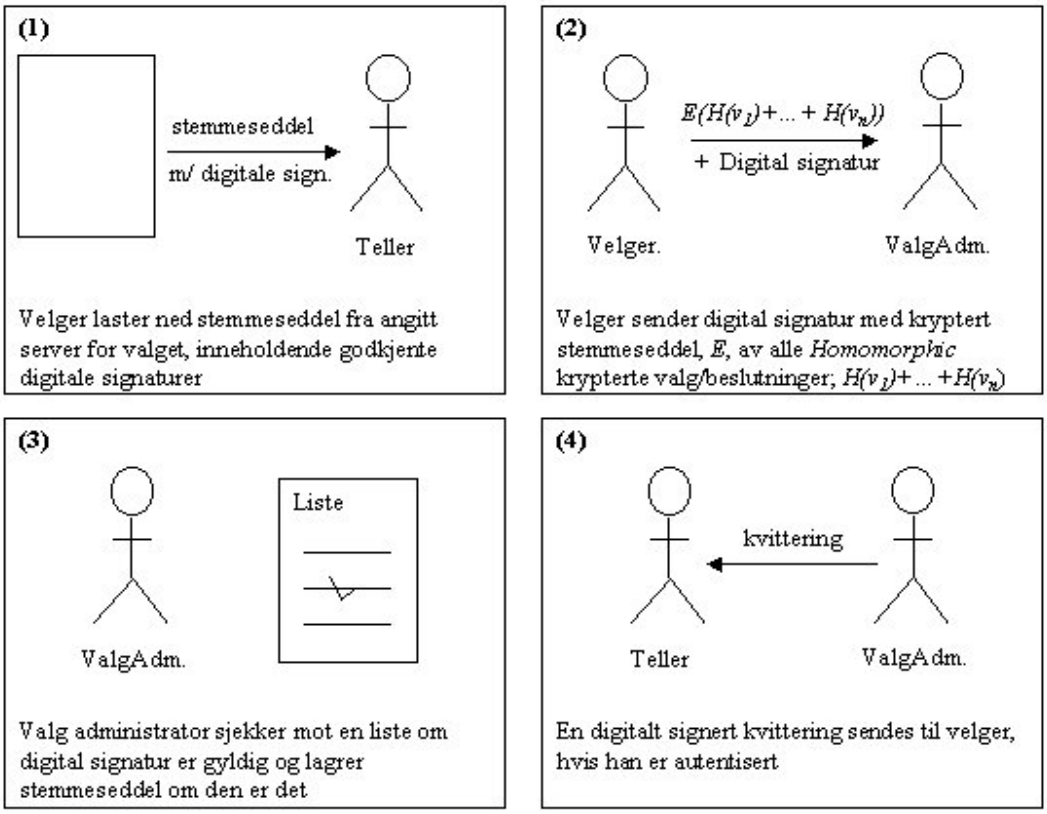
1. Valg autoritetene lager en kryptert nøkkel for valget. Denne nøkkelen er fordelt på forskjellige autoriteter slik at hver av dem får en bit av nøkkelen. Dette vil kreve at hver autoritet må samarbeide for å oppsummere valget.
2. Den offisielle elektroniske stemmeseddelen blir laget, og digitalt signert av hvert valgparti og funksjonær.



Figur 5 – VoteHere: Før valget

Valg:

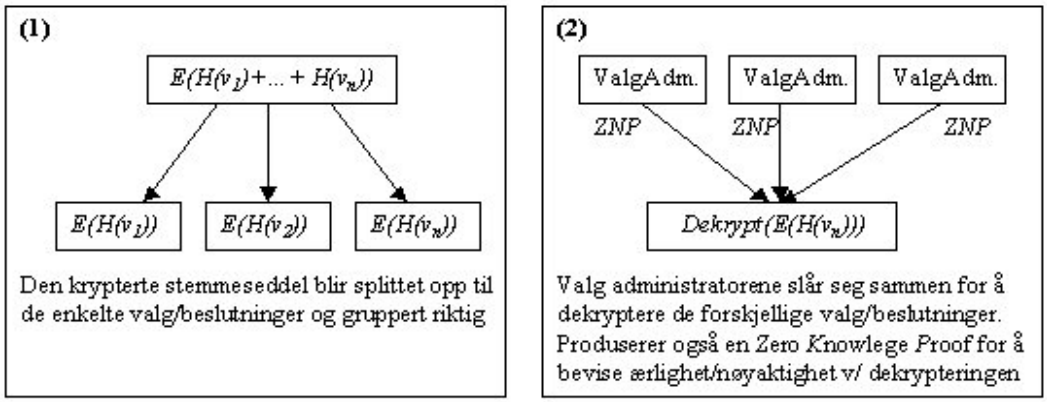
1. Hver velger kan be om en stemmeseddel via Internett fra en angitt server.
2. Når velgeren får stemmeseddelen inneholder den alle autoriserte digitale signaturer. Velgeren kan sjekke signaturene for å være sikker på at stemmeseddelen er korrekt.
3. Gjennom et software grensesnitt, vil velgeren kunne lage sin personlige stemmeseddel som blir kryptert med velgeren sitt valg.
4. For hver beslutning velgeren gjør vil en homomorphic kryptering bli lagt på hvert valg. Disse utgjør velgeren sine krypterte valg/beslutninger. Valgene/beslutningene blir lagt sammen i en sekvens som former, sammen med velgeren sin digitale signatur, velgeren sin signerte krypterte stemmeseddel. Denne blir returnert for opptelling.
5. Når valgadministrasjonen mottar en velger sin stemmeseddel, sjekkes den digitale signaturen for å se om det er en godkjent velger, da særlig om velgeren ikke har valgt før. Er signaturen godkjent blir den signerte stemmeseddelen lagret og en kvittering av valg, digitalt signert av et valgkontor, blir sent tilbake til velgeren.



Figur 6 – VoteHere: Valg

Opptelling:

1. Signerte stemmesedler blir tatt fra hverandre til de individuelle krypterte valg/beslutninger som ble gjort av velgeren. Krypterte valg/beslutninger blir gruppert på en riktig måte.
2. For hvert mulig valg/beslutning, blir den krypterte gjenpart angitt ved bruk av den homomorphic egenskap for alle de krypterte valg/beslutninger.
3. De tellende autoritetene må sammen dekryptere hver av de krypterte gjenpartene. Dette blir gjort slik at hver autoritet ikke avslører noen av de delte krypteringene. Hver autoritet må også produsere en zero knowledge proof som viser at autoriteten sin deltakelse i dekrypteringen var ærlig og nøyaktig.



Figur 7 – VoteHere: Før opptelling

4. De resulterende dekrypteringene utgjør felles det offisielle valg resultatet.

En forklaring av VoteHere sitt krypteringssystem, homomorphic kryptering, omtales i Vedlegg.

7.2 Sensus Polling Protocol

Sensus er laget av Lorrie Faith Cranor og Ron K. Cytron [17] og er bygd på modellen til Fujioka, Okamoto, and Ohta [13] som er nevnt i kapittel 6.1.

Sensus bruker blind signatur for å sikre seg at kun velgere som har rett til å velge kan avgi sin stemme, og at enhver velger kun stemmer en gang, samtidig som man da kan opprettholde velger sin anonymitet. Sensus tillater velgerne å sjekke om stemmene deres er blitt telt opp riktig, og å kunne anonymt klage på valgtellingen dersom deres stemme er telt opp feil. Sensus var opprinnelig lagd som en erstatning for post basert valgsystem. Sensus er i utgangspunktet et småskala valgsystem, men kan ifølge Cranor og Cytron utvides til storskala med mindre modifikasjoner.

Sensus Polling Protocol:
Protokollen krever at man har en *validator*, tallier og en *pollster* modul.

Validator er ansvarlig for valideringen, *tallier* er ansvarlig for opptelling og innhenting av stemmene, og *pollster* er ansvarlig for å utføre all kryptering og data overføring på velger sin vegne.

Gangen i Sensus:
Velger:

1. Gjør klar sin stemmeseddel, V
2. Krypterer stemmeseddelen med hemmelig nøkkel, $m = V^{se}$
3. *Blinder* stemmeseddelen, $b = mk^{ve} (mod\ vn)$
4. Signerer stemmeseddelen, b^{id}
5. Sender stemmeseddelen til *validator, (b, ID,* b^{id}) kryptert med *validator* sin offentlige nøkkel *ve*.

Validator:

6. Verifiserer at signaturen tilhører den registrerte velger
7. Sjekker om han har valgt før
8. Er stemmeseddelen gyldig signerer han stemmeseddelen, b^{vd} kryptert med *ie*
9. Sender stemmeseddelen tilbake til velger

Velger:

10. Fjerner "blindings" laget, $m^{vd} = b^{vd} / k \ (mod\ vn)$
11. Sender den signerte stemmeseddelen til *Tallier*, (m^{vd}, V^{se}) kryptert med *te*

Tallier:

12. Sjekker signaturen til stemmeseddelen, $(V^{se})=(m^{vd})^{ve}$
13. Dersom signaturen er gyldig, plasseres stemmeseddelen på en liste som blir offentliggjort etter valget
14. Signerer den krypterte stemmeseddelen, $(V^{se})^{td}$
15. Returnerer stemmeseddelen til velger som kvittering, $(V^{se})^{td}$, receipt #

Velger:

16. Mottar kvittering fra *Tallier*
17. Sender krypteringsnøkkel til *Tallier*, *sd.*

Tallier:

18. Mottar krypteringsnøkkelen fra velger
19. Dekrypterer stemmeseddelen, V^{se} med *sd*
20. Oppdaterer valgresultatet.

ve, vd, vn = *validator* sine krypteringsnøkler og modulus
V = stemmeseddel
ID = *velger* sitt Id nummer
se, sd = krypteringsnøkkel til stemmeseddel
m = kryptert stemmeseddel
k = stort tilfeldig primtall
ie, id, in = *velger* sine krypteringsnøkler og modulus
te, td, tn = *tallier* sine krypteringsnøkler og modulus
b = blindet stemmeseddel

Sensus er implementert ved hjelp av C og Perl på en Unix maskin, og de har benyttet seg av RSA for kryptering. Videre krever systemet at modulene blir kjørt på en server som støtter CGI scripts.

For mest mulig sikkerhet bør ikke de forskjellige modulene være på samme maskin.

Registrar:
Registrator er ansvarlig for å registrere velgere som har rett til å stemme. *Registrator* lager en liste over personer som er registrerte velgere. Registrerte velgere vil bli listet opp med navn eller identifiserings nummer, en offentlig krypteringsnøkkel, og email adresse dersom ønskelig.

Sensus krever at hver velger blir sendt et identifikasjons nummer samt en hemmelig *token T* før registreringen. Velgere som er blitt godkjente som velgere genererer et par med offentlig og private krypteringsnøkler, og registrerer seg til å stemme ved å sende *registrator* sin identifikasjons nummer , den hemmelige *token T* og den offentlige krypteringsnøkkelen. *Registrar* verifiserer at personen har gitt riktig *token* og legger deres identifikasjons nummer og offentlige krypterings nøkkel til den registrerte velger lista. Den registrerte velger lista inneholder også et felt som sjekker om personen har stemt eller ikke, den er 0 opprinnelig, og blir satt til 1 av *validator* når velgeren har fått sin stemme godkjent.

Pollster:
Pollster sin oppgave er å vise velgeren stemmeseddelen, samle inn velgeren sin stemme avgiving, utføre kryptering på stemmeseddelen, få tak i nødvendig validering og kvitteringer for brukeren, og avlevere stemmeseddelen.

Validator:
Validator er ansvarlig for å sjekke at velgeren har rett til å velge og at han bare velger en gang. *Validator* lager en "blinda" validering sertifikat ved å signere en "blindet" stemmeseddel. Velgeren "ublinder" så validerings sertifikatet og sender det til *tallier* sammen med sin stemmeseddel. Sensus bruker registrerings listen for å få tak i hver enkelt velger sin offentlige krypteringsnøkkel og sjekker deres signatur på stemmeseddelen. *Validator* forandrer valideringsfeltet fra 0 til 1 dersom signaturen godkjennes.

Tallier:
Tallier er ansvarlig for å samle inn stemmesedlene og å telle opp resultatet av valget. Velgere sender først krypterte stemmesedler signert av *validator*, så sjekker *tallier* om signaturen fra *validator* stemmer, og om den krypterte stemmen er unik blant de krypterte stemmene tatt i mot så langt. Dersom stemmeseddelen er unik, signerer *tallier* og sender tilbake til velger som en kvittering. Velgeren sender så krypteringsnøkkelen som trengs for å dekryptere stemmeseddelen, og *tallier* dekrypterer stemmeseddelen. Etter valget publiserer *tallier* en liste over krypterte stemmesedler, krypterings nøkler og dekrypterte stemmesedler, for å tillate uavhengig verifisering av valgresultatet.

7.3 E-VOX Voting System

E-Vox[16] er et system som er laget ved MIT (Massachusetts Institute of Technology), og er basert på "A practical secret voting scheme for large scale elections" av Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta [13]. Professor Rivest[26]var veileder for prosjektet, der Mark Herschberg, Kazho Ohta, Ben Adida, Brandon DuRette, Rachel Greenstadt, og Kevin McDonald deltok. E-Vox er et småskala valgsystem som er blitt brukt til diverse student valg ved MIT, og kan etter forfatteren Herschberg ha flere hundre kanskje flere tusen brukere, dersom serverne er raske nok og båndbredden er stor nok. Og han sier også at systemet lett kan utvides til flere tusen brukere.

Gangen i E-Vox sitt valgsystem:

Registrering:
Registrator:

1. Det blir laget en liste over velgere
2. Passord blir sendt ut til autoriserte velgere

Valg (1): Autentisering av stemmeseddel
Velger:

3. Laster ned websiden ved bruk av en sikker kanal: SSL. Siden inneholder administrator sin offentlige nøkkel (*A-PK*)
4. Velger skriver inn sitt navn og passord
5. Velger gjør sine valg og genererer sin stemmeseddel (*b*)
6. Stemmeseddelen blir avgitt $\xi(b)=HMAC\text{-}SHA_{k1,k2}(b)$
7. Den avgitte stemmeseddel blir blindet $B1=r^e\xi(b) \bmod n$, *e, n* er komponenter til *A-PK, r* er et tilfeldig nummer generert av bruker
8. En operasjons nøkkel blir laget av bruker og sent til administrator kryptert med hans *A-PK,* $E_{A\text{-}PK}(AS\text{-}Key)$
9. Velgeren sender administratoren $E_{AS\text{-}Key}(B1,\text{navn,passord})$, $E_{AS\text{-}Key}(MAC(E_{AS\text{-}Key}(B1,\text{navn,passord})))$

Administrator:

10. Dekrypterer og bekrefter *MAC*
11. Sjekker navn og passord for autentisering
12. Signerer: $Sb=r\xi(b)^d \bmod n$
13. Sender tilbake til velger *Sb, MAC(Sb)*

Velger:

14. Sjekker *MAC*
15. Tar vekk blindingen; $Su=\xi(b)^d \bmod n$
16. Sjekker signaturen

Valg (2):Avgi stemmeseddel
Velger:

17. Velger sender *Su* til Anon (Anonym remailer), kryptert med offentlig nøkkel av telleren, *C-PK;* $E_{C\text{-}PK}$*(CS-Key),* $E_{CS\text{-}Key}$*(ξ(b)d mod n, b, k1, k2), MAC(*$E_{CS\text{-}Key}$*(ξ(b)d mod n, b, k1,k2))*

Anon:

18. Anon; sender dette videre sammen med en *MAC* av alt, til teller

Opptelling:
Teller:

19. Teller dekrypterer meldinger
20. Sjekker signaturene
21. Offentliggjør *b, ξ(b)d mod n, k1,k2*
22. Summerer og offentliggjør resultatet

Kontroll:

23. Velger kan sjekke sin personlige stemme samt alle andre signaturer

MAC: en meldings autentiseringskode, i dette tilfelle en hash verdi av meldingen

8 Prototyp

På bakgrunn av sikkert valgsystem beskrevet i kapittel 2, metoder og protokoller beskrevet i kapittel 6, og de eksisterende systemene vi har nevnt i kapittel 7, ble det utformet et design til bruk for vår prototyp.

8.1 Design

Den største utfordringen var å få valgsystemet til å ta vare på velgeren sin anonymitet. Med anonymitet mener vi det at stemmesedlene ikke kan koples opp mot identiteten til den enkelte velger. Dette opprettholdes ved å benytte seg av kryptering av stemmesedlene. I en prototyp vet heller ingen av de aktørene som er i aksjon under valget, hvem de forskjellige velgerne er, fordi dette er kun oppgitt hos *Registrator*, som stenges før valget begynner.

Prototypen er delt opp som følger:

- *Bruker* som opererer på vegne av en valgdeltaker
- *Kontrollør* som sjekker om en bruker er godkjent til å velge eller ikke
- *Teller* som står for opptelling av valget

Bruker:
Denne delen av systemet tilbyr web sider der velgeren foretar sitt valg og skriver inn sin personlige stemmeId. Stemmen blir kryptert og sammen med stemmeId til velger blir den sendt over til *kontrollør*.

Kontrollør:
Oppgaven til *kontrolløren* er å sjekke om en velger har rett til å velge. Dette gjøres ved å sjekke den innkommende stemmeId mot en liste over godkjente velgere. Er velgeren autorisert, vil *kontrolløren* gi beskjed om dette til velgeren ved hjelp av en web side som gir velgeren mulighet til å avslutte stemmeavgivningen ved å sende den krypterte stemmeseddel til *Teller*.

Teller:
Telleren sin funksjon er å dekryptere stemmesedler og å telle de riktig opp.

8.2 Implementering

Når vi skulle implementere prototypen, valgte vi å benytte oss av asp [7] som gav oss muligheten til å benytte database kall ved hjelp av VBScripting [29]. Dette fordi vi valgte å representere funksjonaliteten i et valgsystem ved hjelp av tabeller. Hovedverktøyet vi benyttet oss av når vi skulle lage prototypen var Microsoft Frontpage og Microsoft Access.

Vi valgte å bygge gangen i implementeringen av systemet på *Two agency* protokollen, beskrevet i kapittel 6.1. Ved å gjøre dette valget settes fokus på å opprettholde anonymiteten ved hjelp av en krypteringsalgoritme uten å tenke på at de forskjellige aktøren (*Registrator*, *Kontrollør*, *Teller*) samarbeider med hverandre.

Algoritmen
Vi valgte RSA algoritmen for implementering i prototypen. RSA fungerer slik at to tilfeldige primtall *p, q* og en offentlig nøkkel *e* velges. Den hemmelige nøkkel *d* finnes ved;

$de=1 \bmod (p-1(q-1)$

En melding m blir omgjort til kryptert melding c med formelen;

$c=m^e \bmod n$, der $n=p*q$

Man kan dekryptere *c* med følgende formel;

$m=c^d \bmod n$

I implementeringen er det antatt at *Teller* har generert nøklene *(e, n), (d,n)* og distribuert offentlig nøkkel *(e, n)* til *Bruker*. *Bruker* krypterer stemmeseddelen som blir laget av velger i prototypen, med offentlig nøkkel og *Teller* dekrypterer denne med den private nøkkelen *(d, n)*.

Gangen i valgsystemet:

Velger:

1. Web side med valg av partier offentliggjøres til velger av *Bruker*
2. Velger gjør sitt *valg* – sender stemmeseddel

Bruker:

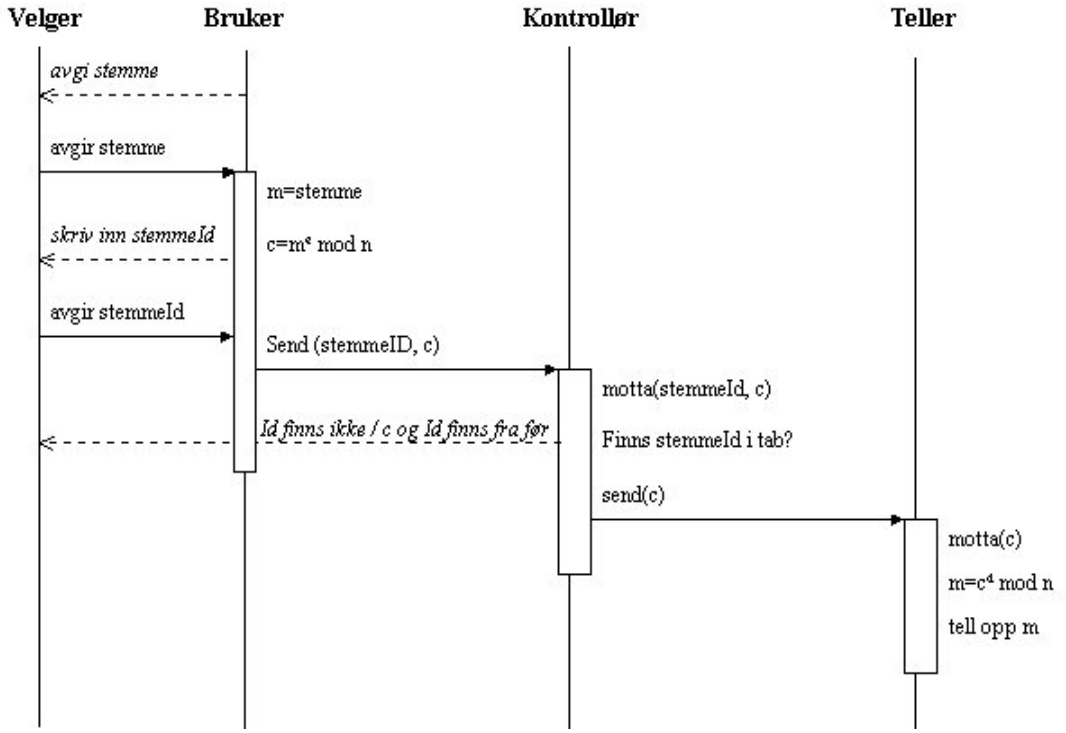
3. *Bruker* tar imot *valg* av velger og setter det lik melding *m*;
   $m=valg$
4. *Bruker* krypterer *m*;
   $c=m^e \bmod n$
5. En ny web side presenteres av *Bruker* for velger der *stemmeId* til velger tastes inn
6. *Bruker* tar velger sine verdier, *c* og *stemmeId*, og sender de til *Kontrollør*

Kontrollør:

7. Mottar *stemmeId* og *c* fra *Bruker*
8. Sjekker om *stemmeId* finnes i *Kontrollør* sin tabell. Finnes ikke *stemmeId* i tabellen presenteres en web side til velger at det er oppgitt ugyldig *stemmeId* og valgprossesen stoppes
9. Sjekker om *c* allerede er i tabellen til *Kontrollør* der tilhørende stemmeId finnes. Er *c* allerede i tabellen , presenterer *Kontrollør* en web side til velger at velger allerede har stemt
10. *Kontrollør* tilbyr velger en web side der velger må bekrefte valget som er gjort
11. Etter bekreftelse fra velger, sender *Kontrollør c* til *Teller*
12. *Kontrollør* lagrer *c* i tabell der tilhørende *stemmeId* finnes, som en bekreftelse at velger har foretatt et valg

Teller:

13. Mottar *c* fra *Kontrollør*
14. Dekrypterer c;
    $m=c^d \bmod n$, der m er opprinnelige melding som representerer velgeren sitt *valg*
15. Legger m til i valgresultatet ved å sjekke m mot tabellen til *Teller* for deretter å inkrementere riktig valg



Figur 8 – Prototyp

Forklaringer:
m= stemmeseddel, *c*= kryptert stemmeseddel, *e*= offentlig nøkkel, *d*= privat nøkkel
p, q= to primtall, $n=p*q$

8.3 Valg av implementasjon

Two agency protokollen har én autoritet som gjør både autorisering og valgopptelling. Samarbeider denne autoriteten med *Registratoren* av systemet vil identiteten til velgeren kunne avsløres. Vi valgte å utvide denne funksjonaliteten for prototypen ved å ha én valgautoritet for autorisering av velger og én autoritet som bare står for opptellingen av valget. Ved å ta med én autoritet som bare står for opptelling av stemmer vil sannsynligheten for å bryte velger sin anonymitet bli mindre. For å bryte anonymiteten må altså *Teller* samarbeide med *Kontrollør* som igjen må samarbeide med *Registrator* av systemet. Selv om *Kontrollør* og *Registrator* samarbeider for å jukse, vil anonymiteten til velgerne opprettholdes hvis *Teller* ikke kommer i kontakt med de to andre aktørene. For å øke sikkerheten på dette punktet er det en mulighet å plassere *Bruker*, *Kontrollør* og *Teller* på separate servere som er lokalisert fra hverandre.

I prototypen vil *Kontrolløren* ta seg av å sende velger sin krypterte stemmeseddel til opptelling hos *Teller*. Ved denne oppbyggingen kan *Kontrolløren* foreta valgjuks ved å sende *Telleren* flere eksemplarer av en innkommende kryptert stemmeseddel som medfører at én stemme blir telt flere ganger. For å hindre dette valgte vi å designe prototypen slik at *Kontrolløren* må få bekreftet av velger sendingen av en kryptert stemmeseddel over til *Teller*.

Det ble valgt å legge program koden til prototypen katalogisert ved *Bruker*, *Kontrollør* og *Teller* under samme plass på en server. Dette fordi vi så det som en fordel å ha all kode samlet for å opprettholde enkeltheten i systemet og fordi det ikke forandrer på prototypens formål; å opprettholde brukers anonymitet gjennom systemet.

8.4 Forutsetninger

Når vi skulle lage prototypen satte vi ned noen viktige forutsetninger for at systemet skulle ivareta brukerens anonymitet.

I vårt system vil velgeren sin identitet være skjult for både *Kontrollør* og *Teller*. Dersom man skal finne ut hva en velger har stemt, må både *Registrator*, *Kontrollør* og *Teller* samarbeide. Så en forutsetning vi har tatt er at de forskjellige aktørene ikke har kontakt med hverandre, og aller helst befinne seg fysisk separert fra hverandre på forskjellige servere. Det viktigste her er at *Registrator* ikke har kontakt med de andre to, siden det kun er han som har det fulle navnet til velgeren.

Det er forutsatt at *Registrator* har sendt en liste til *Kontrolløren*, over de velgere som har registret seg, før valget starter. Denne listen skal kun inneholde velgeren sin stemmeId, og ingen andre opplysninger som kan røpe velger sin identitet.

8.5 Avgrensninger

Det er blitt foretatt en del avgrensninger i implementasjonen. Vi har valgt å konsentrere oss om den delen som har med selve valget å gjør, fra velger sender sin stemmeseddel til teller mottar den. Bakgrunnen for at vi valgte denne avgrensningen var for å se på ivaretakelsen av brukers anonymitet fra stemmen ble avgitt, til den ble talt opp.Før valget kan begynne, må en person som ønsker å benytte seg av valgsystemet, registrere seg hos en *registrator*. Denne sjekker på en manntalls liste om personen er berettiget å stemme. Vi tenker oss en modell der den aktuelle personen fyller ut et skjema på en webside hos *registrator*, der han må fylle ut fødselsnummer navn og adresse. Dersom *registrator* godkjenner en bruker, sendes en unik stemmeId og et personlig passord til brukeren via posten. *Registrator* sender en stund før valgperioden tar til, en fullstendig liste over godkjente stemmeId til *Kontrolløren*. Dermed har ikke *Kontrolløren* noen mulighet til å se hvem som stemmer, med mindre *Registrator* og *Kontrollør* samarbeider.

I vårt system er *Registrator* ikke tatt med, men vi forutsetter at en registrering er gjort og at *Kontrolløren* og brukerne har mottatt de nødvendige data. Når brukeren skal avgi sin stemme, skriver han inn sin stemmeId og passord på en innloggings side, som gir adgang til valgsiden. Prototypen vår begynner etter at innloggingen har funnet sted.

8.6 Forbedringer

Med utgangspunkt i de kriteriene vi nevne i kapittel 6, vil vi gi et forslag på noen forbedre av vårt valg system.

En måte å forbedre nøyaktigheten i systemet på er ved å innføre digitale signaturer, kapittel 3.4. Ved å innføre dette kan man med sikkerhet vite at den som stemmer er den han gir seg ut for å være, samt at velger ikke kan nekte for å ha avlagt en stemme.

For å forbedre anonymiteten i systemet kan en bygge systemet på tankene om blind signatur protokollen, beskrevet i kapittel 6.1. Dette gjør at de forskjellige autoritetene i en valgprosess, kan samarbeide, uten at det går utover anonymiteten til velgeren.

En metode for å øke nøyaktigheten og kontrollerbarheten er å innføre flere *Tellere*, slik at dersom ikke alle *Tellerne* stemmer med hverandre kan man avsløre juks, *Kontrollør* må da sende ut igjen stemmesedlene for ny opptelling.

For å sikre dataoverføringen i systemet mot eventuelle inntrengere er det ønskelig å opprette en sikker kanal mellom de forskjellige aktørene. En vanlig måte å gjøre dette på er å benytte seg av sikringsprotokollen SSL, beskrevet i Vedlegg. En måte å sikre aktørene fra angrep, er å benytte seg av brannmurer, som filtrerer vekk inntrengere.

Grensesnittet i vår prototyp er laget med tanke på vanlige datamaskiner, og ikke til andre mer mobile terminaler. Det vil være en forbedring av mobiliteten til systemet dersom det også vart laget grensesnitt beregnet for mobile terminaler som WAP-telefon og PDA.

9 Drøfting

Det er utviklet et prototyp bygd opp av en struktur med tre hovedaktører, *Registrator*, *Kontrollør* og *Teller*. Oppgaven til *Registrator* er å gi godkjente velgere adgang til valget. *Kontrollør* sjekker om en velgerne har rett til å velge på bakgrunn av en liste over godkjente velgere mottatt fra *Registrator*. *Teller* sørger for at opptellingen av stemmene blir utført.

Et slikt system vil ha en del begrensninger i forhold til et ideelt elektronisk valg system. Et ideelt valg system vil måtte fullstendig tilfredsstille alle de kriteriene vi nevnte i kapittel 6.

Anonymitet
Vår implementering opprettholder anonymiteten til velger ved å kryptere stemmesedlene ved hjelp av *Teller* sin offentlige nøkkel. De aktørene som er i aksjon under valget (*Kontrollør* og *Teller*), vet ikke hvem de forskjellige velgerne er, fordi dette er kun oppgitt hos *Registrator*, som stenges før valget begynner.

Ved å bruke kryptering som eneste måte å ivareta velger sin anonymitet på, vil et samarbeid mellom de forskjellige aktørene, kunne føre til at velgeren sin identitet blir kjent. Vi har benyttet oss av to små primtall for kalkulering av de to krypterings nøklene, disse burde selvfølgelig økes betraktelig for å øke sikkerheten , slik at forsøk på å finne de to nøklene blir vanskeligere.

Fordelen med å ha tre aktører i stedet for to, som beskrevet i *Two-agency* protokollen i kapittel 6.1, er for at anonymiteten til en velger skal brytes så må *Telleren* samarbeide med *Kontrolløren*, som igjen må samarbeide med *Registrator*.

Blind signatur protokollen, beskrevet i kapittel 6.1, retter på problemet med samarbeid mellom de forskjellige aktørene. Dersom det foregår et samarbeid så vil ikke det røpe identiteten til velgeren. Både Sensus og E-Vox bygger på bruk av blind signatur for å opprettholde anonymiteten til velger.

Demokrati
Vi har gjort en antagelse i vår implementering at *Registrator* på forhånd har informasjon over alle personer som har rett til å stemme. Det er tenkt at velger må registrere seg før valget tar til hos *Registrator*, som sender ut en personlig stemmeId til velgeren. Listen over stemmeId blir overført til *Kontrollør*, og vil for *Kontrolløren* fungere som en manntallsliste ved manuelle valg.

Når en velger sender sin krypterte stemmeseddel og stemmeId til *Kontrolløren*, vil han se på sin liste om denne personen er blitt autorisert til å velge eller ikke. I tillegg har *Kontrolløren* også et felt for hver velger, som inneholder den krypterte stemmeseddel. Dette hindrer at personer kan stemme mer en én gang.

Fleksibelt
Vi har implementert vårt system med tanke på det Norske Stortingsvalg, der det er mulig å velge det parti du vil stemme på. Du må også skrive inn personlig stemmeId for å foreta valget. Systemet vårt har kun en type valgalternativ, siden det kun er et parti man kan stemme på. Det er ingen ting som hindrer et oppsett som tillater forskjellige valgalternativer.

VoteHere tilbyr flere slike valg alternativer, og sikrer hvert av disse valgalternativene ved å kryptere hver av dem for seg. Summen av disse krypterte valg alternativene vil da utgjør den fullstendige stemmeseddelen.

Kontrollbarhet
Telleren står for opptelling av valgstemmene. Dette gir ikke tilstrekkelig kontroll over at stemmene er blitt telt opp riktig, siden teller kan manipulere opptellingen uten at andre valgautoriteter kan sjekke det.

Et system med flere *Tellere* og der en lar *Kontrolløren* telle opp antall krypterte stemmesedler han har mottatt, kan man sjekke at de ulike *Tellerne* og *Kontrolløren* er enige i antall stemmer opptalt.

VoteHere sitt valgsystem, kapittel 7.1, lar de forskjellige telle autoritetene få en bit hver av den private krypterings nøkkelen før valget tar til. Ved opptelling må alle tellerne slå seg sammen for å dekryptere stemmesedlene, slik at man kan telle opp resultatet.

Mobilitet
Implementeringen av vårt system gir en web side som vil være tilgjengelige for alle brukere som har en datamaskin med en web browser og Internett tilgang. Dette gjør at mobiliteten til systemet er bra siden en velger kan bruke hvilken som helst datamaskin han måtte ønske, bare den har en internettoppkopling.
For enda større mobilitet hadde det vert ønskelig med en implementering som støtter bruk av WAP-telefoner, PDA'er og eventuelt andre mobilterminaler.

Nøyaktighet
I implementasjonen mottar *Teller* kryptert stemmeseddel fra *Kontrollør*, dekrypterer den og legger stemmen til i valgresultatet.

Før *Kontrollør* sender en kryptert stemmeseddel videre til *Teller*, ber den om velger sin bekreftelse om å sende den. Dette forhindrer at *Kontrollør* sender flere kopier av samme krypterte stemmeseddel til *Teller* for opptelling.

Tilretteleggelse
Implementasjonen tilbyr forskjellige web sider hvor velger avgir sin stemme og skriver inn sin personlige stemmeId før valget blir bekreftet til opptelling.
Valget vil foregå enkelt og raskt for velger som bare behøver å klikke seg videre etter å ha avgitt stemme og skrevet inn stemmeId. Alt av utstyr som trengs er en datamaskin med Internett tilgang og en browser som laster ned valgsidene.

10 Konklusjon

Med utgangspunkt i implementeringen av prototypen, beskrevet i kapittel 8.1. har vi sett at der vil være en del elementer som må forbedres for å opprettholde de tidligere nevnte sikkerhets kriterier, og da kunne brukes som et helhetlig system.

Implementeringen av en prototyp ivaretar bruker sin anonymitet dersom *kontrollør* ikke samarbeider med *registrator*. I vår prototyp har vi benyttet oss av små primtall for kalkulering av de forskjellige nøklene, disse burde økes betraktelig for å øke sikkerheten , slik at forsøk på å finne de to nøklene, som opprettholder velger sin anonymitet, blir vanskeligere.

For å øke nøyaktigheten til systemet kan vi innføre flere telle mekanismer, men for at nøyaktigheten skal bli god går dette ofte utover anonymiteten og omvendt. Systemet konsentrerer seg om velgeren sin anonymitet, og går ikke noe særlig inn på nøyaktighet ved opptelling av stemmer.

En implementering kan utvides til et tenkt elektronisk valgsystem, dersom *Registrator* og *Kontrollør* holdes adskilt, og at nøyaktigheten økes slik at der er mulighet for å kunne kontrollere at valget er telt riktig opp.

Prototypen vi har lagd slik den er nå, kan ikke brukes som et fullstendig offentlig valgsystem over Internett uten at de forskjellige forbedringen vi har skrevet blir utført.

Litteraturreferanser

[1] Security Technologies for the World Wide Web
Forfatter: Rolf Oppliger
Utgitt: Artech House computing library, 2000
ISBN: 1-58053-045-1

[2] Håndbok i datasikkerhet: *Informasjonsteknologi, sårbarhet og sikkerhet*
Forfattere: Torgeir Daler, Roar Gulbrandsen, Birger Melgård og Torbjørn Sjølstad.
Utgitt: Universitetsforlaget AS, 1993
ISBN 82-00-21848-1

[3] Internet cryptography
Forfatter: Richard E. Smith
Utgitt: Addison Wesley Longman Inc., 1997
ISBN 0-201-92480-3

[4] Computer Security
Forfatter: Dieter Gollman
Utgitt: John Wiley & Sons Ltd., 1999
ISBN 0-471-97844-2

[5] The Hacker's Handbook
Forfatter: Hugo Cornwall
Utgitt: Century Communication Ltd., 1985
ISBN 0-7126-0650-5

[6] Web Security & Commerce
Forfatter: Simpson Garfinkel, Gene Spafford
Utgitt: O'Reilly & Associates Inc., 1997
ISBN 1-56592-269-7

[7] ASP/MTS/ADSI, websecurity
Forfatter: Richard Harrison
Utgitt: Prentice Hall, 1999
ISBN 0-13-084465-9

[8] Web programing with asp and COM
Forfatter: Matt J. Crouch
Utgitt: Addison Wesley Longman Inc., 2000
ISBN 0-201-60460-4

[9] Discrete and combinatorial mathematics, an applied introduction, 3.ed.
Forfatter : Ralph P. Grimaldi
Utgitt: Addison Wesley Publishing Company Inc., 1994
ISBN 0-201-60044-7

[10] Secure Commerce On The Internet
Forfatter: Vijay Ahuja
Utgitt: Academic Press Inc., 1997
ISBN 0-12-045597-8

[11] The Internet Security Guidebook, from planning to deployment
Forfattere: Juanita Ellis og Timothy Speed
Utgitt: Academic Press Inc., 2001
ISBN 0-12-237471-1

[12] Blind signatures for untraceable payments. In *Proceedings of Crypto 82*
Forfattere: David Chaum
Utgitt, Plenum Press, New York. 1983
ISBN 0-306-41366-3

[13] A practical secret voting scheme for large scale elections.
In *Advances in Cryptology - AUSCRYPT '92*
Forfattere Fujioka, A, Okamoto, T., og Ohta, K.
Utgitt: , Springer-Verlag, Berlin. 1993
ISBN 3-540-57220-1

[14] Secret ballot elections in computer networks. *Computers and Security*
Forfattere: Nurmi, H., Salomaa, A. og Santean, L.
Utgitt: 1991

Weblinker

[15] LOV 1985-03-01 nr 03: Lov om stortingsvalg, fylkestingsvalg og kommunestyrevalg (Valgloven):
http://www.lovdata.no/all/hl-19850301-003.html

[16] The EVOX Voting System
http://theory.lcs.mit.edu/~cis/voting/voting.html

[17] Sider om Sensus
http://www.research.att.com/~lorrie/pubs/hicss/hicss.html
http://www.ccrc.wustl.edu/~lorracks/sensus/
http://www.research.att.com/~lorrie/voting

[18] Blind signatur
http://www.ccc.de/Library/eCash/encrypt.html
http://www.iks-jena.de/mitarb/lutz/security/cryptfaq/q39.html

[19] Electronic Election and Auctions av Jeannette M. Wing
http://www.cs.cmu.edu/afs/cs/academic/class/15827-f98/www/Slides/lecture8/base.000.html

[20] Forklaring på ord og uttrykk brukt i data sammenheng
http://www.whatis.com

[21] Sidene til VoteHere Inc.
http://www.votehere.com

[22] Generelle sider om valgsystem:
http://www.election.com
http://www.securepoll.com
http://www.thebell.net/

[23] Cryptography and Number Theory for Digital Cash
av J. Orlin Grabbe
http://www.aci.net/kalliste/cryptnum.htm

[24] ElGamal Encryption Scheme
http://cwis.kub.nl/~frw/people/koops/bindtech.htm

[25] Is Internet Voting Safe?
av Deborah M Phillips og David Jefferson
http://www.voting-integrity.org/text/2000/internetsafe.shtml

[26] Ronald R.Rivest sin hjemmeside
http://theory.lcs.mit.edu/~rivest/

[27] ResearchIndex en vitenskaplig litteratur database/bibliotek
http://citeseer.nj.nec.com/

[28] Computer Security Resource Center
http://csrc.nist.gov/encryption/

[29] VBScript Language Reference
http://www.adaptive.net/help/vbscript/vbstoc.htm

[30] Examining Internet Voting in Washington
David M. Elliott, Assistant Director of Elections
State of Washington
http://www.electioncenter.org/voting/InetVotingWhitePaper.html

[31] Some mathematical ideas from modular arithmetic used in RSA
http://www.momentus.com.br/PGP/doc/modulus.html
http://www.momentus.com.br/PGP/doc/howpgp.html

[32] RSA data security:
http://www.rsa.com

Vedlegg

Homomorphic kryptering

Diskret log problem

Ved å betrakte diskret logaritme problemet for en gruppe G inneholdende to primtall p (1024 bit) og q (160 bit), som tilfredsstiller $q \mid p-1$ ser man at G_q er gitt ved:

$G_q = \{1, g, g^2, g^3, \ldots, g^{q-2}, g^{q-1}\}$, hvor g er et element av rekken q, som medfører $g^q=1$, og multiplikasjon blir gjort modulo primtallet p. Vi har da for hver h som er med i G_q eksisterer det en unik x, $0<x<q$ slik at $h = g^x$. Vi kaller da x den diskret log av h med hensyn på g: $x = \log_g h$.

Diskret logg problemet er å beregne $\log_g h$ for en tilfeldig h (og fast g).

ElGamal kryptering

Et offentlig nøkkel system. Kan bli brukt for kryptering og digital signering på en måte som likner på RSA algoritmen.

ElGamal krypteringssystem er en enkel metode for offentlig nøkkel kryptering basert på vanskeligheten til diskré logaritme problemet.

Hvis vi antar et oppsett med p, q og g, beskrevet under diskret log problemet, så har vi følgende krypteringsmetode:

Nøkkel generering:

Hver deltaker i systemet genererer et nøkkel par ved å velge et tilfeldig nummer x, $0<x<q$. Privat nøkkel settes lik x og offentlig nøkkel $h = g^x$.

På grunn av logaritmens vanskelighet vil det være en umulighet å finne x når bare h er opplyst.

Kryptering:

For å kryptere en melding m for en mottaker med offentlig nøkkel h, så anslår man den krypeterte teksten (a, b) som:

$(a, b) = (g^r, h^r m)$, hvor r er et tilfeldig tall, $0<r<q$.

En kryptering består altså av nummer par.

Dekryptering:

For å dekryptere nummer paret (a, b) vil mottaker bruke sin private nøkkel x for å få tilbake meldingen m;

$m = b/a^x$.

Homomorphic ElGamal kryptering

En offentlig nøkkel algoritme *E* blir kalt homomorphic hvis den tilfredsstiller :

$E_{pk}(v1)*E_{pk}(v2) = E_{pk}(v1+v2)$, for hver offentlig nøkkel *PK* og meldinger *v1* og *v2*. Hvis vi multipliserer to krypterte meldinger (med samme offentlig nøkkel) så blir resultatet en kryptering av summen av de opprinnelige meldingene.

ElGamal krypteringssystem kan på en enkel måte gjøres homomorphic for krypterte stemmer *v* (som er med i mengden {0, 1}), ved å sette $m = g^v$.

Ved å multipliserer $(a1, b1) = (g^{r1}, h^{r1}g^{v1})$ og $(a2, b2) = (g^{r2}, h^{r2}g^{v2})$ får vi:

$(a1, b1)*(a2, b2) = (a1a2, b1b2) = (g^{r1+r2}, h^{r1+r2}g^{v1+v2})$, som er ElGamal kryptering av *v1+v2*.

RSA

RSA er et kryptering system laget i 1977 av Ron Rivest, Adi Shamir, og Leonard Adleman. Dette er en algoritme som genererer to krypterings nøkler, en offentlig nøkkel som kan distribueres fritt, og en privat nøkkel som må holdes skjult for omverden.

RSA fungerer slik at to tilfeldige primtall *p, q* og en offentlig nøkkel *e* velges. Den hemmelige nøkkel *d* finnes ved;

$de=1 \bmod (p-1(q-1))$

En melding m blir omgjort til kryptert melding c med formelen;

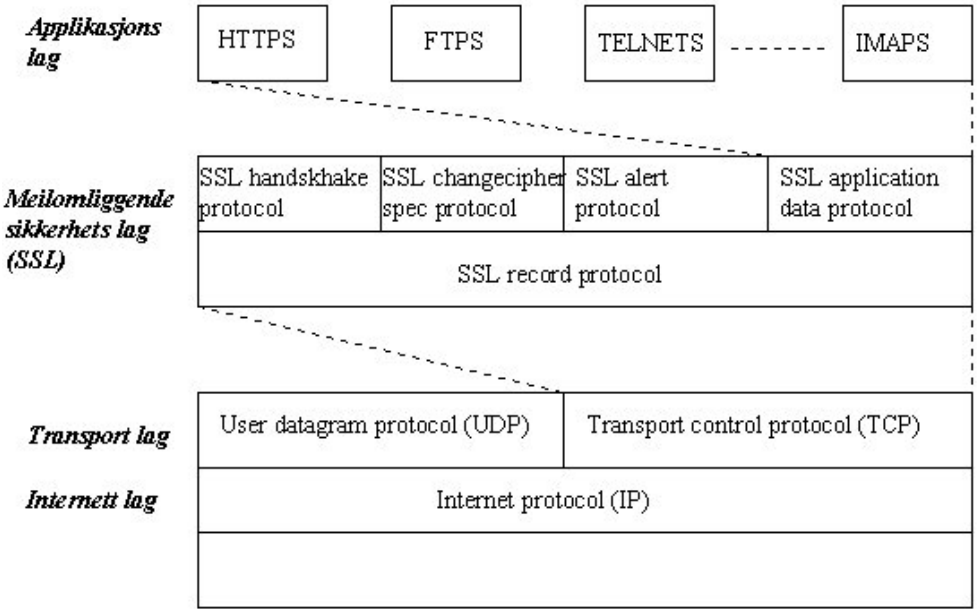
$c=m^e \bmod n$, der $n=p*q$

Man kan dekryptere *c* med følgende formel;

$m=c^d \bmod n$

SSL 3.0

SSL (Secure Sockets Layer) er en sikker applikasjonslagprotokoll, som er designet og utviklet av Netscape Communications. Siste versjon 3.0 kom ut i 1996 og er en forbedring av versjon 2.0 og Microsoft sin PCT protokoll. Den er en kjent standard for sikring av data og tilbyr en sikker kanal for dataoverføring. Den er den protokollstandard som er mest i bruk innefor sikkerhet angående Internett transaksjoner. Den tilbyr en "sikker kanal". Med dette menes en kanal som sikrer data ved overføring fra en klient til server eller omvendt.
Protokollen implementeres over transportlaget (TCP) og under applikasjonslaget. Applikasjonsprotokoller som HTTP, FTP osv. kan bli sikret ved å legge dem "over" SSL. En "adresse" tilhørende applikasjonsprotokollen HTTP vil med SSL tilknytning begynne med https://… der s indikerer beskyttelse av SSL. SSL bruker altså TCP på vegne av overliggende applikasjonsprotokoller.



Figur 9 – SSL arkitektur og protokoll

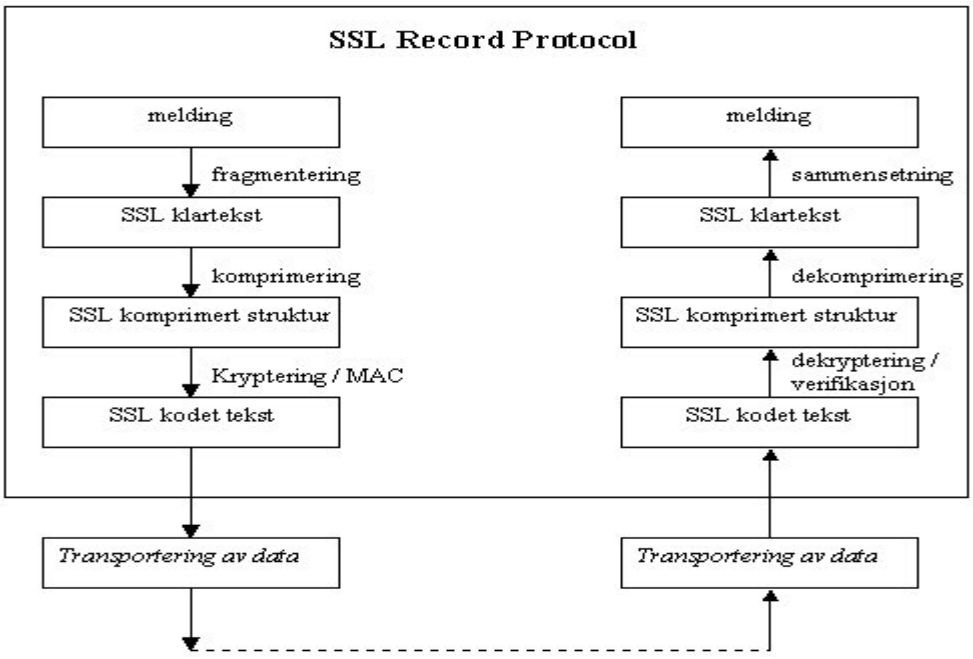
Fordi den er innebygd i alle dagens store web browsere og web servere vil den ved å installere en digital sertifikat aktivere deres SSL egenskaper.
Mens TCP/IP protokollen sender en anonym feilfri strøm av informasjon mellom to maskiner, så vil SSL tilby en del sikkerhetstiltak til informasjonsstrømmen

- autentisering av serveren, ved bruk av digitale signaturer
- autentisering av klienten, ved bruk av digitale signaturer
- data sikkerhet ved bruk av kryptering
- data integritet ved bruk av melding autentiserings koder

De kompliserende parter, både server og klient, kan autentisere hverandre ved bruk av offentlig nøkkel kryptering. Konfidensialiteten til datatrafikken er beskyttet siden forbindelsen er kryptert etter at både server og klient har oppnådd forbindelse og forhandlet om nøkkel. Datastrøm over forbindelsen er sjekket og autentisert ved bruk av MAC (Message Authentication Code), slik at integriteten til datastrømmen ikke forandres.
SSL består av to lag:

- SSL Record protocol
- SSL sub protokoller (Handshake protocol, ChangeCipherSpec protocol, Alert protocol)

SSL record protocol:
Denne protokollen er brukt for innkapsling av forskjellige høyere lags protokoller. Den mottar ugjenkjennelig data fra høyere overføringslag til SSL kryptert tekst. Mottar også kryptert tekst fra lavere overføringslag til opprinnelige data. Overgangene mellom den krypterte teksten til opprinnelig melding skjer ved kryptering/dekryptering, komprimering/dekomprimering og fragmentering/sammensetning:



Figur 10 – SSL Record Protocol

SSL sub protokoller tilbyr spesielle meldingstyper som blir sendt ved å bruke SSL Record protocol.
Handshake protocol bruker SSL Record protocol til å etablere de sikre attributter av en operasjon mellom den SSL aktiverte server og klient.

Prototyp – kode

De forskjellige aktørene involvert i prototypen, *Bruker*, *Kontrollør* og *Teller*, ligger under hver sin mappe på samme server.

Bruker:

index.html

```
<html>

<body>

<p>(Bruker)</p>
<p> </p>
<form method="POST" action="Bruker.asp" onSubmit="">
   <p>Gi din stemme:</p>
  <p><input type="radio" name="Valg" value="1">Arbeiderpartiet</p>
  <p><input type="radio" name="Valg" value="2">Høyre</p>
  <p><input type="radio" name="Valg" value="3">Venstre</p>
  <p> </p>
  <p><input type="submit" value="Stem" name="stem"></p>
</form>
<p> </p>

</body>

</html>
```

bruker.asp

```
<html>
<body>

<p>(Bruker)</p>

<%
    ' --
    M = Trim(Request.Form("Valg"))    ' -- Stemmeseddel
    E = 17                            ' -- Offentlig nøkkel, fått fra teller
    N = 253                           ' -- Summen av to tilfeldige primtall



    C = (M^E) Mod N                   ' -- Kryptert stemmeseddel

%>
<form method="POST" action="http://portal/valg/kontrollor/kontrollor.asp">
   <p>StemmeId</p>
  <p><input type="text" name="txtId" size="20"></p>
  <p><input type="hidden" name="KrValg" value=<%= C %>></p>
  <p><input type="button" value="Send ID" name="ID" onClick="if
(this.form.txtId.value==0) {alert('Du må oppgi stemmeId i feltet!');} else
this.form.submit();"></p>
</form>

</body>

</html>
```

Kontrollør:

kontrollor.asp

```
<html>

<body>
(Kontrollør)
<%

   strID = Trim(Request.Form("txtID"))  ' -- stemmeId fått fra Bruker.asp
   KrValg = Trim(Request.Form("KrValg"))   ' -- Krypert stemmeseddel fått
fra Bruker.asp

   Set objConn = Server.CreateObject("ADODB.Connection")
   Set objRS = Server.CreateObject("ADODB.Recordset")

   objConn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" &
Server.MapPath("kontrollor.mdb"))

   set objRS = objConn.Execute ("SELECT Id, stemmeseddel FROM kontrollor
WHERE Id='" & strID & "'")

   IF objRS("stemmeseddel")<>"" Then

%>
<p><font size="5">Du har valgt tidligere!</font></p>
<p>

<%
   else
%>
<p><font size="5">Bekreft stemmen til o<font
COLOR="#000000">pp</font>telling</font></p>
<p>
<form method="POST" action="http://portal/valg/teller/teller.asp">
   <input type="hidden" name="KrValg" value=<%= KrValg%>>
  <p><input type="submit" value="Bekreft" name="Bekreft"></p>
</form>

<%     ' -- Setter kryptert stemmeseddel inn i tabell
       strSQL = "UPDATE kontrollor SET stemmeseddel='" & KrValg & "' WHERE
Id='" & strID & "'"

       set objRS = objConn.Execute (strSQL)

       objConn.Close
       set objConn = Nothing

   End If
%>

</body>

</html>
```

Teller:

teller.asp

```
<html>

<body>
(Teller)

<%
    KrValg = Trim(Request.Form("KrValg"))       ' -- Kryptert stemmeseddel
mottatt fra Kontrollor.asp

    C = KrValg
    N = 253                                     ' -- Summen av to tilfeldige
primtall
    D = 13                                      ' -- Privat nøkkel til teller

    G = C
    For i=1 To D/2
        F = (C*C) Mod N
        G = (F*G) Mod N
    Next

    M=G                                         ' -- Dekryptert stemmeseddel


    ' -- Opptelling:

    Set objConn = Server.CreateObject("ADODB.Connection")
    Set objRS = Server.CreateObject("ADODB.Recordset")
    objConn.Open("DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" &
Server.MapPath("teller.mdb"))

    If Trim(M)=1 then
        strSQL = "UPDATE teller SET ap=ap+1"
    End if

    If Trim(M)=2 then
        strSQL = "UPDATE teller SET h=h+1"
    End if

    If Trim(M)=3 then
        strSQL = "UPDATE teller SET v=v+1"
    End if

    set objRS = objConn.Execute (strSQL)

    objConn.Close
    set objConn = Nothing

    Response.Write("Stemmen er talt opp!")

%>

</body>

</html>
```