Modelling and verifying e-voting protocols in applied-pi calculus

Stéphanie Delaune and Steve Kremer

LSV, ENS Cachan & CNRS & INRIA Saclay Île-de-France,

Wednesday 1st September

Outline

Lecture 1: Introduction to protocol analysis in applied pi

$\longrightarrow$ today

Lecture 2: Formalisation and verification of security properties

Part I: Privacy-type properties
(based on joint work with M. Ryan)

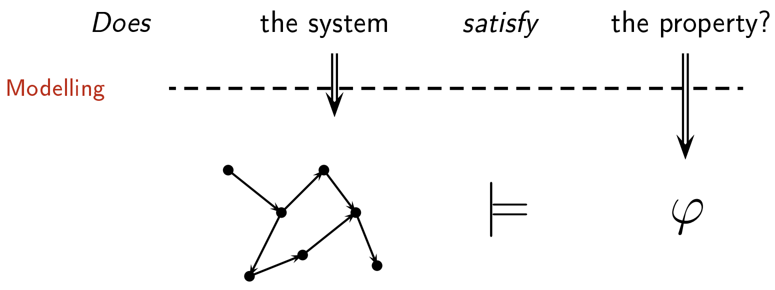
Part II: Verifiability properties
(based on joint work with M. Ryan and B. Smyth)

$\longrightarrow$ on Friday

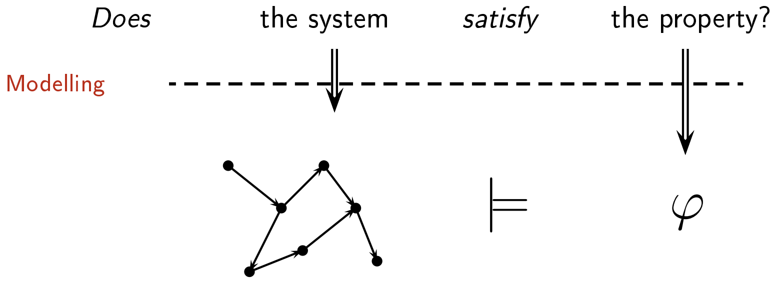
Part I

Formal methods and security protocols

Formal methods for system verification



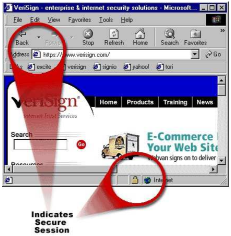
Major successes: formal methods in hardware design, software model-checking of drivers, static analysis of large scale embedded systems, ...

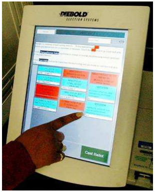


2007 Turing award for Computer aided verification

To Clarke, Emerson and Sifakis: *For their role in developing Model-Checking into a highly effective verification technology, widely adopted in the hardware and software industries.*

Cryptographic protocols everywhere!

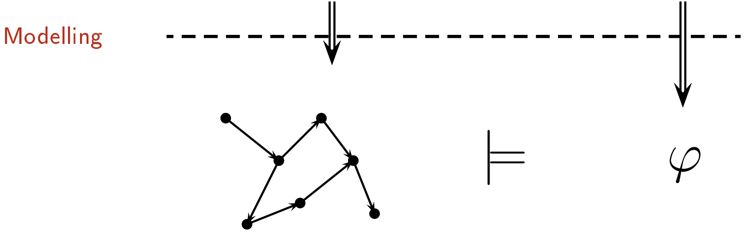




Cryptographic protocol:
a distributed program which uses cryptographic primitives (e.g. encryption, digital signatures, ...) to ensure a security property (e.g. confidentiality, authentication, anonymity, ...)

Formal methods for protocol verification

Does the protocol *satisfy* a security property?

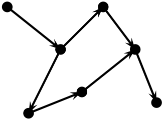


Formal methods for protocol verification

Does the protocol *satisfy* a security property?

Modelling

$|$



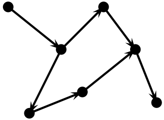
$\models \quad \varphi$

- protocol is executed in adversarial environment

Formal methods for protocol verification

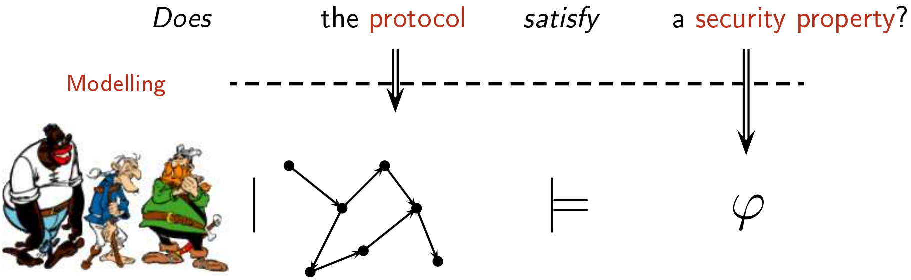
Does the protocol *satisfy* a security property?

Modelling

 $\|$  $\models \varphi$

- protocol is executed in adversarial environment
- in this talk: protocols are modelled in the applied pi calculus

Formal methods for protocol verification

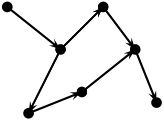


- protocol is executed in adversarial environment
- in this talk: protocols are modelled in the applied pi calculus
- attackers are any process which can be written in the applied pi calculus

Formal methods for protocol verification

Does the protocol *satisfy* a security property?

Modelling

$|$  $\models \quad \varphi$

- protocol is executed in adversarial environment
- in this talk: protocols are modelled in the applied pi calculus
- attackers are any process which can be written in the applied pi calculus
- partial automation using the verification tool ProVerif

The Needham-Schroeder public key protocol (1978)

- $$\begin{array}{rcrr} A & \rightarrow & B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

The Needham-Schroeder public key protocol (1978)

$$A \rightarrow B : \{A, N_a\}_{\mathsf{pub}(B)}$$

- $$B \rightarrow A : \{N_a, N_b\}_{\mathsf{pub}(A)}$$

$$A \rightarrow B : \{N_b\}_{\mathsf{pub}(B)}$$

The Needham-Schroeder public key protocol (1978)

$$A \rightarrow B : \{A, N_a\}_{\mathsf{pub}(B)}$$
$$B \rightarrow A : \{N_a, N_b\}_{\mathsf{pub}(A)}$$

- $A \rightarrow B : \{N_b\}_{\mathsf{pub}(B)}$

The Needham-Schroeder public key protocol (1978)

$$\begin{array}{rcll} A & \rightarrow & B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

$INIT \triangleq$

$\mathsf{in}(c, xpkb).\nu na.$
$\mathsf{out}(c, \mathsf{aenc}(\langle \mathsf{pk}(ska), na\rangle, xpkb)).$
$\mathsf{in}(c, x).$
$\mathsf{if}\ \mathsf{fst}(\mathsf{adec}(x, ska)) = na\ \mathsf{then}$
$\mathsf{let}\ xnb = \mathsf{snd}(\mathsf{adec}(x, ska))\ \mathsf{in}$
$\mathsf{out}(c, \mathsf{aenc}(xnb, xpkb)).0$

$RESP \triangleq$

$\mathsf{in}(c, y)$
$\mathsf{let}\ ypka = \mathsf{fst}(\mathsf{adec}(y, skb))\ \mathsf{in}$
$\mathsf{let}\ yna = \mathsf{snd}(\mathsf{adec}(y, skb))\ \mathsf{in}$
$\nu nb.\mathsf{out}(c, \mathsf{aenc}(\langle yna, nb\rangle, ypka))$
$\mathsf{in}(c, z).$
$\mathsf{if}\ \mathsf{adec}(z, skb) = nb\ \mathsf{then}\ P$

$$NSPK \triangleq \nu ska.\mathsf{out}(\mathsf{pk}(ska)).!INIT \quad | \quad \nu skB.\mathsf{out}(\mathsf{pk}(skb)).!RESP$$

The Needham-Schroeder public key protocol (1978)

$$\begin{array}{rcrr} A & \rightarrow & B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

Questions

- Is N_b a shared secret between A and B?
- When B receives $\{N_b\}_{\mathsf{pub}(B)}$, does this message really originate from A ?

The Needham-Schroeder public key protocol (1978)

$$\begin{array}{rcll} A & \rightarrow & B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

Questions

- Is N_b a shared secret between A and B?
- When B receives $\{N_b\}_{\mathsf{pub}(B)}$, does this message really originate from A ?

An attack was found 17 years after its publication!

A Man-in-the-middle attack

Agent A

Intruder I

Agent B

$$\begin{array}{lcll} A & \longrightarrow & B & : \{N_a, A\}_{\mathsf{pub}(B)} \\ B & \longrightarrow & A & : \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \longrightarrow & B & : \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

A Man-in-the-middle attack

Agent A

$\{N_a, A\}_{\mathrm{pub}(I)}$ $\longrightarrow$

Intruder I

Agent B

- $$\begin{array}{lclcl} A & \longrightarrow & B & : & \{N_a, A\}_{\mathrm{pub}(B)} \\ B & \longrightarrow & A & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\ A & \longrightarrow & B & : & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

A Man-in-the-middle attack

$\xrightarrow{\{N_a, A\}_{\mathrm{pub}(I)}}$ $\xrightarrow{\{N_a, A\}_{\mathrm{pub}(B)}}$

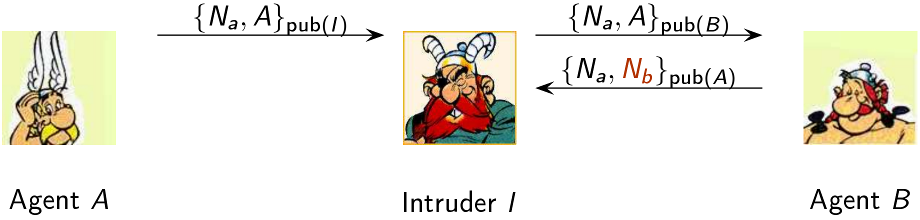
Agent A

Intruder I

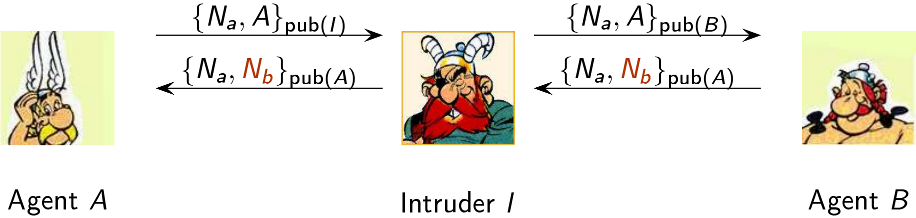
Agent B

- $$\begin{array}{lcll} A & \longrightarrow & B & : \{N_a, A\}_{\mathrm{pub}(B)} \\ B & \longrightarrow & A & : \{N_a, N_b\}_{\mathrm{pub}(A)} \\ A & \longrightarrow & B & : \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

A Man-in-the-middle attack

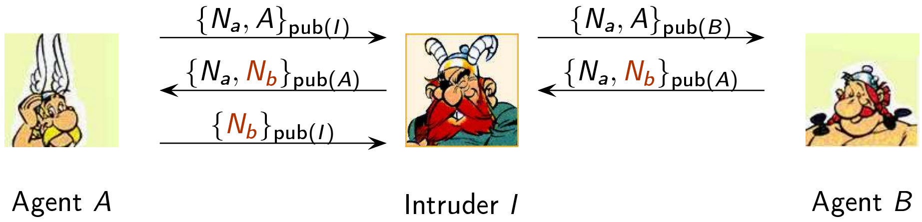


$$
\begin{array}{lcll}
A & \longrightarrow & B & : \{N_a, A\}_{\mathrm{pub}(B)} \\
B & \longrightarrow & A & : \{N_a, N_b\}_{\mathrm{pub}(A)} \\
A & \longrightarrow & B & : \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$



$$
\begin{array}{lcll}
A & \longrightarrow & B & : \{N_a, A\}_{\mathrm{pub}(B)} \\
\bullet\ B & \longrightarrow & A & : \{N_a, N_b\}_{\mathrm{pub}(A)} \\
A & \longrightarrow & B & : \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

A Man-in-the-middle attack



$$
\begin{array}{lcccl}
 & A & \longrightarrow & B & : \{N_a, A\}_{\mathrm{pub}(B)} \\
 & B & \longrightarrow & A & : \{N_a, N_b\}_{\mathrm{pub}(A)} \\
\bullet & A & \longrightarrow & B & : \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

A Man-in-the-middle attack

Agent A

$\{N_a, A\}_{\mathsf{pub}(I)} \longrightarrow$

$\longleftarrow \{N_a, N_b\}_{\mathsf{pub}(A)}$

$\{N_b\}_{\mathsf{pub}(I)} \longrightarrow$

Intruder I

$\{N_a, A\}_{\mathsf{pub}(B)} \longrightarrow$

$\longleftarrow \{N_a, N_b\}_{\mathsf{pub}(A)}$

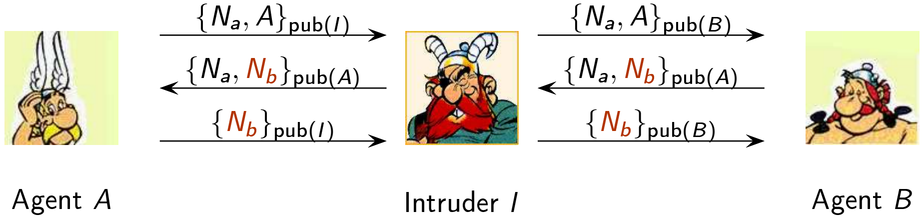
$\{N_b\}_{\mathsf{pub}(B)} \longrightarrow$

Agent B

$$A \longrightarrow B : \{N_a, A\}_{\mathsf{pub}(B)}$$
$$B \longrightarrow A : \{N_a, N_b\}_{\mathsf{pub}(A)}$$

- $A \longrightarrow B : \{N_b\}_{\mathsf{pub}(B)}$

A Man-in-the-middle attack



Answers

- Is N_b a shared secret between A and B?
  $\hookrightarrow$ No

A Man-in-the-middle attack

Agent A

$\{N_a, A\}_{\mathsf{pub}(I)} \longrightarrow$

$\longleftarrow \{N_a, N_b\}_{\mathsf{pub}(A)}$

$\{N_b\}_{\mathsf{pub}(I)} \longrightarrow$

Intruder I

$\{N_a, A\}_{\mathsf{pub}(B)} \longrightarrow$

$\longleftarrow \{N_a, N_b\}_{\mathsf{pub}(A)}$

$\{N_b\}_{\mathsf{pub}(B)} \longrightarrow$

Agent B

Answers

- Is N_b a shared secret between A and B?
  $\hookrightarrow$ No
- When B receives $\{N_b\}_{\mathsf{pub}(B)}$, does this message really originate from A?
  $\hookrightarrow$ No

A Man-in-the-middle attack

$\{N_a, A\}_{pub(I)} \rightarrow$

$\leftarrow \{N_a, N_b\}_{pub(A)}$

$\{N_b\}_{pub(I)} \rightarrow$

$\{N_a, A\}_{pub(B)} \rightarrow$

$\leftarrow \{N_a, N_b\}_{pub(A)}$

$\{N_b\}_{pub(B)} \rightarrow$

Agent A Intruder I Agent B

Answers

- Is N_b a shared secret between A and B?
  $\hookrightarrow$ No
- When B receives $\{N_b\}_{pub(B)}$, does this message really originate from A?
  $\hookrightarrow$ No

Remark : Crypto has not been broken
$\hookrightarrow$ Attack on the protocol logic

Part II

The applied pi calculus

Motivation for using the applied π-calculus

Applied pi-calculus: [Abadi & Fournet, 01]
basic programming language with constructs for concurrency and communication

- based on the π-calculus [Milner *et al.*, 92]
- in some ways similar to the spi-calculus [Abadi & Gordon, 98]

Motivation for using the applied π-calculus

Applied pi-calculus: [Abadi & Fournet, 01]

basic programming language with constructs for concurrency and communication

- based on the π-calculus [Milner *et al.*, 92]
- in some ways similar to the spi-calculus [Abadi & Gordon, 98]

Advantages:

- allows us to model less classical cryptographic primitives
- both reachability and equivalence-based specification of properties
- automated proofs using ProVerif tool [Blanchet]
- powerful proof techniques for hand proofs
- successfully used to analyze a variety of security protocols

Modelling messages as terms

First order terms built over a signature $\mathcal{F}$ (finite set of function symbols), an infinite set of names and an infinite set of variables

t	::=		term
	$\mid$	x	variable x
	$\mid$	n	name n
	$\mid$	$f(t_1, \ldots, t_k)$	application of symbol $f \in \mathcal{F}$

Example: Let $\mathcal{F} = \{\mathsf{enc}(\cdot,\cdot), \mathsf{dec}(\cdot,\cdot), \langle \cdot,\cdot \rangle, \pi_1(\cdot), \pi_2(\cdot)\}$.

$\mathsf{enc}(\langle s_1, a\rangle, k)$ $\quad$ $\mathsf{dec}(\mathsf{enc}(x, y), y)$ $\quad$ $\pi_1(enc(s, k))$

Modelling messages as terms

First order terms built over a signature $\mathcal{F}$ (finite set of function symbols), an infinite set of names and an infinite set of variables

t	::=		term
	|	x	variable x
	|	n	name n
	|	$f(t_1, \ldots, t_k)$	application of symbol $f \in \mathcal{F}$

Example: Let $\mathcal{F} = \{\mathsf{enc}(\cdot,\cdot), \mathsf{dec}(\cdot,\cdot), \langle\cdot,\cdot\rangle, \pi_1(\cdot), \pi_2(\cdot)\}$.

$\mathsf{enc}(\langle s_1, a\rangle, k)$ $\quad$ $\mathsf{dec}(\mathsf{enc}(x, y), y)$ $\quad$ $\pi_1(enc(s, k))$

Term algebra is equipped with an equational theory induced by a finite set of equations

Example: Define E by $\mathsf{dec}(\mathsf{enc}(x, y), y) = x$, $\pi_1(\langle x, y\rangle) = x$, $\pi_2(\langle x, y\rangle) = y$
Then we have that $\pi_1\mathsf{dec}(\mathsf{enc}(\langle s_1, a\rangle, k)) =_E s$.

The applied π-calculus on an example

Syntax:

$$P = \nu s, k.(\mathrm{out}(c_1, enc(s, k)) \mid \mathrm{in}(c_1, y).\mathrm{out}(c_2, dec(y, k))).$$

Special processes: active substitutions $P \mid \{^{M}/_{x}\}$

The applied π-calculus on an example

Syntax:

$$P = \nu s, k.(\text{out}(c_1, enc(s,k)) \mid \text{in}(c_1, y).\text{out}(c_2, dec(y,k))).$$

Special processes: active substitutions $P \mid \{^M/_x\}$

Semantics:

- Operational semantics $\rightarrow$: closed by structural equivalence ($\equiv$) and application of evaluation contexts such that

Comm	$\text{out}(a,x).P \mid \text{in}(a,x).Q \rightarrow P \mid Q$
Then	if $M = M$ then P else $Q \rightarrow P$
Else	if $M = N$ then P else $Q \rightarrow Q \quad (M \neq N)$

The applied π-calculus on an example

Syntax:

$$P = \nu s, k.(\mathsf{out}(c_1, enc(s, k)) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, dec(y, k))).$$

Special processes: active substitutions $P \mid \{^M/_x\}$

Semantics:

- Operational semantics $\rightarrow$: closed by structural equivalence ($\equiv$) and application of evaluation contexts such that

Comm	$\mathsf{out}(a, x).P \mid \mathsf{in}(a, x).Q \rightarrow P \mid Q$
Then	if $M = M$ then P else $Q \rightarrow P$
Else	if $M = N$ then P else $Q \rightarrow Q \quad (M \neq N)$

Example: $P \;\rightarrow\; \nu s, k.\mathsf{out}(c_2, s)$

The applied π-calculus on an example (2)

- Labeled operational semantics $\xrightarrow{\alpha}$

  Labelled transitions where α is either $\mathsf{in}(c, M)$, $(\nu c').\mathsf{out}(c, c')$ or $(\nu x.)\mathsf{out}(c, x)$

  Example:

$$\nu a, \nu k.out(c, enc(a, k)).P \xrightarrow{\nu x.out(c,x)} P \mid \{^{enc(a,k)}/_{x}\}$$

  Allows processes to communicate with an unspecified environment

  Output is done by reference and creates active substitutions

The applied π-calculus on an example (2)

- Labeled operational semantics $\xrightarrow{\alpha}$

  Labelled transitions where α is either in(c, M), $(\nu c')$.out(c, c') or $(\nu x.)$out(c, x)

  Example:

  $$\nu a, \nu k.out(c, enc(a, k)).P \xrightarrow{\nu x.out(c,x)} P \mid \{^{enc(a,k)}/_x\}$$

  Allows processes to communicate with an unspecified environment

  Output is done by reference and creates active substitutions

- Frames

  The frame of a process $\phi(A)$ is build from the process restrictions and active substitutions

  Approximation of the process accounting for the static knowledge exposed to the environment

Deducing secrets

Frame

A frame is a process of the form $\nu\tilde{n}.(\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_n}/_{x_n}\})$.

Example

$P = \nu s, k.(\text{out}(c_2, s) \mid \{enc(s, k)/x_1\}$ $\qquad \phi(P) = \nu s, k.\{enc(s, k)/x_1\}$

Deducibility ($\vdash$)

$\varphi \vdash s$ when

- there exists M, such that $M\sigma =_E t$, where $\varphi = \nu\tilde{n}.\sigma$ and M does not use the names $\tilde{n}$

Deducing secrets

Frame

A frame is a process of the form $\nu\tilde{n}.(\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_n}/_{x_n}\})$.

Example

$$P = \nu s, k.(\mathrm{out}(c_2, s) \mid \{enc(s,k)/x_1\} \qquad \phi(P) = \nu s, k.\{enc(s,k)/x_1\}$$

Deducibility ($\vdash$)

$\varphi \vdash s$ when

- there exists M, such that $M\sigma =_E t$, where $\varphi = \nu\tilde{n}.\sigma$ and M does not use the names $\tilde{n}$

Example 1: $\nu s.\nu k.(\{^{enc(s,k)}/_x\} \mid \{^k/_y\}) \vdash s$

as $dec(x,y)\sigma = dec(enc(s,k),k) =_E s$.

Frame

A frame is a process of the form $\nu\tilde{n}.(\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_n}/_{x_n}\})$.

Example

$$P = \nu s, k.(\mathrm{out}(c_2, s) \mid \{enc(s, k)/x_1\} \qquad \phi(P) = \nu s, k.\{enc(s, k)/x_1\}$$

Deducibility ($\vdash$)

$\varphi \vdash s$ when

- there exists M, such that $M\sigma =_E t$, where $\varphi = \nu\tilde{n}.\sigma$ and M does not use the names $\tilde{n}$

Example 2: $\nu a.\nu k.\{^{enc(a,k)}/_x\} \not\vdash a$

Deducing secrets

Frame

A frame is a process of the form $\nu\tilde{n}.(\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_n}/_{x_n}\})$.

Example

$$P = \nu s, k.(\mathrm{out}(c_2, s) \mid \{enc(s, k)/x_1\} \qquad \phi(P) = \nu s, k.\{enc(s, k)/x_1\}$$

Deducibility ($\vdash$)

$\varphi \vdash s$ when

- there exists M, such that $M\sigma =_E t$, where $\varphi = \nu\tilde{n}.\sigma$ and M does not use the names $\tilde{n}$

A process P ensures the secret of s if for any P' such that $P \xrightarrow{(\alpha)} {}^* P'$ we have that $\phi(P') \nvdash s$.

Static equivalence on frames – passive attacker

Static equivalence on frames ($\approx_s$)

$\varphi \approx_s \psi$ when

- $dom(\varphi) = dom(\psi)$ (the frames coincide on unrestricted variables),
- for all terms U, V, $(U =_E V)\varphi$ iff $(U =_E V)\psi$

Static equivalence on frames – passive attacker

Static equivalence on frames ($\approx_s$)

$\varphi \approx_s \psi$ when

- $dom(\varphi) = dom(\psi)$ (the frames coincide on unrestricted variables),
- for all terms U, V, $(U =_E V)\varphi$ iff $(U =_E V)\psi$

Example 1: $\nu k.(\{^{enc(a,k)}/_x\} \mid \{^k/_y\}) \not\approx_s \nu k.(\{^{enc(b,k)}/_x\} \mid \{^k/_y\})$

because of the test $dec(x, y) = a$

Static equivalence on frames – passive attacker

Static equivalence on frames ($\approx_s$)

$\varphi \approx_s \psi$ when

- $dom(\varphi) = dom(\psi)$ (the frames coincide on unrestricted variables),
- for all terms U, V, $(U =_E V)\varphi$ iff $(U =_E V)\psi$

Example 1: $\nu k.(\{^{enc(a,k)}/_x\} \mid \{^k/_y\}) \not\approx_s \nu k.(\{^{enc(b,k)}/_x\} \mid \{^k/_y\})$

because of the test $dec(x, y) = a$

Example 2: $\nu k.\{^{enc(a,k)}/_x\} \approx_s \nu k.\{^{enc(b,k)}/_x\}$

Static equivalence on frames – passive attacker

Static equivalence on frames ($\approx_s$)

$\varphi \approx_s \psi$ when

- $dom(\varphi) = dom(\psi)$ (the frames coincide on unrestricted variables),
- for all terms U, V, $(U =_E V)\varphi$ iff $(U =_E V)\psi$

Example 1: $\nu k.(\{^{enc(a,k)}/_x\} \mid \{^k/_y\}) \not\approx_s \nu k.(\{^{enc(b,k)}/_x\} \mid \{^k/_y\})$

because of the test $dec(x, y) = a$

Example 2: $\nu k.\{^{enc(a,k)}/_x\} \approx_s \nu k.\{^{enc(b,k)}/_x\}$

Formalizes the idea that an attacker cannot distinguish two frames

Equivalence of processes

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Intuition:

An adversary cannot distinguish two processes, even if it can arbitrarily interact with them

Usefull for modelling privacy properties: more on this on Friday in Stéphanie's talk

Part III

Analysing the protocol by Fujioka, Okamoto, Ohta

[KremerRyan'05]

FOO'92 : "unusual" cryptographic primitives

- Anonymous channels
  - Implemented using MixNets, Onion Routing, ...
- Commitment
  - To commit to m, I invent a new random r and send you commit(m, r).
  - Later, I'll send you r, which you can use to reveal m.
  - It is binding: one cannot find r', such that the commitment opens correctly to m'
- Blind signatures
  - I want you to sign m but I don't want you to see its value.
  - I send you blind(m, r). You sign it.
  - I use r to extract your signature on m.

Part 1

Voter V

$x = \text{commit}(v, r)$

$e = \text{blind}(x, b)$

V $\longrightarrow$ A : $\sigma_V(e)$

check V is legitimate

A $\longrightarrow$ V : $\sigma_A(e)$

$\text{unblind}(\sigma_A(e), b) = \sigma_A(x)$

Admin A

FOO 92 protocol [FujiokaOkamotoOhta92]

Part 1

Voter V

$x = \text{commit}(v, r)$

$e = \text{blind}(x, b)$

$V \longrightarrow A \; : \; \sigma_V(e)$

check V is legitimate

$A \longrightarrow V \; : \; \sigma_A(e)$

$\text{unblind}(\sigma_A(e), b) = \sigma_A(x)$

Admin A

Part 2

Voter V

$V \longrightarrow C \; : \; \sigma_A(x)$

enter $(\ell, \sigma_A(x))$ into list

Collector C

FOO 92 protocol [FujiokaOkamotoOhta92]

Part 1

$x = \text{commit}(v, r)$

$e = \text{blind}(x, b)$

Voter V

V $\longrightarrow$ A : $\sigma_V(e)$

check V is legitimate

A $\longrightarrow$ V : $\sigma_A(e)$

$\text{unblind}(\sigma_A(e), b) = \sigma_A(x)$

Admin A

Part 2

Voter V

V $\longrightarrow$ C : $\sigma_A(x)$

enter $(\ell, \sigma_A(x))$ into list

Collector C

Part 3

Voter V

publish list $(\ell_i, \sigma_A(x_i))$

V $\longrightarrow$ C : ℓ_i, r

open x using r

publish v

Collector C

Signature and equational theory

Signature

`commit/2.`	*commitment*
`open/2.`	*open commitment*
`sign/2.`	*digital signature*
`checksign/2.`	*open digital signature*
`pk/1.`	*get public key from private key*
`host/1.`	*get host from public key*
`getpk/1.`	*get public key from host*
`blind/2.`	*blinding*
`unblind/2.`	*undo blinding*

Equational theory

```
            open(commit(m,r),r)  =  m.
             getpk(host(pubkey)) =  pubkey.
     checksign(sign(m,sk),pk(sk)) = m.
          unblind(blind(m,r),r)  =  m.
unblind(sign(blind(m,r),sk),r)   =  sign(m,sk).
```

Voter process

- ascii version of applied π-calculus (input to ProVerif tool)
- Hypothesis: All channels are anonymous, unless identification is explicitly given in the message

```
processV =
new blinder; new r;
let blindedcommitedvote=blind(commit(v,r),blinder) in
out(ch,(hostv,sign(blindedcommitedvote,skv)));
in(ch,m2);
let blindedcommitedvote0=checksign(m2,pka) in
if blindedcommitedvote0=blindedcommitedvote then
let signedcommitedvote=unblind(m2,blinder) in
out(ch,signedcommitedvote);
in(ch,(l,=signedcommitedvote));
out(ch,(l,r)).
```

The other processes

- Admin and collector processes similar
- Main process puts everything together:

```
new ska; new skv;
new privCh;
let pka=pk(ska) in
let hosta = host(pka) in
let pkv=pk(skv) in
let hostv=host(pkv) in
out(ch,pka); out(ch,hosta);
out(ch,pkv); out(ch,hostv);
((out(privCh,pkv); out(privCh,pk(ski))) |
(!processV)|(!processA)|(!processC))
```

Blanchet's ProVerif tool

- Designed and implemented by Bruno Blanchet
  http://www.proverif.ens.fr/
- Input is given in the applied π-calculus
- Expressive: can model algebraic properties of the crypto, via rewrite rules and equations
- Analyses secrecy/reachability properties of protocols as well as equivalence properties
- Applied π-calculus is translated into Horn clauses, describing acquisition of knowledge by the attacker
- Unbounded number of sessions
- Sound, but not complete (false attacks are possible)
- Termination not guaranteed

Fairness

Fairness ensures that you cannot obtain exit polls, i.e. early results

Can be modeled as a secrecy property: the vote of a honest voter stays secret until the opening phase

Even a corrupt administrator cannot learn votes : modeled by outputting the admin's private key

No need for a corrupt collector (collector never uses his private key)

Proofs automated by ProVerif

Fairness using stronger notions of secrecy

Modeling fairness as deducibility may be too weak

Only few possible values for votes make elections particularly vulnerable to offline guessing attacks, aka dictionnary attacks

Example: $\varphi = \{^{enc(v,pk)}/_{x}\}$ where $v \in \{0, 1\}$

Fairness using stronger notions of secrecy

Modeling fairness as deducibility may be too weak

Only few possible values for votes make elections particularly vulnerable to offline guessing attacks, aka dictionnary attacks

Example: $\varphi = \{^{enc(v,pk)}/_x\}$ where $v \in \{0, 1\}$

Offline guessing attacks can be modelled using static equivalence

$$\nu v.(\varphi \mid \{^{v}/_x\} \approx_s \nu v.(\varphi \mid \nu v'.\{^{v'}/_x\})$$

Intuition:
the attacker cannot distinguish the right guess v from a wrong guess v'

Fairness using stronger notions of secrecy

Modeling fairness as deducibility may be too weak

Only few possible values for votes make elections particularly vulnerable to offline guessing attacks, aka dictionnary attacks

Example: $\varphi = \{^{enc(v,pk)}/_x\}$ where $v \in \{0,1\}$

Offline guessing attacks can be modelled using static equivalence

$$\nu v.(\varphi \mid \{^{v}/_x\} \approx_s \nu v.(\varphi \mid \nu v'.\{^{v'}/_x\})$$

Intuition:
the attacker cannot distinguish the right guess v from a wrong guess v'

We can verify an even stronger property: strong secrecy [Blanchet'04]

$$\forall M, N. \qquad P\{^M/_v\} \approx_o P\{^N/_v\}$$

Fairness using stronger notions of secrecy

Modeling fairness as deducibility may be too weak

Only few possible values for votes make elections particularly vulnerable to offline guessing attacks, aka dictionnary attacks

Example: $\varphi = \{^{enc(v,pk)}/_x\}$ where $v \in \{0,1\}$

Offline guessing attacks can be modelled using static equivalence

$$\nu v.(\varphi \mid \{^v/_x\} \approx_s \nu v.(\varphi \mid \nu v'.\{^{v'}/_x\})$$

Intuition:
the attacker cannot distinguish the right guess v from a wrong guess v'

We can verify an even stronger property: strong secrecy [Blanchet'04]

$$\forall M, N. \qquad P\{^M/_v\} \approx_o P\{^N/_v\}$$

All of these properties have been automatically checked using ProVerif

Eligibility

- Only legitimate voters can vote and only once
- Do not register intruder and require to vote a challenge vote

```
Modified collector
[...]
new attack;
if voteV=challengeVote then
    out(ch, attack)
else
    out(ch, voteV).
```

- Proof done by ProVerif
- Corrupt administrator: trivial attack found by Proverif

Outline

Lecture 1: Introduction to protocol analysis in applied pi

$\longrightarrow$ today

Lecture 2: Formalisation and verification of security properties

Part I: Privacy-type properties
(based on joint work with M. Ryan)

Part II: Verifiability properties
(based on joint work with M. Ryan and B. Smyth)

$\longrightarrow$ on Friday

Outline

Lecture 2: Formalisation and verification of security properties

Part I: Privacy-type properties
(based on joint work with M. Ryan)

Stéphanie Delaune

Part II: Verifiability properties
(based on joint work with M. Ryan and B. Smyth)

Steve Kremer

Privacy-type security properties

Privacy: the fact that a particular voter voted in a particular way is not revealed to anyone

Receipt-freeness: a voter cannot prove that she voted in a certain way (this is important to protect voters from coercion)

Coercion-resistance: same as receipt-freeness, but the coercer interacts with the voter during the protocol, (*e.g.* by preparing messages)

How can we express privacy?

Classically modeled as an equivalence between two slightly different processes P_1 and P_2.

In applied pi calculus, such an equivalence can be:

1. Testing equivalence ($P_1 \approx_t P_2$)
2. Observational equivalence ($P_1 \approx_o P_2$)

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Example 1: $\mathrm{out}(a, s) \not\approx_t \mathrm{out}(a, s')$

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Example 1: $\mathsf{out}(a, s) \not\approx_t \mathsf{out}(a, s')$

$\longrightarrow$ $C[\_] = \mathsf{in}(a, x).\mathsf{if}\ x = s\ \mathsf{then}\ \mathsf{out}(c, ok) \mid \_$

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Example 2:

$$\nu s.\mathsf{out}(a, \mathsf{enc}(s, k)).\mathsf{out}(a, \mathsf{enc}(s, k'))$$
$$\not\approx_t$$
$$\nu s, s'.\mathsf{out}(a, \mathsf{enc}(s, k)).\mathsf{out}(a, \mathsf{enc}(s', k'))$$

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Example 2:

$$\nu s.\mathsf{out}(a, \mathsf{enc}(s, k)).\mathsf{out}(a, \mathsf{enc}(s, k'))$$
$$\not\approx_t$$
$$\nu s, s'.\mathsf{out}(a, \mathsf{enc}(s, k)).\mathsf{out}(a, \mathsf{enc}(s', k'))$$

$\longrightarrow C[\_] = \mathsf{in}(a, x).\mathsf{in}(a, y).\mathsf{if}\ (\mathsf{dec}(x, k) = \mathsf{dec}(y, k'))\ \mathsf{then}\ \mathsf{out}(c, ok) \mid \_$

Testing equivalence

Testing equivalence ($P \approx_t Q$)

for all closing evaluation contexts $C[\_]$, we have that:

$$C[P] \Downarrow c \text{ if, and only if, } C[Q] \Downarrow c.$$

$\longrightarrow$ $P \Downarrow c$ when P can send a message on the channel c.

Example 3: $\nu s.\mathsf{out}(a, s) \approx_t \nu s.\nu k.\mathsf{out}(a, \mathsf{enc}(s, k))$

Observational equivalence

Observational equivalence ($\approx_o$)

The largest symmetric relation $\mathcal{R}$ on processes such that $P \; \mathcal{R} \; Q$ implies

1. if $P \Downarrow c$, then $Q \Downarrow c$,
2. if $P \rightarrow^* P'$, then $Q \rightarrow^* Q'$ and $P' \; \mathcal{R} \; Q'$ for some Q',
3. $C[A] \; \mathcal{R} \; C[B]$ for all closing evaluation contexts $C[\_]$.

Observational equivalence

Observational equivalence ($\approx_o$)

The largest symmetric relation $\mathcal{R}$ on processes such that $P \ \mathcal{R} \ Q$ implies

1. if $P \Downarrow c$, then $Q \Downarrow c$,
2. if $P \rightarrow^* P'$, then $Q \rightarrow^* Q'$ and $P' \ \mathcal{R} \ Q'$ for some Q',
3. $C[A] \ \mathcal{R} \ C[B]$ for all closing evaluation contexts $C[\_]$.

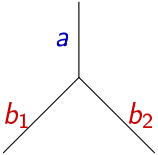
Lemma

We have that: $P \approx_o Q \implies P \approx_t Q$

May testing vs observational equivalence

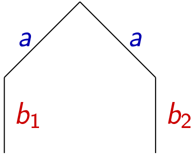
In general, testing equivalence does not imply observational equivalence.

Example:



Process P

$\mathsf{out}(c, a).(\mathsf{out}(c, b_1) + \mathsf{out}(c, b_2))$



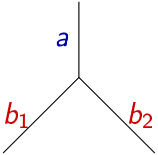
Process Q

$\mathsf{out}(c, a).\mathsf{out}(c, b_1) + \mathsf{out}(c, a).\mathsf{out}(c, b_2)$

May testing vs observational equivalence

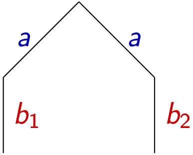
In general, testing equivalence does not imply observational equivalence.

Example:



Process P

$\mathrm{out}(c, a).(\mathrm{out}(c, b_1) + \mathrm{out}(c, b_2))$



Process Q

$\mathrm{out}(c, a).\mathrm{out}(c, b_1) + \mathrm{out}(c, a).\mathrm{out}(c, b_2)$

$\approx_t \; = \; \approx_o$?

On determinate processes, the two notions coincide.

Outline

Formalising Privacy

Formalisation of privacy

Classically modeled as equivalences between two slightly different processes P_1 and P_2, but

- changing the identity

$$S[V_A\{^a/_v\}] \approx S[V_B\{^a/_v\}]$$

  does not work, as identities are revealed

- changing the vote

$$S[V_A\{^a/_v\}] \approx S[V_A\{^b/_v\}]$$

  does not work, as the votes are revealed at the end

Formalisation of privacy

Classically modeled as equivalences between two slightly different processes P_1 and P_2, but

- changing the identity

$$S[V_A\{^a/_v\}] \approx S[V_B\{^a/_v\}]$$

  does not work, as identities are revealed

- changing the vote

$$S[V_A\{^a/_v\}] \approx S[V_A\{^b/_v\}]$$

  does not work, as the votes are revealed at the end

Solution

Consider 2 honest voters and swap their votes.

Formal Definition of privacy

Definition (S. Kremer & M. Ryan, 2005)

A voting protocol respects privacy if

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}].$$

Formal Definition of privacy

Definition (S. Kremer & M. Ryan, 2005)

A voting protocol respects privacy if

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}].$$

Some remarks

- robust in case of an unanimous scrutin
- flexible w.r.t. authorities required to be honest

Limitation: This definition does not say anything about the privacy of a voter who wants to nullify her vote.

Example 1

Voter process

$$V = \mathsf{out}(ch, \{v\}_{\mathsf{pub}(S)})$$

What about privacy?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \stackrel{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

Example 1

Voter process

$$V = \mathrm{out}(ch, \{v\}_{\mathrm{pub}(S)})$$

What about privacy?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \stackrel{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

i.e.

$$\mathrm{out}(ch, \{a\}_{\mathrm{pub}(S)}) \mid \mathrm{out}(ch, \{b\}_{\mathrm{pub}(S)}) \stackrel{?}{\approx} \mathrm{out}(ch, \{b\}_{\mathrm{pub}(S)}) \mid \mathrm{out}(ch, \{a\}_{\mathrm{pub}(S)})$$

Example 1

Voter process

$$V = \text{out}(ch, \{v\}_{\text{pub}(S)})$$

What about privacy?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \overset{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

i.e.

$$\text{out}(ch, \{a\}_{\text{pub}(S)}) \mid \text{out}(ch, \{b\}_{\text{pub}(S)}) \overset{?}{\approx} \text{out}(ch, \{b\}_{\text{pub}(S)}) \mid \text{out}(ch, \{a\}_{\text{pub}(S)})$$

$\longrightarrow$ The equivalence holds.

Some remarks:

- *ch* is assumed to be an anomymous channel;
- the server is not assumed to be honest.

Example 2

Voter process

$$V(Id) = \mathsf{out}(ch, \langle Id, \{v\}_{\mathsf{pub}(S)} \rangle)$$

What about privacy (for someone who does not know priv(S))?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \overset{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

Example 2

Voter process

$$V(Id) = \mathrm{out}(ch, \langle Id, \{v\}_{\mathrm{pub}(S)} \rangle)$$

What about privacy (for someone who does not know priv(S))?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \overset{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

i.e.

$$\mathrm{out}(ch, \langle A, \{a\}_{\mathrm{pub}(S)} \rangle) \mid \mathrm{out}(ch, \langle B, \{b\}_{\mathrm{pub}(S)} \rangle)$$
$$\overset{?}{\approx}$$
$$\mathrm{out}(ch, \langle A, \{b\}_{\mathrm{pub}(S)} \rangle) \mid \mathrm{out}(ch, \langle B, \{a\}_{\mathrm{pub}(S)} \rangle)$$

Example 2

Voter process

$$V(Id) = \mathrm{out}(ch, \langle Id, \{v\}_{\mathrm{pub}(S)}\rangle)$$

What about privacy (for someone who does not know priv(S))?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \overset{?}{\approx} V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

i.e.

$$\mathrm{out}(ch, \langle A, \{a\}_{\mathrm{pub}(S)}\rangle) \mid \mathrm{out}(ch, \langle B, \{b\}_{\mathrm{pub}(S)}\rangle)$$
$$\overset{?}{\approx}$$
$$\mathrm{out}(ch, \langle A, \{b\}_{\mathrm{pub}(S)}\rangle) \mid \mathrm{out}(ch, \langle B, \{a\}_{\mathrm{pub}(S)}\rangle)$$

$\longrightarrow$ The equivalence does not hold (with deterministic encryption).

$\longrightarrow$ The equivalence holds with probabilistic encryption.

Example: Fujioka *et al.* protocol (1992)

First Phase:
the voter gets a "token" from the administrator.

1. $V \to A$: $V, sign(blind(commit(vote, r), b), V)$
2. $A \to V$: $sign(blind(commit(vote, r), b), A)$

Voting phase:

3. $V \to C$: $commit(vote, r), \; sign(commit(vote, r), A)$
4. $C \to$: $l, \; commit(vote, r), \; sign(commit(vote, r), A)$

Counting phase:

5. $V \to C$: l, r
6. C publishes the outcome of the vote

What about privacy?

$$V_A\{^a/_v\} \mid V_B\{^b/_v\} \approx V_A\{^b/_v\} \mid V_B\{^a/_v\}$$

Example: Fujioka *et al.* protocol (1992)

Process synchronisation: the protocol is divided into 3 phases
$\longrightarrow$ synchronisation is crucial for privacy to hold.

Example: Fujioka *et al.* protocol (1992)

Process synchronisation: the protocol is divided into 3 phases
$\longrightarrow$ synchronisation is crucial for privacy to hold.

Authorities: privacy holds even if the authorities are corrupted

- we do not require any private keys to be secret;
- we have just to ensure that both voters use the same public key for the administrator.

Formalising
Receipt-Freeness

Receipt-freeness: Leaking secrets to the coercer

To model receipt-freeness we need to specify that a coerced voter cooperates with the coercer by leaking secrets on a channel ch

We denote by V^{ch} the process built from the process V as follows:

- $0^{ch} \mathrel{\widehat{=}} 0$,
- $(P \mid Q)^{ch} \mathrel{\widehat{=}} P^{ch} \mid Q^{ch}$,
- $(\nu n.P)^{ch} \mathrel{\widehat{=}} \nu n.\mathrm{out}(ch, n).P^{ch}$,
- $(\mathrm{in}(u, x).P)^{ch} \mathrel{\widehat{=}} \mathrm{in}(u, x).\mathrm{out}(ch, x).P^{ch}$,
- $(\mathrm{out}(u, M).P)^{ch} \mathrel{\widehat{=}} \mathrm{out}(u, M).P^{ch}$,
- ...

We denote by $V^{\backslash out(ch,\cdot)} \mathrel{\widehat{=}} \nu ch.(V \mid !\mathrm{in}(ch, x))$.

Receipt-freeness

Definition (S. Delaune, S. Kremer & M. Ryan, 2006)

A voting protocol is receipt-free if there exists a process V', satisfying

- $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$,
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Receipt-freeness

Definition (S. Delaune, S. Kremer & M. Ryan, 2006)

A voting protocol is receipt-free if there exists a process V', satisfying

- $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$,
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Limitations:

- This definition does not take into account randomization and forced-abstention attacks.

Example 1

Voter process

$$V = \mathrm{out}(ch, \{v\}_{\mathrm{pub}(S)})$$

What about receipt-freeness?

i.e. Does there exists V' such that

1. $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$,
2. $V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\} \approx V' \mid V_B\{^c/_v\}$.

Example 1

Voter process

$$V = \mathrm{out}(ch, \{v\}_{\mathrm{pub}(S)})$$

What about receipt-freeness?

i.e. Does there exists V' such that

1. $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$,
2. $V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\} \approx V' \mid V_B\{^c/_v\}$.

The process $V_A\{^a/_v\}$ satisfies the two requirements.

1. $V_A\{^a/_v\}^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$,
2. $V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\} \approx V_A\{^a/_v\} \mid V_B\{^c/_v\}$.

$\longrightarrow$ Receipt-freeness holds.

Example 2 (with probabilistic encryption)

Voter process

$$V(Id) = \nu r.\mathrm{out}(ch, \langle Id, \{v\}^{r}_{\mathrm{pub}(S)}\rangle)$$

What about receipt-freeness?

Example 2 (with probabilistic encryption)

Voter process

$$V(Id) = \nu r.\mathrm{out}(ch, \langle Id, \{v\}^{r}_{\mathrm{pub}(S)} \rangle)$$

What about receipt-freeness?

$\longrightarrow$ Receipt-freeness does not hold: r can be used as a receipt.

We have that:

$$V_A\{^c/_v\}^{chc} = \nu r.\mathrm{out}(chc, r).\mathrm{out}(ch, \langle A, \{c\}^{r}_{\mathrm{pub}(S)} \rangle).$$

Example: Fujioka *et al.* protocol (1992)

First Phase:

the voter gets a “token” from the administrator.

1.	$V \to A$	:	$V, sign(blind(commit(vote, r), b), V)$
2.	$A \to V$	:	$sign(blind(commit(vote, r), b), A)$

Voting phase:

3.	$V \to C$	:	$commit(vote, r),\ sign(commit(vote, r), A)$
4.	$C \to$	:	$l,\ commit(vote, r),\ sign(commit(vote, r), A)$

Counting phase:

5.	$V \to C$	:	l, r
6.	C publishes the outcome of the vote		

What about receipt-freenes?

Example: Fujioka *et al.* protocol (1992)

This protocol is not receipt-free and it was not designed with receipt-freeness in mind.

$\longrightarrow$ the blinding factor b_A, the commitment key r_A, and the private key of the voter can be used as a receipt

Example: Fujioka *et al.* protocol (1992)

This protocol is not receipt-free and it was not designed with receipt-freeness in mind.

$\longrightarrow$ the blinding factor b_A, the commitment key r_A, and the private key of the voter can be used as a receipt

How can we ensure receipt-freeness ?

1. reencryption mechanism
2. trapdoor commitment scheme
   $\longrightarrow$ not always sufficient to ensure coercion-resistance

Receipt-freeness implies privacy

Proposition

If a voting protocol is receipt-free then it also respects privacy (for the same context S).

Receipt-freeness implies privacy

Proposition

If a voting protocol is receipt-free then it also respects privacy (for the same context S).

Proof. By hypothesis, there exists a process V', such that

- $V'^{\setminus out(chc,\cdot)} \approx V_A\{^a/_v\}$, and
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Receipt-freeness implies privacy

Proposition

If a voting protocol is receipt-free then it also respects privacy (for the same context S).

Proof. By hypothesis, there exists a process V', such that

- $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$, and
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Apply the evaluation context $\nu chc.(\ \_\ \mid !\mathrm{in}(chc, x))$ on both sides:

$$S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}]^{\backslash out(chc,\cdot)} \approx S[V' \mid V_B\{^c/_v\}]^{\backslash out(chc,\cdot)}$$

Receipt-freeness implies privacy

Proposition

If a voting protocol is receipt-free then it also respects privacy (for the same context S).

Proof. By hypothesis, there exists a process V', such that

- $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$, and
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Apply the evaluation context $\nu chc.(\ \_\ \mid !\mathrm{in}(chc, x))$ on both sides:

$$S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}]^{\backslash out(chc,\cdot)} \approx S[V' \mid V_B\{^c/_v\}]^{\backslash out(chc,\cdot)}$$

Then, we show that we can push $\backslash out(chc, \cdot)$ inside:

$$S[(V_A\{^c/_v\}^{chc})^{\backslash out(chc,\cdot)} \mid V_B\{^a/_v\}] \approx S[V'^{\backslash out(chc,\cdot)} \mid V_B\{^c/_v\}]$$

Receipt-freeness implies privacy

Proposition

If a voting protocol is receipt-free then it also respects privacy (for the same context S).

Proof. By hypothesis, there exists a process V', such that

- $V'^{\backslash out(chc,\cdot)} \approx V_A\{^a/_v\}$, and
- $S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}]$.

Apply the evaluation context $\nu chc.(\ \_\ \mid !\mathrm{in}(chc,x))$ on both sides:

$$S[V_A\{^c/_v\}^{chc} \mid V_B\{^a/_v\}]^{\backslash out(chc,\cdot)} \approx S[V' \mid V_B\{^c/_v\}]^{\backslash out(chc,\cdot)}$$

Then, we show that we can push $\backslash out(chc,\cdot)$ inside:

$$S[(V_A\{^c/_v\}^{chc})^{\backslash out(chc,\cdot)} \mid V_B\{^a/_v\}] \approx S[V'^{\backslash out(chc,\cdot)} \mid V_B\{^c/_v\}]$$

Thus Privacy holds: $S[V_A\{^c/_v\} \mid V_B\{^a/_v\}] \approx S[V_A\{^a/_v\} \mid V_B\{^c/_v\}]$ □

Formalising Coercion Resistance

Coercion-resistance (1)

Leaking secrets to the coercer V^{c_1,c_2}:

- the coercer will receive the message from the coerced voter V on c_2;
- the coercer will give some prepared messages on c_1.

Coercion-resistance (1)

Leaking secrets to the coercer V^{c_1,c_2}:

- the coercer will receive the message from the coerced voter V on c_2;
- the coercer will give some prepared messages on c_1.

First approximation

There exists V' such that

$$S[V_A\{^?/_v\}^{c_1,c_2} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}].$$

Coercion-resistance (1)

Leaking secrets to the coercer V^{c_1,c_2}:

- the coercer will receive the message from the coerced voter V on c_2;
- the coercer will give some prepared messages on c_1.

First approximation

There exists V' such that

$$S[V_A\{^?/_v\}^{c_1,c_2} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}].$$

Problems:

- This assumes that the coercer will vote c.
- If the coercer votes $c' \neq c$, then the equivalence will not hold.

Coercion-resistance (2)

First approximation

There exists V' such that

$$S[V_A\{^?/_v\}^{c_1,c_2} \mid V_B\{^a/_v\}] \approx S[V' \mid V_B\{^c/_v\}].$$

To get rid of this problem, two possible solutions:

- add some conditions to ensure that the coercer will vote c.
  $\longrightarrow$ our approach (with Steve Kremer and Mark D. Ryan)
  (CSFW'06, Journal of Computer Security'09)
- allow the voter V_B to adapt his choice to counterbalance the vote done by the coerced voter.
  $\longrightarrow$ approach followed by Backes et al. (CSF'08). To achieve this, they rely on an extractor process.

Outline

Verification of equivalence-based properties

How can we establish privacy in e-voting protocols?

$\longrightarrow$ we have to establish equivalence properties between processes

Main difficulties

- quantification over all contexts,
- some specific features (anonymous channel, synchronisation phase, bulletin board)
- quite complexe cryptographic primitives
  $\longrightarrow$ *e.g.* blind signatures, reencryption mechanism, ...

How can we establish privacy in e-voting protocols?

$\longrightarrow$ we have to establish equivalence properties between processes

Main difficulties

- quantification over all contexts,
- some specific features (anonymous channel, synchronisation phase, bulletin board)
- quite complexe cryptographic primitives
  $\longrightarrow$ *e.g.* blind signatures, reencryption mechanism, ...

Manual proofs are quite error-prone.

Existing automated tools designed for secrecy and authentication are not well-suited for verifying e-voting protocols.

Static equivalence on frames - passive attacker

$\longrightarrow$ Intuitively, static equivalence formalizes the idea that an attacker cannot distinguish two sequences of messages

Static equivalence on frames - passive attacker

$\longrightarrow$ Intuitively, static equivalence formalizes the idea that an attacker cannot distinguish two sequences of messages

Example: $\mathsf{E} = \{\mathsf{dec}(\mathsf{enc}(x, y), y) = x\}$

$$\phi_1 \;=\; yes,\; no,\; k,\; \{yes\}_k \quad \text{and} \quad \phi_2 \;=\; yes,\; no\; k,\; \{no\}_k$$

$\longrightarrow$ not statically equivalent, choose $M = \mathsf{dec}(x_4, x_3)$ and $N = x_1$

Results on static equivalence

Decidability results:

- for the class of subterm convergent equational theories;
- for many theories involving an AC operator
  $\longrightarrow$ *e.g.* XOR, Abelian group, ...
- for specific theories used in e-voting, *e.g.* blind signatures, trapdoor bit commitment, re-encryption, ...

Results on static equivalence

Decidability results:

- for the class of subterm convergent equational theories;
- for many theories involving an AC operator
  $\longrightarrow$ *e.g.* XOR, Abelian group, ...
- for specific theories used in e-voting, *e.g.* blind signatures, trapdoor bit commitment, re-encryption, ...

Combination result:

- for disjoint equational theories

Results on static equivalence

Decidability results:

- for the class of subterm convergent equational theories;
- for many theories involving an AC operator
  $\longrightarrow$ *e.g.* XOR, Abelian group, . . .
- for specific theories used in e-voting, *e.g.* blind signatures, trapdoor bit commitment, re-encryption, . . .

Combination result:

- for disjoint equational theories

Existing tools:

- YAPA - Yet Another Protocol Analyser

  `http://www.lsv.ens-cachan.fr/~baudet/yapa/`

- KiSs - Knowledge In Security protocolS

  `http://www.lsv.ens-cachan.fr/~ciobaca/kiss/`

The ProVerif tool (B. Blanchet)

`http://www.proverif.ens.fr/`

Input: processes written in applied pi calculus

Characteristics

- unbounded number of sessions
- primitives given by an equational theory
- security properties: (strong) secrecy, correspondence properties, equivalence properties
- sound but not complete
  $\longrightarrow$ sometimes, the tool reports some false attacks

Limitation

ProVerif tries to establish diff-equivalence (too strong).

Going beyond the ProVerif tool

Let $P(x_1, x_2) = \mathsf{out}(x_1); \mathsf{synch}; \mathsf{out}(x_2)$.

$$P(a, b) \mid P(b, a) \approx P(a, a) \mid P(b, b).$$

$\longrightarrow$ ProVerif fails to establish this equivalence.

To overcome this limitation (Joint with M. Ryan and B. Smyth):

- we propose a transformation to conlude in more cases;
- Then, using ProVerif on the resulting processes, we propose the first automated proof of privacy for the FOO protocol.

Going beyond the ProVerif tool

Let $P(x_1, x_2) = \mathrm{out}(x_1); \mathrm{synch}; \mathrm{out}(x_2)$.

$$P(a, b) \mid P(b, a) \approx P(a, a) \mid P(b, b).$$

$\longrightarrow$ ProVerif fails to establish this equivalence.

To overcome this limitation (Joint with M. Ryan and B. Smyth):

- we propose a transformation to conlude in more cases;
- Then, using ProVerif on the resulting processes, we propose the first automated proof of privacy for the FOO protocol.

Still some limitations:

- some primitives can not be handled, *e.g.* reencryption, trapdoor bit commitment, ...
- unable in general to establish receipt-freeness properties.

Another approach – constraint solving

$\longrightarrow$ bounded number of sessions (*i.e.* processes without replication)

Another approach – constraint solving

$\longrightarrow$ bounded number of sessions (*i.e.* processes without replication)

Step 1: reduction to the problem of checking symbolic equivalence between constraint systems.

$\longrightarrow$ for simple processes — joint work with V. Cortier

$\longrightarrow$ for general processes — joint work with S. Kremer and M. Ryan

- this reduction is sound but not complete.

Another approach – constraint solving

⟶ bounded number of sessions (*i.e.* processes without replication)

Step 1: reduction to the problem of checking symbolic equivalence between constraint systems.

⟶ for simple processes — joint work with V. Cortier

⟶ for general processes — joint work with S. Kremer and M. Ryan

- this reduction is sound but not complete.

Step 2: decision procedures for symbolic equivalence

⟶ several procedures already exist for subterm convergent theories

⟶ we propose another one (for a specific set of primitives) together with an efficient implementation (ADECS tool)

joint work with V. Cheval and H. Comon-Lundh

`http://www.lsv.ens-cachan.fr/~cheval/`

Conclusion for privacy-type properties

Formalising properties in applied pi

- Nice definitions. The quantification on V' in the receipt-freenees property should not be a problem.
- These definitions can be reused to model similar properties in other applications, *e.g.* privacy in Vehicular Ad-hoc NETwork, privacy-type properties in e-auction, ...

Verification in applied pi (of equivalence-based properties)

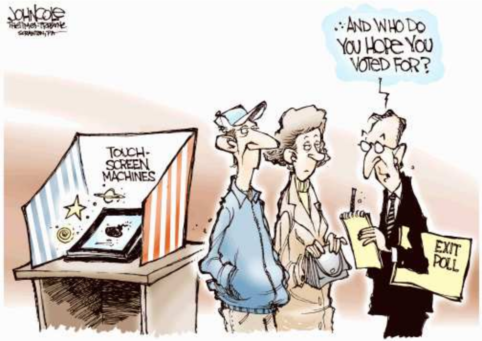
- still an active research area;
- existing results and procedure are quite limited;
- Challenge: a verification tool that performs automated proofs of privacy-type properties in e-voting protocols.

Outline

Lecture 2: Formalisation and verification of security properties

Part II: Verifiability properties
(based on joint work with M. Ryan and B. Smyth)

Steve Kremer



Formalising Verifiability

verifiability

Election verifiability

verifiability
auditability

Election verifiability

end-to-end $\begin{cases} \text{verifiability} \\ \text{auditability} \end{cases}$

Election verifiability

end-to-end { verifiability, auditability }

- Election results can be fully verified by voters/observers
- The software provided by election authorities does not need to be trusted
- The software used to perform the verification can be sourced independently

Election verifiability

Individual verifiability

A voter can check her own vote is included in the tally.

Universal verifiability

Anyone can check that the declared outcome corresponds to the tally.

Eligibility verifiability

Anyone can check that only eligible votes are included in the declared outcome.

Remarks

- Verifiability $\neq$ correctness
- What system components need to be trusted in order to carry out these checks?

We suppose that the protocol involves

- Voter credentials (typically, a public part and a private part for each voter)
- A bulletin board, on which are placed entries corresponding to voter's outputs.

Election verifiability

A protocol satisfies *election verifiability* if there are tests ϕ^{IV}, ϕ^{UV} and ϕ^{EV} satisfying certain acceptability conditions.

Formalizing voting processes

Voting process specification: $\langle V, A \rangle$ where

- V plain process without replication (the voter)
- A a closed evaluation context s.t. $fv(V) = \{v\}$ (the admins)

Voting process

$$\mathrm{VP}_n(s_1, \ldots, s_n) = A[V\{^{s_1}/_v\} \mid \cdots \mid V\{^{s_n}/_v\}]$$

models n voters casting votes for $s_1, \ldots, s_n$

Formalizing voting processes

Voting process specification: $\langle V, A \rangle$ where

- V plain process without replication (the voter)
- A a closed evaluation context s.t. $fv(V) = \{v\}$ (the admins)

Voting process

$$\mathsf{VP}_n(s_1, \ldots, s_n) = A[V\{^{s_1}/_v\} \mid \cdots \mid V\{^{s_n}/_v\}]$$

models n voters casting votes for $s_1, \ldots, s_n$

Voting on Satan's computer

- Extend attacker model to software and hardware, i.e. V, A only represent the trusted parts of the protocol
- Ideally this is only the interaction between the voter and the terminal!
- In practice some parts need to be added, motivated by auditing parts, distributed authorities, ...

We add to the applied pi calculus a $\mathrm{rec}(r, t)$ construct: adds a special entry $\{^t/_r\}$ to frame not accessible to the attacker

the process $R(P)$ is like P but replaces

- νn by $\nu n.\mathrm{rec}(r, n)$ for some fresh r;
- $\mathrm{in}(c, x)$ by $\nu n.\mathrm{rec}(r, x)$ for some fresh r.

Augmented voting process

$$\mathrm{VP}_n^+(s_1, \ldots, s_n) = A[V_1^+ \mid \cdots \mid V_n^+]$$

where $V_i^+ = R(V)\{^{s_i}/_v\}\{^{r_i}/_r \mid r \in rvR(V)\}$

Example: a "raising hands" protocol

Idea: Voter simply outputs her signed vote.

Admin: generates and distributes keys via a private channel d

$$A \mathrel{\hat{=}} \nu d.\nu skA.(\quad !\nu skv.\mathrm{out}(d, skv).\mathrm{out}(c, \mathrm{sign}(skA, \mathrm{pk}(skv))) \;|\; \{^{\mathrm{pk}(skA)}/_{x_{pkA}}\} \;|\; \_\ )$$

Voter: received private key and outputs signed vote

$$V \mathrel{\hat{=}} \mathrm{in}(d, x_{skv}).\mathrm{out}(c, \langle \mathrm{pk}(x_{skv}), \mathrm{sign}(x_{skv}, v)\rangle)$$

Verifiability tests

We require the existence of tests

$$\phi^{IV}(v, w, \tilde{x}, y, \tilde{r}) \qquad \phi^{UV}(\tilde{v}, \tilde{x}, \tilde{y}, \tilde{z}) \qquad \phi^{EV}(\tilde{w}, \tilde{x}, \tilde{y}, \tilde{z})$$

where

- v refers to the vote, $\tilde{v}$ to the declared outcome
- w refers to the public cred., $\tilde{w}$ to all voters' public cred.
- $\tilde{x}$ expected to refer to global election values
- y expected to refer to the voter's ballot on the BB, $\tilde{y}$ to all voters ballots
- $\tilde{r}$ refer to the voter's private data
- $\tilde{z}$ expected to refer to outputs generated for UV and EV

Individual and universal verifiability

A voting specification $\langle V, A\rangle$ satisfies individual and universal verifiability if $\exists \phi^{IV}, \phi^{UV}$ s.t.

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\forall i, j. \quad \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \phi^{IV}\{^{s_j}/_v, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma \Rightarrow i = j \quad (1)$$

$$\phi^{UV}\sigma \wedge \phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma \quad (2)$$

$$\bigwedge_{1\le i\le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \ \Rightarrow\ \tilde{s} \simeq \tilde{v}\sigma \quad (3)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\bigwedge_{1\le i\le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \quad (4)$$

A voting specification $\langle V, A\rangle$ satisfies individual and universal verifiability if $\exists \phi^{IV}, \phi^{UV}$ s.t.

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\forall i, j.\quad \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \phi^{IV}\{^{s_j}/_{v}, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma \Rightarrow i = j \quad (1)$$

$$\phi^{UV}\sigma \wedge \phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma \quad (2)$$

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \ \Rightarrow\ \tilde{s} \simeq \tilde{v}\sigma \quad (3)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\{^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \quad (4)$$

Intuition: a same BB entry y cannot validate ϕ^{IV} for two different voters

Individual and universal verifiability

A voting specification $\langle V, A\rangle$ satisfies individual and universal verifiability if $\exists \phi^{IV}, \phi^{UV}$ s.t.

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \dots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\forall i, j.\quad \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \phi^{IV}\{^{s_j}/_{v}, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma \Rightarrow i = j \quad (1)$$

$$\phi^{UV}\sigma \wedge \phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma \quad (2)$$

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \;\Rightarrow\; \tilde{s} \simeq \tilde{v}\sigma \quad (3)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \dots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\{^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \quad (4)$$

Intuition: for a same election ϕ^{UV} can only validate one outcome

Individual and universal verifiability

A voting specification $\langle V, A\rangle$ satisfies individual and universal verifiability if $\exists \phi^{IV}, \phi^{UV}$ s.t.

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\begin{array}{lr} \forall i,j. \quad \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \phi^{IV}\{^{s_j}/_v, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma \Rightarrow i = j & (1) \\ \phi^{UV}\sigma \wedge \phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma & (2) \\ \bigwedge_{1\le i\le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \;\Rightarrow\; \tilde{s} \simeq \tilde{v}\sigma & (3) \end{array}$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\bigwedge_{1\le i\le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \quad (4)$$

Intuition: if ϕ^{IV}s hold on votes $s_1, \ldots, s_n$ then ϕ^{UV} can only validate this particular outcome

A voting specification $\langle V, A \rangle$ satisfies individual and universal verifiability if $\exists \phi^{IV}, \phi^{UV}$ s.t.

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\begin{array}{lr} \forall i, j. \quad \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}\}\sigma \wedge \phi^{IV}\{^{s_j}/_v, ^{\tilde{r}_j}/_{\tilde{r}}\}\sigma \Rightarrow i = j & (1) \\ \phi^{UV}\sigma \wedge \phi^{UV}\{^{\tilde{v}'}/_{\tilde{v}}\}\sigma \Rightarrow \tilde{v}\sigma \simeq \tilde{v}'\sigma & (2) \\ \bigwedge_{1 \le i \le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \ \Rightarrow \ \tilde{s} \simeq \tilde{v}\sigma & (3) \end{array}$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\bigwedge_{1 \le i \le n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \quad (4)$$

Avoids vaccuous tests where ϕ^{IV}, ϕ^{UV} are false

Example: "raising hands" verifiability

The expected BB entry should be

$$\langle pk(skv), \mathrm{sign}(skv, v)\rangle$$

Define the tests

$$\phi^{IV} \mathrel{\hat{=}} y =_E \langle \mathrm{pk}(r_{skv}), \mathrm{sign}(r_{skv}, v)\rangle \quad \phi^{UV} \mathrel{\hat{=}} \bigwedge_{1 \le i \le n} \mathrm{getmsg}(\pi_2(y_i)) =_E v_i$$

Example: “raising hands” verifiability

The expected BB entry should be

$$\langle pk(skv), \mathrm{sign}(skv, v)\rangle$$

Define the tests

$$\phi^{IV} \mathrel{\hat=} y =_E \langle \mathrm{pk}(r_{skv}), \mathrm{sign}(r_{skv}, v)\rangle \quad \phi^{UV} \mathrel{\hat=} \bigwedge_{1\leq i\leq n} \mathrm{getmsg}(\pi_2(y_i)) =_E v_i$$

Easy proof that individual and universal verifiability hold:

(1) Suppose that $\phi_i^{IV}\sigma$ and $\phi_j^{IV}\sigma$ hold, i.e.,

$$y\sigma =_E \langle \mathrm{pk}(r_{skv_i}\sigma), \mathrm{sign}(r_{skv_i}\sigma, s_i)\rangle \quad y\sigma =_E \langle \mathrm{pk}(r_{skv_j}\sigma), \mathrm{sign}(r_{skv_j}\sigma, s_j)$$

Hence, $r_{skv_i}\sigma =_E r_{skv_j}\sigma$. From the voting process spec. for every reachable σ, $i \neq j$ implies that $r_{skv_i}\sigma \neq_E r_{skv_j}\sigma$.

(2,3) Immediate.

(4) Holds for $C = \_$.

Example: FOO

What are the minimal parts of the protocol to be trusted?

The voting process specification

$$V_{\text{foo}} \triangleq \nu rnd.\text{out}(c, v).\text{out}(c, rnd) \quad \text{and} \quad A_{\text{foo}}[\_] \triangleq \_$$

where *rnd* is intended to be the randomness used for the commitment

The augmented voting process

$$\begin{aligned} \text{VP}_n^+(s_1, \dots, s_n) \;=\; & \nu\, rnd.\text{rec}(r_1, rnd).\text{out}(c, s_1).\text{out}(c, rnd) \mid \dots \mid \\ & \nu\, rnd.\text{rec}(r_n, rnd).\text{out}(c, s_n).\text{out}(c, rnd) \end{aligned}$$

Example: FOO

What are the minimal parts of the protocol to be trusted?

The voting process specification

$$V_{\text{foo}} \mathrel{\hat{=}} \nu rnd.\text{out}(c, v).\text{out}(c, rnd) \quad \text{and} \quad A_{\text{foo}}[\_] \mathrel{\hat{=}} \_$$

where *rnd* is intended to be the randomness used for the commitment

The augmented voting process

$$\begin{aligned} \text{VP}_n^+(s_1, \ldots, s_n) \; = \; & \nu\, rnd.\text{rec}(r_1, rnd).\text{out}(c, s_1).\text{out}(c, rnd) \mid \ldots \mid \\ & \nu\, rnd.\text{rec}(r_n, rnd).\text{out}(c, s_n).\text{out}(c, rnd) \end{aligned}$$

Remark: Other properties need different trust assumptions!

Example: FOO

The expected BB entry should be

$$\langle r, \mathsf{commit}(r, v)\rangle$$

Define the tests

$$\phi^{IV} \triangleq y =_E \langle r, \mathsf{commit}(r, v)\rangle \quad \phi^{UV} \triangleq \bigwedge_{1 \leq i \leq n} v_i =_E \mathsf{open}(\pi_1(y), \pi_2(y))$$

Theorem

$\langle V_{\mathsf{foo}}, A_{\mathsf{foo}}\rangle$ *satisfies individual and universal verifiability.*

Election verifiability

A voting specification $\langle V, A \rangle$ satisfies election verifiability if $\exists \phi^{IV}, \phi^{UV}, \phi^{EV}$ s.t. additionally

Let $X = fv(\phi^{EV}) \backslash dom \mathrm{VP}_n^+(s_1, \ldots, s_n)$

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X \backslash \tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (5)$$

$$\bigwedge_{1 \leq i \leq n} \phi_i^{IV}\sigma \wedge \phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (6)$$

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X \backslash \tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \quad (7)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\bigwedge_{1 \leq i \leq n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \wedge \phi^{EV}\sigma \quad (8)$$

A voting specification $\langle V, A\rangle$ satisfies election verifiability if $\exists \phi^{IV}, \phi^{UV}, \phi^{EV}$ s.t. additionally

Let $X = fv(\phi^{EV}) \backslash dom\mathrm{VP}_{\mathrm{n}}^{+}(s_1, \ldots, s_n)$

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^{*} B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_{x} \mid x \in X \backslash \tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (5)$$

$$\bigwedge_{1 \leq i \leq n} \phi_i^{IV}\sigma \wedge \phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \quad \Rightarrow \quad \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (6)$$

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_{x} \mid x \in X \backslash \tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \quad (7)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^{*} B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\bigwedge_{1 \leq i \leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\{^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \wedge \phi^{EV}\sigma \quad (8)$$

Intuition: given ballots $\tilde{y}\sigma$, provided by the environment, ϕ^{EV} succeeds for a unique list of public credentials

A voting specification $\langle V, A\rangle$ satisfies election verifiability if $\exists \phi^{IV}, \phi^{UV}, \phi^{EV}$ s.t. additionally

Let $X = fv(\phi^{EV}) \backslash dom \mathrm{VP}_{\mathrm{n}}^{+}(s_1, \ldots, s_n)$

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^{+}(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^{*} B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_{x} \mid x \in X \backslash \tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (5)$$

$$\bigwedge_{1\leq i\leq n} \phi_i^{IV}\sigma \wedge \phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \quad \Rightarrow \quad \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (6)$$

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_{x} \mid x \in X \backslash \tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \quad (7)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^{+}(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^{*} B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_{v}, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_{y}\}\{^{y_i}/_{y}\}\sigma \wedge \phi^{UV}\sigma \wedge \phi^{EV}\sigma \quad (8)$$

Intuition: if BB contains the ballots of voters with public cred. $\tilde{w}\sigma$ then ϕ^{EV} only holds on these credentials

A voting specification $\langle V, A\rangle$ satisfies election verifiability if $\exists \phi^{IV}, \phi^{UV}, \phi^{EV}$ s.t. additionally

Let $X = fv(\phi^{EV}) \backslash dom \mathrm{VP}_n^+(s_1, \ldots, s_n)$

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X \backslash \tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \qquad (5)$$

$$\bigwedge_{1 \leq i \leq n} \phi_i^{IV}\sigma \wedge \phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \quad \Rightarrow \quad \tilde{w}\sigma \simeq \tilde{w}'\sigma \qquad (6)$$

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X \backslash \tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \qquad (7)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu \tilde{n}.\sigma$:

$$\bigwedge_{1 \leq i \leq n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \wedge \phi^{EV}\sigma \qquad (8)$$

Intuition: given a set of credentials $\tilde{w}$, only one set of BB entries $\tilde{y}$ are accepted by ϕ^{EV}

Election verifiability

A voting specification $\langle V, A\rangle$ satisfies election verifiability if $\exists \phi^{IV}, \phi^{UV}, \phi^{EV}$ s.t. additionally

Let $X = fv(\phi^{EV}) \backslash dom \mathrm{VP}_n^+(s_1, \ldots, s_n)$

Soundness. $\forall C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X\backslash\tilde{y}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (5)$$

$$\bigwedge_{1\leq i\leq n} \phi_i^{IV}\sigma \wedge \phi^{EV}\{^{\tilde{w}'}/_{\tilde{w}}\}\sigma \Rightarrow \tilde{w}\sigma \simeq \tilde{w}'\sigma \quad (6)$$

$$\phi^{EV}\sigma \wedge \phi^{EV}\{^{x'}/_x \mid x \in X\backslash\tilde{w}\}\sigma \Rightarrow \tilde{y}\sigma \simeq \tilde{y}'\sigma \quad (7)$$

Effectiveness. $\exists C, B$ s.t. $C[\mathrm{VP}_n^+(s_1, \ldots, s_n)] \xrightarrow{(\alpha)}{}^* B$, $\phi(B) \equiv \nu\tilde{n}.\sigma$:

$$\bigwedge_{1\leq i\leq n} \phi^{IV}\{^{s_i}/_v, ^{\tilde{r}_i}/_{\tilde{r}}, ^{y_i}/_y\}\{^{y_i}/_y\}\sigma \wedge \phi^{UV}\sigma \wedge \phi^{EV}\sigma \quad (8)$$

Avoids vaccuous tests where $\phi^{IV}, \phi^{UV}, \phi^{EV}$ are false

Concluding remarks

Election verifiability may ensure the needed transparency for electronic voting to be acceptable

Formal definition of election verifiability as tests with acceptability conditions (generally rather easy to prove)

We have analysed

- FOO: individual and universal verifiable, but not election verifiability
- Helios 2.0: individual and universal verifiable, but not election verifiability
- JCJ/Civitas: verifies election verifiability

Allows for each of the protocols to identify the trust assumptions

Detailed analysis available in [Kremer, Ryan, Smyth, ESORICS 2010]
http://www.bensmyth.com/publications/10tech/CSR-10-06.pdf