Improving Automated Symbolic Analysis of Ballot Secrecy for E-voting Protocols: A Method Based on Sufficient Conditions

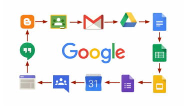
Euro S&P 2019

Lucca Hirschi & Cas Cremers

June 19th, 2019

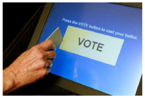
wins (CSF'10)

wins (BlackHat'15)



wins (FMSE'08)

wins (CCS'10)



wins (CSF'11)

Extremely complex setting

- insecure network
- active attacker
- parties running concurrently

Formal methods

- mathematical & exhaustive analysis
- formal guarantees
- automated & mechanised

Symbolic Model

Cryptographic primitives assumed perfect

- primitives modelled as function symbols & equational theory
- *e.g.* 🔒, 🔑 $\longmapsto$ $\mathsf{enc}(\cdot,\cdot), \mathsf{dec}(\cdot,\cdot)$ & $\mathsf{dec}(\mathsf{enc}(m,k),k) = m$

Security protocols

- each party $\longmapsto$ process in a process algebra

Attacker 😈 = network (worst case scenario)

- eavesdrop: he learns all protocol outputs
- injections: he chooses all protocol inputs

Symbolic Model

Cryptographic primitives assumed perfect

- primitives modelled as function symbols & equational theory
- *e.g.* , $\longmapsto$ $\mathsf{enc}(\cdot,\cdot), \mathsf{dec}(\cdot,\cdot)$ & $\mathsf{dec}(\mathsf{enc}(m,k),k) = m$

Security protocols

- each party $\longmapsto$ process in a process algebra

Attacker = network (worst case scenario)

- eavesdrop: he learns all protocol outputs
- injections: he chooses all protocol inputs

Security properties encoded as:

- reachability statements (*e.g.* for secrecy)
- or behavioral equivalence statements (*e.g.* for privacy)

Benefit: high level of automation and tool support!

Symbolic Model

Cryptographic primitives assumed perfect

- primitives modelled as function symbols & equational theory
- *e.g.* , $\longmapsto$ $\mathsf{enc}(\cdot,\cdot), \mathsf{dec}(\cdot,\cdot)$ & $\mathsf{dec}(\mathsf{enc}(m,k),k) = m$

S

A

S

TLS 1.3 (IETF) [S&P17, CCS'17, S&P16]

5G AKA (3GPP) [NDSS'19, CCS'18]

...

- or behavioral equivalence statements (*e.g.* for privacy)

Benefit: high level of automation and tool support!

Symbolic Verification of E-Voting Protocols

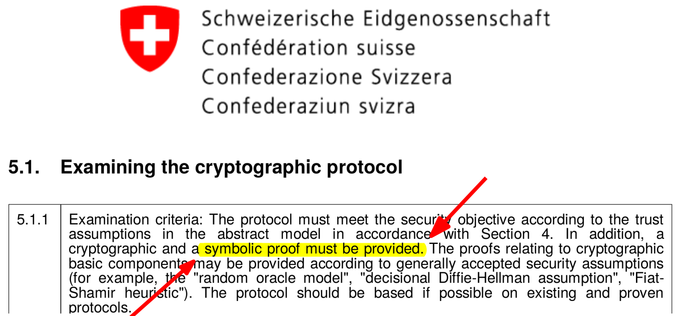
Remote E-Voting Protocols:

- actually used: Estonia, Australia, Switzerland, many smaller elections
- 2 crucial properties: verifiability (of the election) and privacy (of the votes)
- hard to get right + extremely strong threat model

Symbolic Verification of E-Voting Protocols

Remo

- ac
- 2 otes)
- ha


Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

5.1. Examining the cryptographic protocol

5.1.1	Examination criteria: The protocol must meet the security objective according to the trust assumptions in the abstract model in accordance with Section 4. In addition, a cryptographic and a symbolic proof must be provided. The proofs relating to cryptographic basic components may be provided according to generally accepted security assumptions (for example, the "random oracle model", "decisional Diffie-Hellman assumption", "Fiat-Shamir heuristic"). The protocol should be based if possible on existing and proven protocols.

Federal Law!

Symbolic Verification of E-Voting Protocols

Remote E-Voting Protocols:

- actually used: Estonia, Australia, Switzerland, many smaller elections
- 2 crucial properties: verifiability (of the election) and privacy (of the votes)
- hard to get right + extremely strong threat model

This Work: Improve ballot privacy verification technique

- new verification technique based on sufficient conditions
- extends the scope + more efficient

Applied π-Calculus

Model of messages: function symbols & equational theory

Model of protocols: Process algebra

- Process:

P, Q	:=	$\mathsf{in}(c, x).P$	input
	$\vert$	$\mathsf{out}(c, m).P$	output
	$\vert$	$i : P$	phase (can be executed >= phase i)

Applied π-Calculus

Model of messages: function symbols & equational theory

Model of protocols: Process algebra

- Process:

P, Q	:=	$\mathsf{in}(c, x).P$	input
	$\mid$	$\mathsf{out}(c, m).P$	output
	$\mid$	$i : P$	phase (can be executed >= phase i)
	$\mid$	$P \mid Q$	parallel
	$\mid$	$!\ P$	replication
	$\mid$	$\texttt{if}\ Test\ \texttt{then}\ P\ \texttt{else}\ Q$	conditional
	$\mid$	$\mathsf{new}\ X.P$	creation of name
	$\mid$	0	null

Applied π-Calculus

Model of messages: function symbols & equational theory

Model of protocols: Process algebra

- Process:

P, Q	$:=$	$\mathsf{in}(c, x).P$	input
	$\mid$	$\mathsf{out}(c, m).P$	output
	$\mid$	$i : P$	phase (can be executed >= phase i)
	$\mid$	$P \mid Q$	parallel
	$\mid$	$!\ P$	replication
	$\mid$	$\texttt{if}\ Test\ \texttt{then}\ P\ \texttt{else}\ Q$	conditional
	$\mid$	$\mathsf{new}\ X.P$	creation of name
	$\mid$	0	null

- Frame (ϕ): the set of messages revealed to ('s knowledge)
- Configuration: $A = (\mathcal{P}; \phi; j)$ ($\mathcal{P}$ multiset of processes, $j \in \mathbb{N}$)

E-Voting and Privacy

E-Voting Protocol (simplified)

- Roles as processes: Voter: $V($👤, ✉$)$ and authorities: $A \in \mathcal{R}$
- Tally as a function `Tally` over frames
- Honest Trace: a fixed, full, honest execution of $\{V($🎅, ✓$)\} \cup \mathcal{R}$

E-Voting and Privacy

E-Voting Protocol (simplified)

- Roles as processes: Voter: $V($🧑$,$✉$)$ and authorities: $A \in \mathcal{R}$
- Tally as a function `Tally` over frames
- Honest Trace: a fixed, full, honest execution of $\{V($🎅$,\checkmark)\} \cup \mathcal{R}$

Ballot Privacy (simplified)

$V($🎅$,\checkmark) \mid V($🤠$,\times) \mid !\mathcal{A} \;\approx\; V($🎅$,\times) \mid V($🤠$,\checkmark) \mid !\mathcal{A}$

Where $\approx$ is a behavioral equivalence: 👹 cannot tell both sides apart.

"👹 cannot establish meaningful link between a voter and his vote"

E-Voting and Privacy

E-Voting Protocol (simplified)

- Roles as processes: Voter: $V($👤$,$✉$)$ and authorities: $A \in \mathcal{R}$
- Tally as a function `Tally` over frames
- Honest Trace: a fixed, full, honest execution of $\{V($🎅$,✓)\} \cup \mathcal{R}$

Ballot Privacy (simplified)

$$V(🎅, ✓) \mid V(🤠, ✗) \mid !\mathcal{A} \quad \approx \quad V(🎅, ✗) \mid V(🤠, ✓) \mid !\mathcal{A}$$

Where $\approx$ is a behavioral equivalence: 😈 cannot tell both sides apart.

Trivial Example: $V($👤$,$✉$) := 1 : \mathsf{out}(c,$👤$).\mathsf{out}(c,$✉$)$

E-Voting and Privacy

E-Voting Protocol (simplified)

- Roles as processes: Voter: $V($👤$,$✉$)$ and authorities: $A \in \mathcal{R}$
- Tally as a function `Tally` over frames
- Honest Trace: a fixed, full, honest execution of $\{V($🎅$, ✓)\} \cup \mathcal{R}$

Ballot Privacy (simplified)

$$V(🎅, ✓) \mid V(🤠, ✗) \mid !\mathcal{A} \quad \approx \quad V(🎅, ✗) \mid V(🤠, ✓) \mid !\mathcal{A}$$

Where $\approx$ is a behavioral equivalence: 😈 cannot tell both sides apart.

Trivial Example: $V($👤$,$✉$) := 1 : \mathsf{out}(c,$👤$).\mathsf{out}(c,$✉$)$ attack 😈!

In $V($🎅$, ✓) \mid V($🤠$, ✗)$, 😈 can "block" 🤠 and observes 🎅's ✉: $✓ \neq ✗$

E-Voting and Privacy

E-Voting Protocol (simplified)

- Roles as processes: Voter: $V(\text{🧑}, \text{✉})$ and authorities: $A \in \mathcal{R}$
- Tally as a function `Tally` over frames
- Honest Trace: a fixed, full, honest execution of $\{V(\text{🎅}, ✓)\} \cup \mathcal{R}$

Ballot Privacy (simplified)

$$V(\text{🎅}, ✓) \mid V(\text{🤠}, \times) \mid !\mathcal{A} \quad \approx \quad V(\text{🎅}, \times) \mid V(\text{🤠}, ✓) \mid !\mathcal{A}$$

Where $\approx$ is a behavioral equivalence: 👹 cannot tell both sides apart.

Trivial Example: $V(\text{🧑}, \text{✉}) := 1 : \mathsf{out}(c, \text{🧑}).\ 2 : \mathsf{out}(c, \text{✉})$ secure ☺

$\leadsto$ 👹 has to let both 🤠 and 🎅 reach phase 2 before getting any ✉

Problem

State-of-the-art: ≈ approximated by "diff-equivalence" (when ∞ sessions)

Ballot privacy: $V($🎅$, ✓) \mid V($🤠$, \times) \mid !\mathcal{A} \approx V($🎅$, \times) \mid V($🤠$, ✓) \mid !\mathcal{A}$

Problem

State-of-the-art: $\approx$ approximated by "diff-equivalence" (when ∞ sessions)

Ballot privacy: $V(\,, \mathsf{diff}[\checkmark, \times]) \mid V(\,, \mathsf{diff}[\times, \checkmark]) \mid !\mathcal{A}$

Problem

State-of-the-art: $\approx$ approximated by "diff-equivalence" (when ∞ sessions)

Ballot privacy: $V(\;, \mathsf{diff}[✓, ✗]) \mid V(\;, \mathsf{diff}[✗, ✓]) \mid !\mathcal{A}$

diff-equivalence = "$\approx$ for who knows internal structure of processes"

Implications:

- knows when actions are triggered by the same process/agent

Structural links given to vs. ballot privacy=absence of certain links:

$\leadsto$ systematic false attacks on ballot secrecy

$\leadsto$ *ad hoc* work-arounds with limited applicability *e.g.* swaps of processes

Our hybrid approach: privacy via sufficient conditions

Methodology:

- focus on some class of protocols and some privacy goal
- identify conditions (inspired by generic classes of attacks)
- that are sufficient (soundness),
- fundamentally simpler and easier to check (checkability), and
- met by (secure) protocols (tightness)

Our hybrid approach: privacy via sufficient conditions

Methodology:

- focus on some class of protocols and some privacy goal
- identify conditions (inspired by generic classes of attacks)
- that are sufficient (soundness),
- fundamentally simpler and easier to check (checkability), and
- met by (secure) protocols (tightness)

Goal: More precise & efficient verification techniques + extends the scope.

First developed for untraceability:

L.H., D. Bælde, and S. Delaune. *"A method for unbounded verification of privacy-type properties".* Journal **JCS'19** and conference **S&P'16.**

Leaking Status

Take for instance: $V($👤$,$✉$) =$ new $n.1 :$ out$($👤$).P.$out$($✉$).$out(n)

V= 1:

Out(id)	In(y)	Out(u)

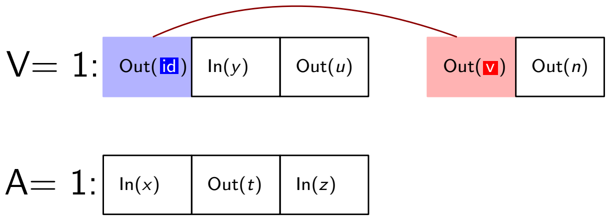
Out(v)	Out(n)

A= 1:

In(x)	Out(t)	In(z)

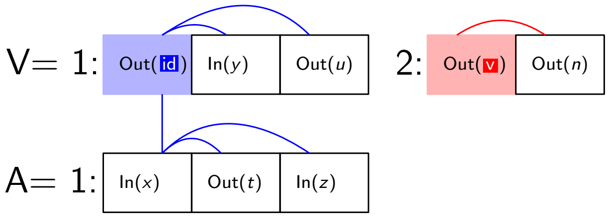
Leaking Status

Take for instance: $V($👤$,$✉$)$ = new $n.1$: out(👤)$.P.$out(✉).out(n)



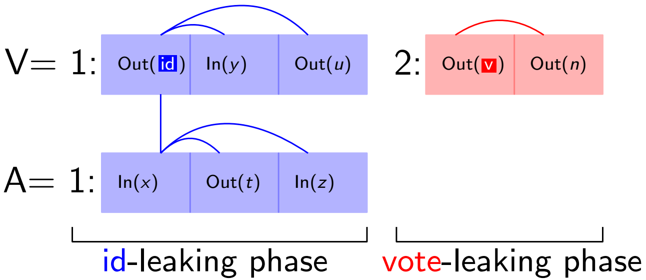
Leaking Status

Take for instance: $V($👤, ✉$)$ = new $n.1$: out(👤)$.P.$ 2 : out(✉).out(n)



Leaking Status

Take for instance: $V($👤$,$✉$)$ = new $n.1:$ out(👤)$.P.\ 2:$ out(✉).out(n)

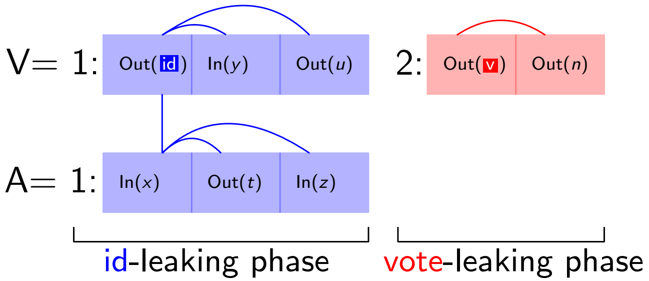


- At most 1 type of leak in a single phase $\rightsquigarrow$ phase leaking status

id-leaking phases unlinkable to vote
$\wedge$ vote-leaking phases unlinkable to id

Leaking Status

Take for instance: $V($👤$,$✉$)$ = new $n.1$: out(👤)$.P.\ 2$: out(✉).out(n)



- At most 1 type of leak in a single phase $\rightsquigarrow$ phase leaking status

  id-leaking phases unlinkable to vote
  $\wedge$ vote-leaking phases unlinkable to id

- Similarly: name has at most 1 type of leak $\rightsquigarrow$ name leaking status

1: Dishonest Condition

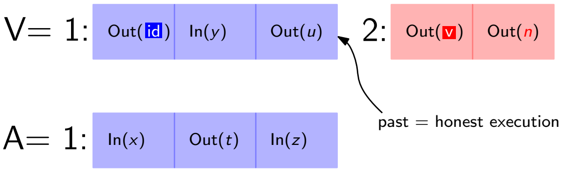
Idea: if a deviation from the honest execution at phase i has some impact at phase $j > i \rightsquigarrow$ 👹 may link phases i and j.

e.g. taint credential at phase 1 and observe it at phase 2

Dishonest Condition (Informal)

For any execution, if a voter process V at phase j is still present at the end, then it followed the honest trace up to $j-1$.

- Prevent a class of attacks
- Allow us to focus on less executions (those that meet the condition)



1: Dishonest Condition

Idea: if a deviation from the honest execution at phase i has some impact at phase $j > i \rightsquigarrow$ 👹 may link phases i and j.

e.g. taint credential at phase 1 and observe it at phase 2

Dishonest Condition (Informal)

For any execution, if a voter process V at phase j is still present at the end, then it followed the honest trace up to $j - 1$.

- Prevent a class of attacks
- Allow us to focus on less executions (those that meet the condition)

$$\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathsf{id}}_A, \mathbf{n}^{\mathsf{v}}_1) = \Big\{ 1{:}\ \boxed{\mathsf{Out}(\boxed{A})}\boxed{\mathsf{In}(y)}\boxed{\mathsf{Out}(u)}\ ,\quad 1{:}\ \boxed{\mathsf{In}(x)}\boxed{\mathsf{Out}(t)}\boxed{\mathsf{In}(z)} \Big\}$$

$$\mathcal{R}^{\mathsf{v}}(\mathbf{n}^{\mathsf{id}}_A, \mathbf{n}^{\mathsf{v}}_1) = \Big\{ 2{:}\ \boxed{\mathsf{Out}(\boxed{1})}\boxed{\mathsf{Out}(n_1)} \Big\}$$

less structural links with “standalone phase-processes” ☺

2: Relation Condition

We would like to check the absence of 👤-✉ relation for all phase-processes.

(less structural links now ☺)

$\mathrm{diff}[\mathbf{n}^{\mathsf{v}}_{\checkmark}, \mathbf{n}^{\mathsf{v}}_{\times}]$ in id-leaking phase-processes

Defined as the diff-equivalence of:

$$\begin{aligned}\mathcal{B} = \quad & \{\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🎅}, \mathrm{diff}[\mathbf{n}^{\mathbf{v}}_{\checkmark}, \mathbf{n}^{\mathbf{v}}_{\times}]),\\ & \mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🤠}, \mathrm{diff}[\mathbf{n}^{\mathbf{v}}_{\times}, \mathbf{n}^{\mathbf{v}}_{\checkmark}])\\ & \} \uplus\, !\mathcal{R}\end{aligned}$$

2: Relation Condition

We would like to check the absence of 👤-✉ relation for all phase-processes.

(less structural links now ☺)

$$\mathrm{diff}[\mathbf{n}^{\mathsf{v}}_{\checkmark}, \mathbf{n}^{\mathsf{v}}_{\times}] \text{ in id-leaking phase-processes}$$
$$\mathrm{diff}[\mathbf{n}^{\mathsf{id}}_{\text{🎅}}, \mathbf{n}^{\mathsf{id}}_{\text{🤠}}] \text{ in vote-leaking phase-processes}$$

Defined as the diff-equivalence of:

$$\begin{aligned}\mathcal{B} = \quad & \{\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{\text{🎅}}, \mathrm{diff}[\mathbf{n}^{\mathbf{v}}_{\checkmark}, \mathbf{n}^{\mathbf{v}}_{\times}]), \quad \mathcal{R}^{\mathsf{v}}(\mathrm{diff}[\mathbf{n}^{\mathbf{id}}_{\text{🎅}}, \mathbf{n}^{\mathbf{id}}_{\text{🤠}}], \mathbf{n}^{\mathbf{v}}_{\checkmark}), \\ & \mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{\text{🤠}}, \mathrm{diff}[\mathbf{n}^{\mathbf{v}}_{\times}, \mathbf{n}^{\mathbf{v}}_{\checkmark}]), \quad \mathcal{R}^{\mathsf{v}}(\mathrm{diff}[\mathbf{n}^{\mathbf{id}}_{\text{🤠}}, \mathbf{n}^{\mathbf{id}}_{\text{🎅}}], \mathbf{n}^{\mathbf{v}}_{\times}) \\ & \} \uplus !\mathcal{R}\end{aligned}$$

2: Relation Condition

We would like to check the absence of 👤-✉ relation for all phase-processes.

(less structural links now ☺)

diff$[\mathbf{n}^{\mathsf{v}}_{✓}, \mathbf{n}^{\mathsf{v}}_{✗}]$ in id-leaking phase-processes

diff$[\mathbf{n}^{\mathsf{id}}_{🎅}, \mathbf{n}^{\mathsf{id}}_{🤠}]$ in vote-leaking phase-processes

Defined as the diff-equivalence of:

$$\mathcal{B} = \quad \{\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🎅}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{✓}, \mathbf{n}^{\mathbf{v}}_{✗}]), \quad \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🎅}, \mathbf{n}^{\mathbf{id}}_{🤠}], \mathbf{n}^{\mathbf{v}}_{✓}),$$

$$\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🤠}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{✗}, \mathbf{n}^{\mathbf{v}}_{✓}]), \quad \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🤠}, \mathbf{n}^{\mathbf{id}}_{🎅}], \mathbf{n}^{\mathbf{v}}_{✗})$$

$$\} \uplus !\mathcal{R}$$

Relation Condition (Informal)

The Honest Relations Condition is satisfied if $\mathcal{B}$ is diff-equivalent.

Our Results

Theorem (soundness)

For any $E = (V(👤, ✉), \mathcal{R}, \text{Tally})$, if the Dishonest, Relation, and Tally conditions hold then E satisfies ballot secrecy.

(Tally condition omitted)

- We provide an algorithm for computing models checking the conditions and heuristics to find leaking status (checkability) (tool is FW)
- We verify some case studies + benchmarks (tightness):

Protocol	Ballot Secrecy	Our verif. time	Previous state of the art
FOO	✓	0.04	0.26
Lee 1	✓	0.04	46
Lee 2	✓	0.05	†
Lee 3	✓	0.01	†
Lee 4	👹	6.64	169.94
JCJ	✓	18.79	×
Belenios	✓	0.02	×

×: false attack †: non-termination (>45h)

Conclusion

Summary

- Three tight, sufficient conditions for ballot privacy
- Expands the class of protocols and threat models that can be verified
- More efficient verification

Future Work

- Extend our result with more precise Tally
- Combine with the new BPRIV privacy definition [S&P'15, Euro S&P'19]
- Provide a tool with ProVerif/Tamarin as back-end
- Reuse methodology for other contexts/privacy properties

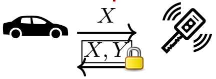
`lucca.hirschi@inria.fr`

Backup Slides

Symbolic Model

Big Picture

Protocol's specification $\longmapsto$ Protocol's model

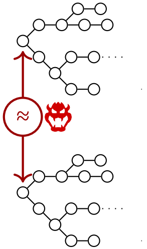


$$P = \mathsf{in}(x).\ \mathsf{new}\ Y.\ \mathsf{out}(\mathsf{enc}((x,Y),k))$$

$$P = \ldots$$

Undecidable

$\approx$ between transition systems



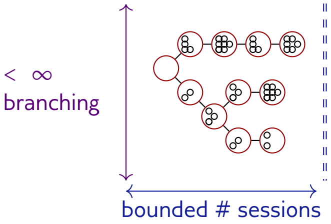
Privacy goal $\longmapsto$ $\approx$ between scenarios

e.g. cannot track

e.g. , $\approx$,

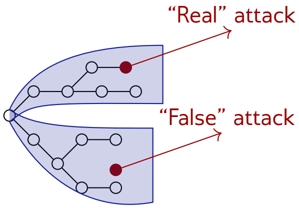
Two Approaches for Verifying ≈ Automatically

Decision for $< \infty$ sessions



- bound the number of sessions
- symbolic semantics
  ⇝ finite description of
- exhaustive exploration of symbolic executions
- Tools: Apte, Akiss, Spec

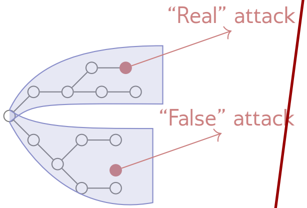
Semi-decision for ∞ sessions



- over-approximations of & semantics
- strong form of ≈ (*i.e.* diff-equivalence)
- Tools: ProVerif, Tamarin, Maude-NPA

Limitation of Semi-decision Procedures

Semi-decision for ∞ sessions



- over-approximations of 👹 & semantics
- strong form of ≈ (*i.e.* diff-equivalence)
- Tools: ProVerif, Tamarin, Maude-NPA

- Serious Precision Issue (privacy)
- ⇝ systematic false attacks for *e.g.* unlinkability, vote-privacy (e-Passport, RFID protocols, 4G, e-voting ...)

Applied π-Calculus

Model of messages: Term algebra

- Function symbols $\qquad$ $\mathsf{enc}(\cdot,\cdot),\ \mathsf{dec}(\cdot,\cdot)$
- Equational theory $=_{\mathsf{E}}$ + computation relation $\downarrow$ $\qquad$ $\mathsf{dec}(\mathsf{enc}(x,y),y) \downarrow x$

Model of protocols: Process calculus

- Process:

P, Q	$:=$	0	null
	$\mid$	$\mathsf{in}(c,x).P$	input
	$\mid$	$\mathsf{out}(c,m).P$	output
	$\mid$	$\texttt{let}\ x = v\ \texttt{then} P\ \texttt{else}\ Q$	conditional
	$\mid$	$P \mid Q$	parallel
	$\mid$	$!\ P$	replication
	$\mid$	$\mathsf{new}\ n.P$	creation of name
	$\mid$	$i : P$	weak phase

- Frame (ϕ): the set of messages revealed to $\qquad$ ('s knowledge)

$$\phi = \{ \underbrace{w_1}_{\text{handle}} \mapsto \underbrace{\mathsf{enc}(m,k)}_{\text{out. message}}, w_2 \mapsto k\}$$

- Configuration: $A = (\mathcal{P}; \phi; j)$

Applied-π - Semantics

- Recipes: terms built using handles

e.g. $R = \mathsf{dec}(w_1, w_2)$ for $\phi = \{w_1 \mapsto \mathsf{enc}(m, k), w_2 \mapsto k\}$

$R\phi =_{\mathsf{E}} m$

"How builds messages from its knowledge"

Applied-π - Semantics

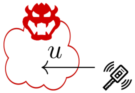
- Recipes: terms built using handles

  e.g. $R = \mathsf{dec}(w_1, w_2)$ for $\phi = \{w_1 \mapsto \mathsf{enc}(m,k), w_2 \mapsto k\}$
  $R\phi =_{\mathsf{E}} m$

  "How builds messages from its knowledge"

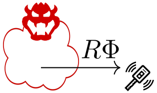
- Protocol's output:

$$(\{i : \mathsf{out}(c,u).P\} \cup \mathcal{P}; \phi; i) \xrightarrow{\mathsf{out}(c,w)} (\{i : P\} \cup \mathcal{P}; \phi \cup \{w \mapsto u\}; i) \quad \text{if } w \text{ fresh}$$



- Protocol's input:

$$(\{i : \mathsf{in}(c,x).P\} \cup \mathcal{P}; \phi; i) \xrightarrow{\mathsf{in}(c,R)} (\{i : P\{x \mapsto R\phi\}\} \cup \mathcal{P}; \phi; i)$$



- + expected rules for conditional (modulo $=_{\mathsf{E}}$) & others

$\rightsquigarrow$ controls all the network

Applied-π - Trace Equivalence

Static Equivalence (intuitively)

$\Phi \sim \Psi$ when

- $\mathrm{dom}(\Phi) = \mathrm{dom}(\Psi)$ and
- for all tests, it holds on $\Phi \iff$ it holds on Ψ (modulo $=_E$)

Trace Equivalence

$A \approx B$: for any $A \xrightarrow{t} A'$ there exists $B \xrightarrow{t'} B'$ such $\Phi(A') \sim \Phi(B')$ and $\mathrm{obs}(t) = \mathrm{obs}(t')$

(and the converse).

Privacy

Unlinkability

$$\mathcal{M} := !\mathsf{new\ Id}.\ !\mathsf{new\ Sess}.(P_{\text{📡}} \mid P_{\text{🚗}}) \approx^{?} !\mathsf{new\ Id}.\ \mathsf{new\ Sess}.(P_{\text{📡}} \mid P_{\text{🚗}})$$

😈 cannot establish meaningful link between two interactions (with same Id)

Anonymity

$$\mathcal{M} \mid !\mathsf{new\ Sess}.(P_{\text{📡}}(\mathsf{Id}_0) \mid P_{\text{🚗}}(\mathsf{Id}_0)) \approx^{?} \mathcal{M}$$

😈 cannot establish meaningful link between an interaction and identity Id_0

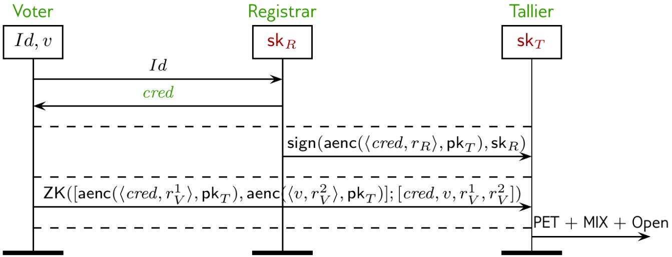
Ballot Secrecy

$$V(\text{🎅}, \checkmark) \mid V(\text{🤠}, \times) \mid !\mathcal{A} \approx^{?} V(\text{🎅}, \times) \mid V(\text{🤠}, \checkmark) \mid !\mathcal{A}$$

😈 cannot establish meaningful link between a voter and his vote

Goal: Analyzing Ballot Secrecy

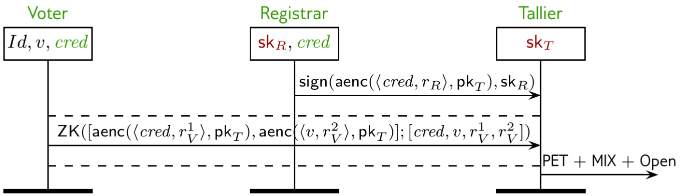
Often, only the core voting protocol is analyzed.



Voter
Registrar
Tallier
Id, v
sk_R
sk_T
Id
$cred$
$\mathsf{sign}(\mathsf{aenc}(\langle cred, r_R\rangle, \mathsf{pk}_T), \mathsf{sk}_R)$
$\mathsf{ZK}([\mathsf{aenc}(\langle cred, r_V^1\rangle, \mathsf{pk}_T), \mathsf{aenc}(\langle v, r_V^2\rangle, \mathsf{pk}_T)]; [cred, v, r_V^1, r_V^2])$
PET + MIX + Open

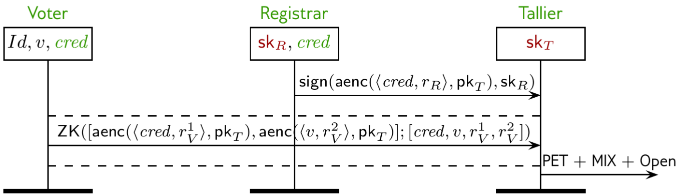
Goal: Analyzing Ballot Secrecy

Often, only the core voting protocol is analyzed.


Voter
$Id, v, cred$
Registrar
$\mathsf{sk}_R, cred$
Tallier
sk_T
$\mathsf{sign}(\mathsf{aenc}(\langle cred, r_R\rangle, \mathsf{pk}_T), \mathsf{sk}_R)$
$\mathsf{ZK}([\mathsf{aenc}(\langle cred, r_V^1\rangle, \mathsf{pk}_T), \mathsf{aenc}(\langle v, r_V^2\rangle, \mathsf{pk}_T)]; [cred, v, r_V^1, r_V^2])$
PET + MIX + Open

Goal: Analyzing Ballot Secrecy

Often, only the core voting protocol is analyzed.


Voter
$Id, v, cred$
Registrar
$\mathsf{sk}_R, cred$
Tallier
sk_T
$\mathsf{sign}(\mathsf{aenc}(\langle cred, r_R\rangle, \mathsf{pk}_T), \mathsf{sk}_R)$
$\mathsf{ZK}([\mathsf{aenc}(\langle cred, r_V^1\rangle, \mathsf{pk}_T), \mathsf{aenc}(\langle v, r_V^2\rangle, \mathsf{pk}_T)]; [cred, v, r_V^1, r_V^2])$
PET + MIX + Open

We would like to take into account important aspects such as:

- registration, credential delivery
- authentication
- voting
- tallying

We would like to:

- compare different threat models (no security if everything is compromised)
- identify minimal honesty assumptions

Verifying Ballot Secrecy

$V($🎅$, \checkmark) \mid V($🤠$, \times) \mid !\mathcal{A} \approx^{?} V($🎅$, \times) \mid V($🤠$, \checkmark) \mid !\mathcal{A}$

Diff-equivalence yields false attacks

Take: $V($👤$,$ ✉$) = 1 : \mathsf{out}(c,$ 👤$).\ 2 : \mathsf{out}(c,$ ✉$)$

With diff-equivalence, 👹 can link all actions from 🎅 (resp. 🤠)

$\rightsquigarrow$ attacker can link 👤 and ✉

State-of-the-Art

Weakening diff-equivalence (improving the tool):

- Swapping approach – Idea:[DRS'08], Proof+ProVerif:[BB'16], Tamarin:[DDKS'17]: allows to change biprocess pairing at sync. barriers

Hybrid approaches:

- type system [CGLM'17]
- small attack property [ACK'16]

State-of-the-Art

Weakening diff-equivalence (improving the tool):

- Swapping approach – Idea:[DRS'08], Proof+ProVerif:[BB'16], Tamarin:[DDKS'17]: allows to change biprocess pairing at sync. barriers
  Limitations:
  - no swap/phase under replication $\rightsquigarrow$
    - no honest authority present in $\neq$ phases
    - no threat model with no dishonest voters
  - introduction of new internal communication $\rightsquigarrow$
    - false attacks in presence of fresh data going through phases $\big(1 : \mathsf{new}\ n.2 : \mathsf{out}(c,(v,n))\big)$

Hybrid approaches:

- type system [CGLM'17]
- small attack property [ACK'16]

State-of-the-Art

Weakening diff-equivalence (improving the tool):

- Swapping approach – Idea:[DRS'08], Proof+ProVerif:[BB'16], Tamarin:[DDKS'17]: allows to change biprocess pairing at sync. barriers
  Limitations:
  - no swap/phase under replication $\leadsto$
    - no honest authority present in $\neq$ phases
    - no threat model with no dishonest voters
  - introduction of new internal communication $\leadsto$
    - false attacks in presence of fresh data going through phases $\big(1 : \text{new } n.2 : \text{out}(c, (v, n))\big)$

Hybrid approaches:

- type system [CGLM'17] but pairing is as rigid as diff-equivalence, standard primitives only
- small attack property [ACK'16] but only 1 phase, performance issues

In practice, interesting threat models and modeling of *e.g.* Lee, JCJ, Belenios are out of the scope

Our contribution – Big Picture

We develop a privacy via sufficient conditions approach for ballot secrecy and a large class of e-voting protocols (soundness, checkability, tightness).

We apply our technique on FOO, Lee, JCJ and Belenios (with registration):

- false attacks using previous techniques (*e.g.* JCJ, Belenios)
- much better performance (*e.g.* $*10^2$, termination for LEE)

Our contribution – Big Picture

We develop a privacy via sufficient conditions approach for ballot secrecy and a large class of e-voting protocols (soundness, checkability, tightness).

We apply our technique on FOO, Lee, JCJ and Belenios (with registration):

- false attacks using previous techniques (*e.g.* JCJ, Belenios)
- much better performance (*e.g.* $*10^2$, termination for LEE)

Main Limitation:

- Tallier is too unrealistic: no revote policy, homomorphic tallying

Class of e-voting protocols

(Honest) Roles:

- Voter: $V($👤$,$✉$) = i : \text{new } \vec{n}.V'$ such that V' has no $!, |$ or new
- $A \in \mathcal{R}$ authority session, same format +(?) voters
- Some role $A_c \in \mathcal{R}$ is the bulletin box and $A_b \ni \text{out}(c_b, t)$ "stores in BB"

Tally:

- Made of a public term Ψ_b (correct form?) and private term Extract (check validity and extract vote)
- "Tally" $= !i_f : \text{in}(c, x).\texttt{let } (\_, v) = (\Psi_b[x], \text{Extract}[x]) \text{ in out}(c, v)$

Honest Trace: (symbolic) trace th s.t. $(\mathcal{R} \cup \{V($🎅$, \checkmark)\}; \phi_0; 1) \xrightarrow{\text{th}} (\varnothing; \phi; i_f)$

Class of e-voting protocols

(Honest) Roles:

- Voter: $V($👤$,$✉$) = i : \text{new } \vec{n}.V'$ such that V' has no $!, |$ or new
- $A \in \mathcal{R}$ authority session, same format +(?) voters
- Some role $A_c \in \mathcal{R}$ is the bulletin box and $A_b \ni \text{out}(c_b, t)$ "stores in BB"

Tally:

- Made of a public term Ψ_b (correct form?) and private term Extract (check validity and extract vote)
- "Tally" $= !i_f : \text{in}(c, x).\texttt{let } (\_, v) = (\Psi_b[x], \text{Extract}[x]) \text{ in out}(c, v)$

Honest Trace: (symbolic) trace th s.t. $(\mathcal{R} \cup \{V($🎅$,\checkmark)\}; \phi_0; 1) \xrightarrow{\text{th}} (\emptyset; \phi; i_f)$

E-Voting Protocol: $(\mathcal{V}; \phi_0; V($👤$,$✉$), \mathcal{R}, (\Psi_b, \text{Extract}))$

Ballot Secrecy

$$V(🎅, ✓) \mid V(🤠, ✗) \mid !\mathcal{R} \mid \mathsf{Tally} \;\approx^{?}\; V(🎅, ✗) \mid V(🤠, ✓) \mid !\mathcal{R} \mid \mathsf{Tally}$$

(Weak) phases are not enough

Take: $V(👤, ✉) = 1 : \mathsf{out}(c, 👤).\ 2 : \mathsf{out}(c, ✉)$

In $V(🎅, ✓) \mid V(🤠, ✗)$, 😈 can block 🤠 and observes 🎅's ✉: $✓ \neq ✗$

But strong phases suffer from theoretical limitations w.r.t. replications.

Idea:

- Executions with strong phases = executions with weak phases that wait for all processes at each phase jump

Ballot Secrecy

$$V(🎅,\checkmark) \mid V(🤠,\times) \mid !\mathcal{R} \mid \text{Tally} \approx^{?}_{\text{fair}} V(🎅,\times) \mid V(🤠,\checkmark) \mid !\mathcal{R} \mid \text{Tally}$$

(Weak) phases are not enough

Take: $V(👤,✉) = 1 : \text{out}(c,👤).\ 2 : \text{out}(c,✉)$

In $V(🎅,\checkmark) \mid V(🤠,\times)$, 😈 can block 🤠 and observes 🎅's ✉: $\checkmark \neq \times$

But strong phases suffer from theoretical limitations w.r.t. replications.

Idea:

- Executions with strong phases = executions with weak phases that wait for all processes at each phase jump

$$\cap$$

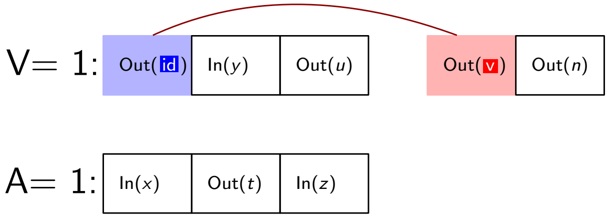
- Fair executions = executions with weak phases that wait for 🎅 and 🤠

Ballot Secrecy: Use weak phases+$\approx_{\text{fair}}$ instead of strong phases+$\approx$

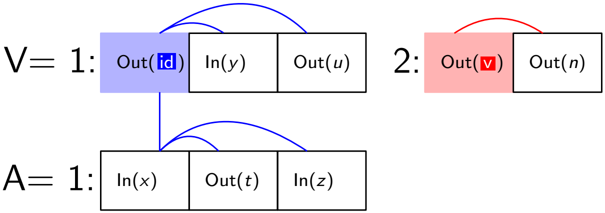
Leaking Status

V= 1:	Out(id)	In(y)	Out(u)	Out(v)	Out(n)
A= 1:	In(x)	Out(t)	In(z)		

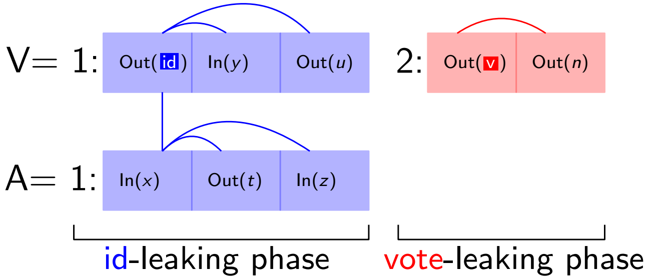
Leaking Status



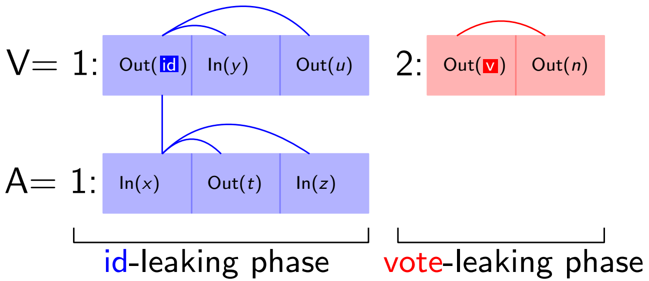
Leaking Status



Leaking Status



Leaking Status

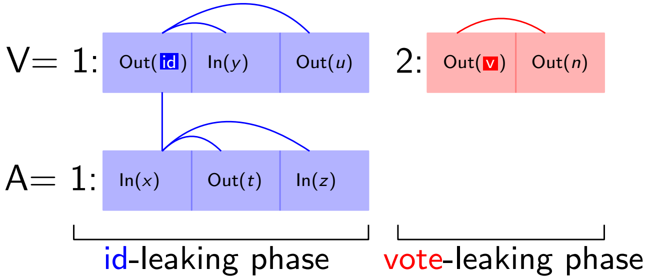


- at most 1 type of leak in a single phase $\rightsquigarrow$ phase leaking status

id-leaking phases unlinkable to v
$\wedge$ vote-leaking phases unlinkable to id

$\approx$(diff$[v_1, v_2]$ in id-leaking phases
diff$[id_1, id_2]$ in vote-leaking phases)

Leaking Status



- at most 1 type of leak in a single phase $\rightsquigarrow$ phase leaking status

id-leaking phases unlinkable to v
$\wedge$ vote-leaking phases unlinkable to id

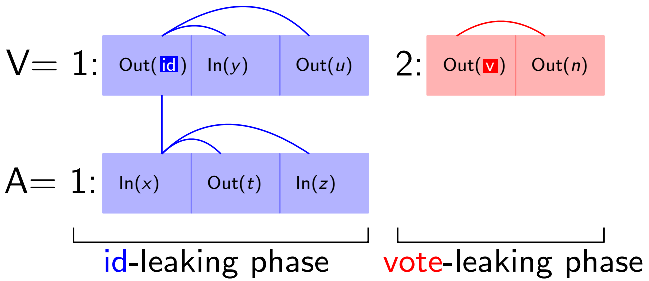
$$\approx \left(\begin{array}{l} \text{diff}[v_1, v_2] \text{ in id-leaking phases} \\ \text{diff}[id_1, id_2] \text{ in vote-leaking phases} \end{array} \right)$$

- name has at most 1 type of link $\rightsquigarrow$ name leaking status

id-leaking phases/names unlinkable to v
$\wedge$ vote-leaking phases/names unlinkable to id

$$\approx \left(\begin{array}{l} \text{diff}[\mathbf{n}_1^v, \mathbf{n}_2^v] \text{ in id-leaking phases} \\ \text{diff}[\mathbf{n}_1^{id}, \mathbf{n}_2^{id}] \text{ in vote-leaking phases} \end{array} \right)$$

Leaking Status



- at most 1 type of leak in a single phase $\rightsquigarrow$ phase leaking status

id-leaking phases unlinkable to v
$\wedge$ vote-leaking phases unlinkable to id
$\approx$(diff$[v_1, v_2]$ in id-leaking phases
diff$[id_1, id_2]$ in vote-leaking phases)

- name has at most 1 type of link $\rightsquigarrow$ name leaking status

id-leaking phases/names unlinkable to v
$\wedge$ vote-leaking phases/names unlinkable to id
$\approx$(diff$[\mathbf{n}_1^{v}, \mathbf{n}_2^{v}]$ in id-leaking phases
diff$[\mathbf{n}_1^{id}, \mathbf{n}_2^{id}]$ in vote-leaking phases)

But diff-equivalence is still problematic

Phase-Process and Dishonest Condition

Idea: if a deviation from the honest execution in phase i has some impact in phase $j > i \rightsquigarrow$ 😈 may link phases i and j.

e.g. “weaken”/taint credential in phase 1 and observe it in phase 2

Dishonest Condition (Informal)

For any fair execution $(\mathcal{S};\phi_0;1) \xrightarrow{\texttt{t.phase}(j)} (\mathcal{P};\phi;j)$, if a process at phase j annotated [👤, ✉] for 👤 $\in$ {🎅, 🤠} and ✉ $\in \mathcal{V}$ is present in $\mathcal{P}$ then it followed th up to phase j.

Phase-Process and Dishonest Condition

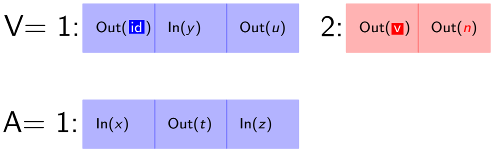
Idea: if a deviation from the honest execution in phase i has some impact in phase $j > i \leadsto$ 😈 may link phases i and j.

e.g. "weaken"/taint credential in phase 1 and observe it in phase 2

Dishonest Condition (Informal)

For any fair execution $(\mathcal{S}; \phi_0; 1) \xrightarrow{\texttt{t.phase}(j)} (\mathcal{P}; \phi; j)$, if a process at phase j annotated [👤, ✉] for 👤 $\in$ {🎅, 🤠} and ✉ $\in \mathcal{V}$ is present in $\mathcal{P}$ then it followed th up to phase j.

- Prevent a class of attacks
- Allow us to focus on less executions (those that meet the condition)



Phase-Process and Dishonest Condition

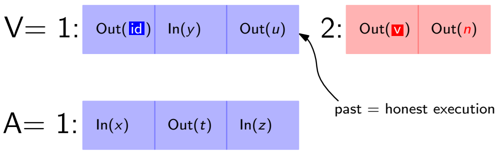
Idea: if a deviation from the honest execution in phase i has some impact in phase $j > i \rightsquigarrow$ 😈 may link phases i and j.

e.g. “weaken”/taint credential in phase 1 and observe it in phase 2

Dishonest Condition (Informal)

For any fair execution $(\mathcal{S}; \phi_0; 1) \xrightarrow{\texttt{t.phase}(j)} (\mathcal{P}; \phi; j)$, if a process at phase j annotated [👤, ✉] for 👤 ∈ {🎅, 🤠} and ✉ ∈ $\mathcal{V}$ is present in $\mathcal{P}$ then it followed th up to phase j.

- Prevent a class of attacks
- Allow us to focus on less executions (those that meet the condition)



Phase-Process and Dishonest Condition

Idea: if a deviation from the honest execution in phase i has some impact in phase $j > i \rightsquigarrow$ 😈 may link phases i and j.

e.g. "weaken"/taint credential in phase 1 and observe it in phase 2

Dishonest Condition (Informal)

For any fair execution $(\mathcal{S};\phi_0;1) \xrightarrow{\texttt{t.phase}(j)} (\mathcal{P};\phi;j)$, if a process at phase j annotated [👤, ✉] for 👤 ∈ {🎅, 🤠} and ✉ ∈ $\mathcal{V}$ is present in $\mathcal{P}$ then it followed th up to phase j.

- Prevent a class of attacks
- Allow us to focus on less executions (those that meet the condition)

$$\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathsf{id}}_A, \mathbf{n}^{\mathsf{v}}_1) = \{\, 1{:}\ \boxed{\mathsf{Out}(\boxed{A})}\ \boxed{\mathsf{In}(y)}\ \boxed{\mathsf{Out}(u)}\ ,\quad 1{:}\ \boxed{\mathsf{In}(x)}\ \boxed{\mathsf{Out}(t)}\ \boxed{\mathsf{In}(z)}\ \}$$

$$\mathcal{R}^{\mathsf{v}}(\mathbf{n}^{\mathsf{id}}_A, \mathbf{n}^{\mathsf{v}}_1) = \{2{:}\ \boxed{\mathsf{Out}(\boxed{1})}\ \boxed{\mathsf{Out}(n_1)}\ \}$$

Relation Condition

We would like to check the absence of 👤-✉ relation for all phase-processes.

(less structural links now ☺)

diff$[\mathbf{n}^{\mathsf{v}}_{✓}, \mathbf{n}^{\mathsf{v}}_{×}]$ in id-leaking process-phases

diff$[\mathbf{n}^{\mathsf{id}}_{🎅}, \mathbf{n}^{\mathsf{id}}_{🤠}]$ in vote-leaking process-phases

Formally defined through a bi-process:

$$\begin{aligned}\mathcal{B} = \quad & (\{\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🎅}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{✓}, \mathbf{n}^{\mathbf{v}}_{×}]), \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🎅}, \mathbf{n}^{\mathbf{id}}_{🤠}], \mathbf{n}^{\mathbf{v}}_{✓}), \\ & \quad \mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🤠}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{×}, \mathbf{n}^{\mathbf{v}}_{✓}]), \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🤠}, \mathbf{n}^{\mathbf{id}}_{🎅}], \mathbf{n}^{\mathbf{v}}_{×})\} \\ & \quad \uplus\ !\mathcal{R}; \phi_0; 1)\end{aligned}$$

Relation Condition

We would like to check the absence of 👤-✉ relation for all phase-processes.

(less structural links now ☺)

diff$[\mathbf{n}^{\mathsf{v}}_{✓}, \mathbf{n}^{\mathsf{v}}_{✗}]$ in id-leaking process-phases

diff$[\mathbf{n}^{\mathsf{id}}_{🎅}, \mathbf{n}^{\mathsf{id}}_{🤠}]$ in vote-leaking process-phases

Formally defined through a bi-process:

$$\begin{aligned}\mathcal{B} = \quad & (\{\mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🎅}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{✓}, \mathbf{n}^{\mathbf{v}}_{✗}]), \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🎅}, \mathbf{n}^{\mathbf{id}}_{🤠}], \mathbf{n}^{\mathbf{v}}_{✓}), \\ & \mathcal{R}^{\mathsf{id}}(\mathbf{n}^{\mathbf{id}}_{🤠}, \mathsf{diff}[\mathbf{n}^{\mathbf{v}}_{✗}, \mathbf{n}^{\mathbf{v}}_{✓}]), \mathcal{R}^{\mathsf{v}}(\mathsf{diff}[\mathbf{n}^{\mathbf{id}}_{🤠}, \mathbf{n}^{\mathbf{id}}_{🎅}], \mathbf{n}^{\mathbf{v}}_{✗})\} \\ & \uplus\ !\mathcal{R}; \phi_0; 1)\end{aligned}$$

Relation Condition (Informal)

The Honest Relations Condition is satisfied if $\mathcal{B}$ is diff-equivalent and th is phase-oblivious.

th is phase-oblivious when it dœs not connect a handle and a recipe of different leaking status

Tally Condition

Goal: prevents ballot secrecy attacks that exploit the tally's outcome.

Ballots are either:

1. (honest): stems from an honest execution of or
2. (dishonest): dœs not depend on data that can be linked to an identity

$\rightsquigarrow$ the vote Tally would extract is insensible to the swap $\leftrightarrow$

Tally Condition

Goal: prevents ballot secrecy attacks that exploit the tally's outcome.

Ballots are either:

1. (honest): stems from an honest execution of 🎅 or 🤠
2. (dishonest): dœs not depend on data that can be linked to an identity

   $\rightsquigarrow$ the vote Tally would extract is insensible to the swap 🎅 $\leftrightarrow$ 🤠

Tally Condition (Informal)

$\forall$ fair execution $\mathcal{B} \xrightarrow{\text{t}} (\mathcal{P}', (\phi_l, \phi_r))$, for any ballot $w\phi_l$ in the BB, either:

1. there exists a voter $V($👤, ✉$)$, 👤 $\in \{$🎅, 🧑$\}$ who had an honest interaction and who has cast w
2. or there exists some $v \in \mathcal{V} \cup \{\bot\}$ such that $\mathrm{Extract}(w\phi_l) \downarrow v$ and $\mathrm{Extract}(w\phi_r) \downarrow v$.

Tally Condition

Goal: prevents ballot secrecy attacks that exploit the tally's outcome.
Ballots are either:

1. (honest): stems from an honest execution of or
2. (dishonest): dœs not depend on data that can be linked to an identity
   $\rightsquigarrow$ the vote Tally would extract is insensible to the swap $\leftrightarrow$

Tally Condition (Informal)

$\forall$ fair execution $\mathcal{B} \xrightarrow{t} (\mathcal{P}', (\phi_l, \phi_r))$, for any ballot $w\phi_l$ in the BB, either:

1. there exists a voter $V($, ✉$)$, $\in \{$, $\}$ who had an honest interaction and who has cast w
2. or there exists some $v \in \mathcal{V} \cup \{\perp\}$ such that $\mathrm{Extract}(w\phi_l) \downarrow v$ and $\mathrm{Extract}(w\phi_r) \downarrow v$.

2. Ballot can depend on data from vote-leaking phases but not from id-leaking phases
$\rightsquigarrow$ bias leaking information on a ballot unlinkable to or is ok
$\rightsquigarrow$ refines ballot independence

Our Results

Theorem (soundness)

For any $E = (\mathcal{V}; \phi_0; V($👤, ✉$), \mathcal{R}, (\Psi_b, \text{Extract}))$, if the Dishonest, Relation and Tally conditions hold then E satisfies ballot secrecy.

Our Results

Theorem (soundness)

For any $E = (\mathcal{V}; \phi_0; V(👤, ✉), \mathcal{R}, (\Psi_b, \text{Extract}))$, if the Dishonest, Relation and Tally conditions hold then E satisfies ballot secrecy.

- We provide an algorithm for computing models checking the conditions and heuristics to find leaking status (checkability) (tool is FW)
- We apply our techniques to several case studies and compare ourselves with the swapping technique (tightness):

Protocol	Ballot Secrecy	Our verif. time	Swapping technique verif. time	
FOO	✓	0.04	0.26	
Lee 1	✓	0.04	46	
Lee 2	✓	0.05	†	(collapsed-phases: 45.33)
Lee 3	✓	0.01	†	(collapsed-phases: 269.06)
Lee 4	✗	6.64	169.94	
JCJ	✓	18.79	✗	
Belenios	✓	0.02	✗	

Conclusion

Reusing core ideas

- Adapt for the case of receipt-freeness and cœrcion-resistance
- Reuse methodology for other contexts/privacy properties
- Infer generic framework (*e.g.* separation btw. data and active deviation issues)
- Extract guidelines for privacy from our conditions (?)

Future Work

- Extend our result with more precise Tally:
- Combine with the new BPRIV privacy definition [S&P'15, Euro S&P'19]
- Provide a tool with ProVerif/Tamarin as back-end
- Reuse methodology for other contexts/privacy properties