Static analysis
for
Safety and Security

Hanne Riis Nielson
LBT – Language Based Technology
Informatics and Mathematical Modelling

Artist2 Motives – Trento, February 2007

Validation of
Cryptographic Protocols
using
Static Analysis

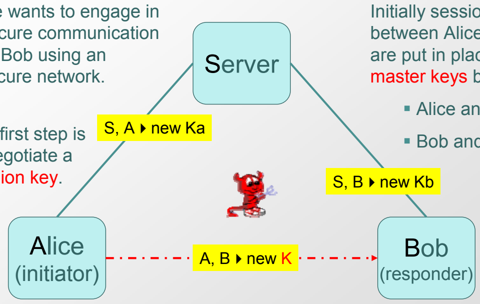
What is the problem?

Alice wants to engage in a secure communication with Bob using an insecure network.

The first step is to negotiate a session key.

Initially session keys between Alice and Bob are put in place using master keys between:

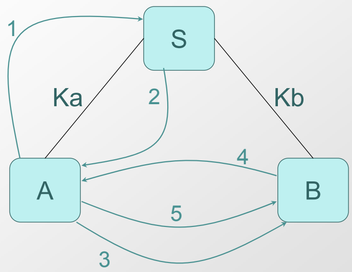
- Alice and Server
- Bob and Server



Given S, A ‣ new Ka and S, B ‣ new Kb
the goal is to achieve A, B ‣ new K

Authenticity
Confidentiality

Needham-Schroeder Symmetric Key Protocol



Protocol narration

1. A → S: A, B, Na
2. S → A: E[Ka](Na,B,K,E[Kb](K,A))
3. A → B: E[Kb](K,A)
4. B → A: E[K](Nb)
5. A → B: E[K](Nb-1)

Does the protocol live up to our expectations?

The Denning-Sacco attack

1. A → S: A, B, Na
2. S → A: E[Ka](Na,B,K,E[Kb](K,A))
3. A → B: E[Kb](K,A)
4. B → A: E[K](Nb)
5. A → B: E[K](Nb-1)

Na: A knows that message 2 is a reply to message 1

Nb: B knows that message 5 is a reply to message 4

The attacker M discovers an old key K' (and the message E[Kb](K',A))

3. M(A) → B: E[Kb](K',A)
4. B → M(A): E[K'](Nb)
5. M(A) → B: E[K'](Nb-1)

B believes he is talking to A!

A is convinced that K is fresh and known to no others than B (and S).

Denning-Sacco's replay attack shows that B does not have a similar guarantee.

Getting it right …

This is 3-5 line programs that people still manage to get wrong!

- Needham-Schroeder protocols [1978]
  - Replay attack — after 3 years [1981]
- Needham-Schroeder public key protocol [1978]
  - Man-in-the-middle attack — after 17 years [1995]
- Denning-Sacco public key protocol [1981]
  - Masquarade attack — after 13 years [1994]
- …

Why is it so difficult?

- we try to program a computer system that is under the control of an intelligent and malicious agent
- the properties we want to ensure are extremely subtle

The problems

- Protocol must be unambiguous; each step must be well-defined and there must be no chance of misunderstanding.
- The protocol must be complete; there must be a specified action for every possible situation.

Specify the protocol using a programming language with a well-defined semantics

- The assumptions under which the protocol operates must be clear.
- It must be clear what security goals the protocol is assumed to provide.

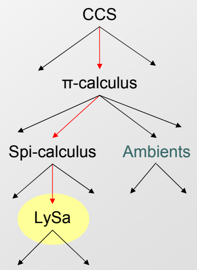
Give a formal specification

- It must be ensured that the protocol really fulfils the security goals under the given assumptions.

Formal validation using static analysis

Process calculi

- Support massive parallelism.
- Incorporate communication.
- Can be extended to handle cryptographic primitives.
- Can be extended to handle mobility and locations.
- Have a formal semantics.
- Are subject to automatic analysis.



Tiny but powerful languages for modelling communicating systems.

LySa syntax

Expressions:

$E ::= n$
$\quad | \; x$
$\quad | \; \{E_1,\ldots,E_k\}_{E_0}$

symmetric encryption and decryption with pattern matching

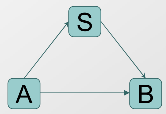
Processes:

$P ::= 0$
$\quad | \; P1 \mid P2$
$\quad | \; !\,P$
$\quad | \; (\nu\, n)\, P$
$\quad | \; \langle E_1, \ldots, E_k \rangle .\, P$
$\quad | \; (E_1,\ldots,E_j; x_{j+1}, \ldots, x_k).\, P$
$\quad | \; \text{decrypt } E \text{ as } \{E_1,\ldots,E_j; x_{j+1},\ldots,x_k\}_{E_0} \text{ in } P$

Pattern match: the values in the first j positions must match

Encoding WMF in LySa

Wide Mouthed Frog:



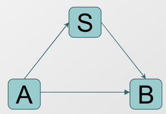
(1) A→S: A, E[Ka](B, K)
(2) S→B: E[Kb](A, K)
(3) A→B: E[K](m)

In LySa:

process for Alice	process for Server	process for Bob

Encoding WMF in LySa

Wide Mouthed Frog:



(1) A→S: A, E[Ka](B, K)
(2) S→B: E[Kb](A, K)
(3) A→B: E[K](m)

Alice

0. (νK)
1. $\langle A, S, A, \{B, K\}_{K^A}\rangle.$

...

Server

1′. $(A, S, A; x).$
1″. decrypt x as $\{B; x^K\}_{K^A}$ in

...

Bob

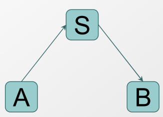
...

sender
receiver
message

Encoding WMF in LySa

Wide Mouthed Frog:

(1) A→S: A, E[Ka](B, K)
(2) S→B: E[Kb](A, K)
(3) A→B: E[K](m)



0. (νK)
1. $\langle A, S, A, \{B, K\}_{K^A}\rangle.$
3. $(\nu m)\ \langle A, B, \{m\}_K\rangle.0$

2′. $\mid (S, B; y).$
2″. decrypt y as $\{A; y^K\}_{K^B}$ in
3′. $(A, B; z).$
3″. decrypt z as $\{; z^m\}_{y^K}$ in 0

1′. $\mid (A, S, A; x).$
1″. decrypt x as $\{B; x^K\}_{K^A}$ in
2. $\langle S, B, \{A, x^K\}_{K^B}\rangle.0$

Annotations for security properties

- Confidentiality (or secrecy)
  - A protocol preserves confidentiality of a message if there does not exists an execution of the protocol in which the attacker learns the message.
- Authentication (of origin)
  - A protocol maintains authentication of origin if each principal can be sure that a message assumed to come from a given principal indeed does come from that principal and furthermore that the message is intended for him.

Authentication in LySa

Focus on encryptions (rather than communication)

- when they are created, specify where they are intended to be decrypted

Crypto-points

$$\{E_1, \cdots, E_k\}^{\ell}_{E_0} [\text{dest } \mathcal{L}]$$

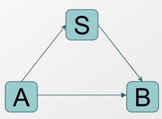
- when they are decrypted, specify where they are expected to have been encrypted

$$\text{decrypt } E \text{ as } \{E'_1, \cdots, E'_j; x_{j+1}, \cdots, x_k\}^{\ell}_{E'_0} [\text{orig } \mathcal{L}] \text{ in } P$$

Encoding WMF in LySa

Wide Mouthed Frog:

(1) A→S: A, E[Ka](B, K)
(2) S→B: E[Kb](A, K)
(3) A→B: E[K](m)



0. (νK)
1. $\langle A, S, A, \{B, K\}^{A}_{K^A} [\text{dest } S] \rangle.$
3. $(\nu\, m) \langle A, B, \{m\}^{A}_{K} [\text{dest } B] \rangle$

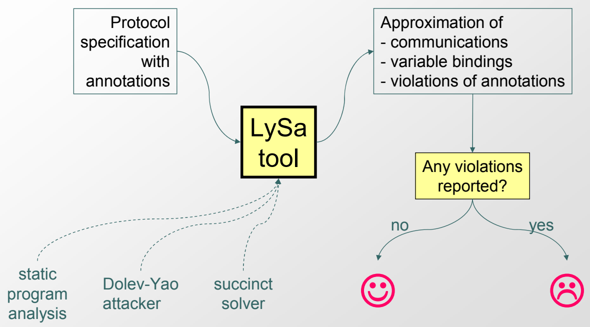
2′. $\mid (S, B; y).$
2″. decrypt y as $\{A; y^K\}^{S}_{K^B}$ [orig S] in
3′. $(A, B; z).$
3″. decrypt z as $\{; z\}^{B}_{y^K}$ [orig A] in 0

1′. $\mid (A, S, A; x).$
1″. decrypt x as $\{B; x^K\}^{S}_{K^A}$ [orig A] in
2. $\langle S, B, \{A, x^K\}^{S}_{K^B} [\text{dest } B] \rangle$

Semantics and analysis

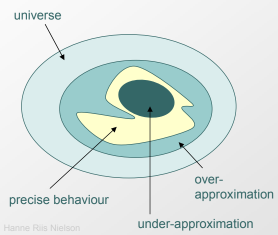
- Standard semantics
  - Does *not* check the annotations.
    This is the semantics we are really interested in!
- Reference Monitor semantics
  - Extension of the standard semantics: it *checks* the annotations and *stops* the execution if they are violated.
- Static program analysis
  - Approximates the reference monitor semantics.
    If no violations of the annotations are reported then the correctness of the analysis guaranteed that the reference monitor never kicks in (and hence that we can dispense with it).

Validating the protocol



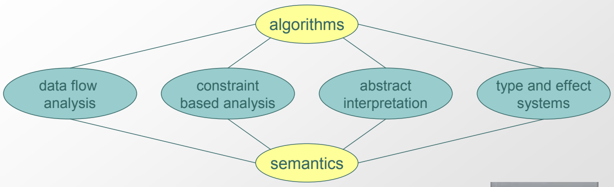
Static program analysis

- The aim is to efficiently compute safe approximations to the behaviour of programs/systems/models without actually running them



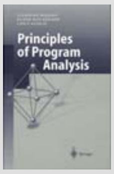
- In general, it is impossible to compute the precise answer
- So we make a choice between over-approximation and under-approximation – never a mix!
- It is an art to make the trade-off between precision and efficiency

Static program analysis



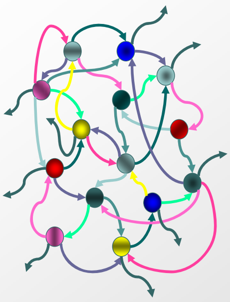
Crucial properties:

- Semantic correctness
  — or we cannot believe in the results
- Efficient implementations
  — or we cannot afford the results



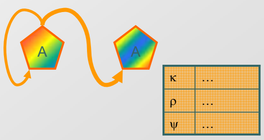
The idea

Semantics | Analysis



Abstract configuration

Abstract execution step:



The methodology

Specification of what it means for an analysis result to be acceptable for a given process

→ Implementation of the analysis

Best analysis results
Smart implementation tricks
Efficient algorithms and data structures
…

↓ Semantic correctness of the analysis

How can we interpret the analysis results?

Specification $(\rho, \kappa) \models_{\mathrm{RM}} P : \psi$

- What does it mean for the analysis result κ, ρ and ψ to be acceptable for the process P?
  - κ must capture *all* the communications that P might perform
  - ρ must, for each variable x, capture *all* the potential values that x might have during the execution of P
  - ψ must capture *all* the potential origin / destination violations that could happen during the execution of P

Analysis judgements

- for terms: $\rho \models E : \vartheta$
  the term E may evaluate to one of the values of the set ϑ in the context given by ρ
- for processes: $(\rho, \kappa) \models_{\mathrm{RM}} P : \psi$
  the process P may give rise to the origin / destination violation ψ in the context given by ρ and κ

  $(\ell, \mathcal{L}) \in \psi$ something encrypted at ℓ may unintentionally be decrypted at $\mathcal{L}$

Analysis of terms $\rho \models E : \vartheta$

- Idea: over-estimate the set of values that a term might have

$$\frac{n \in \vartheta}{\rho \models n : \vartheta} \qquad \frac{\rho(\lfloor x \rfloor) \subseteq \vartheta}{\rho \models x : \vartheta}$$

$$\frac{\begin{array}{l}\wedge_{i=0}^{k}\, \rho \models E_i : \vartheta_i \;\wedge \\ \forall V_0, V_1, \cdots, V_k : \wedge_{i=0}^{k} V_i \in \vartheta_i \\ \quad \Rightarrow \{V_1, \cdots, V_k\}_{V_0}^{\ell}[\mathrm{dest}\ \mathcal{L}] \in \vartheta\end{array}}{\rho \models \{E_1, \cdots, E_k\}_{E_0}^{\ell}[\mathrm{dest}\ \mathcal{L}] : \vartheta}$$

$\rho : \mathcal{X} \rightarrow \wp(\mathcal{V})$

maps variables to sets of values

Analysis of processes

- The idea: imitate what semantics is doing!

$$\frac{[\![E_1]\!] = [\![E_1']\!]}{\langle E_1, E_2\rangle.P \mid (E_1'; x_2).Q \to P \mid Q[E_2/x_2]}$$

- Evaluate the terms and compare their values while ignoring the annotations
- If they agree then communicate and bind the new variables

In the analysis input and output are considered separately

$(\rho, \kappa) \models_{RM} P : \psi$

Analysis of communication

output:

$$\frac{\begin{array}{c}\rho \models E_1 : \vartheta_1 \ \wedge \rho \models E_2 : \vartheta_2 \wedge \\ \forall V_1, V_2 : V_1 \in \vartheta_1 \ \wedge \ V_2 \in \vartheta_2 \ \Rightarrow \langle V_1, V_2\rangle \in \kappa \ \wedge \\ (\rho, \kappa) \models_{RM} P : \psi\end{array}}{(\rho, \kappa) \models_{RM} \langle E_1, E_2\rangle.P : \psi}$$

$V \mathrel{E} \vartheta$ iff $\exists V' \in \vartheta : [\![V]\!] = [\![V']\!]$

input:

$$\frac{\begin{array}{c}\rho \models E_1 : \vartheta_1 \ \wedge \\ \forall \langle V_1, V_2\rangle \in \kappa : V_1 \mathrel{E} \vartheta_1 \Rightarrow V_2 \in \rho(x_2) \wedge \\ (\rho, \kappa) \models_{RM} P : \psi\end{array}}{(\rho, \kappa) \models_{RM} (E_1; x_2).P : \psi}$$

$\kappa \subseteq \wp(\mathcal{V}^*)$ includes all the message sequences that *might* flow on the network

Analysis of processes

- The idea: imitate what semantics is doing!

$$\frac{[\![E_0]\!] = [\![E_0']\!] \quad \wedge \quad [\![E_1]\!] = [\![E_1']\!] \quad \wedge \quad \mathrm{RM}(\ell, \mathcal{L}', \ell', \mathcal{L})}{\begin{array}{c}\text{decrypt } \{E_1, E_2\}^{\ell}_{E_0} [\text{dest } \mathcal{L}] \text{ as } \{E_1'; x_2\}^{\ell'}_{E_0'} [\text{orig } \mathcal{L}'] \text{ in } P \\ \to_{\mathrm{RM}} P[E_2/x_2]\end{array}}$$

- Evaluate the terms and compare their values while ignoring their annotations
- Consult the reference monitor and if the annotations are satisfied then decrypt and bind the new variables

$\mathrm{RM}(\ell, \mathcal{L}', \ell', \mathcal{L}) = \ell \in \mathcal{L}' \ \wedge \ell' \in \mathcal{L}$

The semantics models perfect cryptography:
D[K](E[K](P)) = P

$(\rho, \kappa) \models_{\mathsf{RM}} P : \psi$

Analysis of decryption

$$\frac{\begin{array}{l} \rho \models E : \vartheta \;\wedge \\ \rho \models E_0 : \vartheta_0 \;\wedge\; \rho \models E_1 : \vartheta_1 \;\wedge \\ \forall\, \{V_1, V_2\}^{\ell}_{V_0}[\mathsf{dest}\ \mathcal{L}] \in \vartheta : V_0 \mathrel{\mathsf{E}} \vartheta_0 \;\wedge\; V_1 \mathrel{\mathsf{E}} \vartheta_1 \\ \quad \Rightarrow V_2 \in \rho(x_2) \;\wedge \\ \quad (\neg \mathsf{RM}(\ell, \mathcal{L}', \ell', \mathcal{L}) \Rightarrow (\ell, \ell') \in \psi) \;\wedge \\ \quad (\rho, \kappa) \models_{\mathsf{RM}} P : \psi \end{array}}{(\rho, \kappa) \models_{\mathsf{RM}} \mathsf{decrypt}\ E\ \mathsf{as}\ \{E_1 ; x_2\}^{\ell'}_{E_0}\ [\mathsf{orig}\ \mathcal{L}']\ \mathsf{in}\ P : \psi}$$

The analysis models perfect cryptography:
D[K](E[K](P)) = P

$(\rho, \kappa) \models_{\mathsf{RM}} P : \psi$

Analysis of processes

$$(\rho, \kappa) \models_{\mathsf{RM}} 0 : \psi \qquad \frac{(\rho, \kappa) \models_{\mathsf{RM}} P : \psi}{(\rho, \kappa) \models_{\mathsf{RM}} (\nu\, n) P : \psi}$$

$$\frac{(\rho, \kappa) \models_{\mathsf{RM}} P_1 : \psi \;\wedge\; (\rho, \kappa) \models_{\mathsf{RM}} P_2 : \psi}{(\rho, \kappa) \models_{\mathsf{RM}} P_1 | P_2 : \psi}$$

$$\frac{(\rho, \kappa) \models_{\mathsf{RM}} P : \psi}{(\rho, \kappa) \models_{\mathsf{RM}} !\, P : \psi}$$

Given a process, these clauses define a monotone function on complete lattices; its least fixed point is the analysis result.

Semantic properties

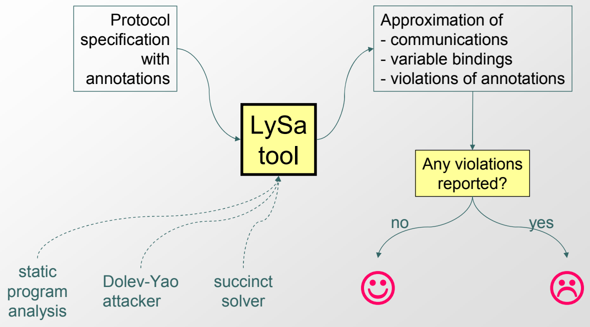
- Theorem: The analysis information is preserved under evaluation:

  If $P \to_{\mathcal{R}} Q$ *and* $(\rho, \kappa) \models_{\mathsf{RM}} P : \psi$
  then $(\rho, \kappa) \models_{\mathsf{RM}} Q : \psi$.

- Theorem: If the analysis does not report any origin/destination violations then the reference monitor will never abort the semantics

  If $(\rho, \kappa) \models_{\mathsf{RM}} P : \emptyset$
  then RM *cannot abort* P

Validating the protocol



Dolev-Yao attacker

- The attacker can
  - receive and send messages on the network
  - encrypt and decrypt messages using known keys
  - create new keys, nonces, messages, etc
- We specify the attacker at the analysis level as a logical formula using ρ and κ and ψ
- It can be proved that this is the hardest attacker – any other attacker will be subsumed by this one.

$$(1)\ \wedge_{k\in\mathcal{A}_\kappa} \forall\langle V_1,\cdots,V_k\rangle \in \kappa : \wedge_{i=1}^{k} V_i \in \rho(z_\bullet)$$
$$(2)\ \wedge_{k\in\mathcal{A}^+_{\text{Enc}}} \forall\{V_1,\cdots,V_k\}^{\ell}_{V_0}[\text{dest } \mathcal{L}] \in \rho(z_\bullet) :$$
$$V_0 \text{ E } \rho(z_\bullet) \Rightarrow (\wedge_{i=1}^{k} V_i \in \rho(z_\bullet) \wedge (\neg \text{RM}(\ell,\mathcal{C},\ell_\bullet,\mathcal{L}) \Rightarrow (\ell,\ell_\bullet) \in \psi))$$
$$(3)\ \wedge_{k\in\mathcal{A}^+_{\text{Enc}}} \forall V_0,\cdots,V_k : \wedge_{i=0}^{k} V_i \in \rho(z_\bullet) \Rightarrow \{V_1,\cdots,V_k\}^{\ell_\bullet}_{V_0}[\text{dest } \mathcal{C}] \in \rho(z_\bullet)$$
$$(4)\ \wedge_{k\in\mathcal{A}_\kappa} \forall V_1,\cdots,V_k : \wedge_{i=1}^{k} V_i \in \rho(z_\bullet) \Rightarrow \langle V_1,\cdots,V_k\rangle \in \kappa$$
$$(5)\ \{n_\bullet\} \cup \lfloor \mathcal{N}_f \rfloor \subseteq \rho(z_\bullet)$$

The validation procedure

Definition: *P* guarantees static authentication if $(\rho,\kappa) \models_{RM} P : \varnothing$ and $(\rho,\kappa,\varnothing)$ is satisfied by the attacker formula

Definition: *P* guarantees dynamic authentication if *P* | *Q* cannot abort regardless of the choice of the attacker *Q*

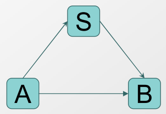
Theorem: If *P* guarantees static authentication then *P* guarantees dynamic authentication

Implementation details

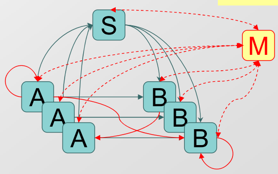
- Implemented in Standard ML.
- The Flow Logic specification of the analysis is (in a number of steps) transformed into a formula in ALFP (Alternation-free Least Fixed Point logic); the transformation involves encoding (potentially infinite) sets of terms by tree grammars.
- The Succinct Solver, a state-of-the-art constraints solver, will compute the least solution to the analysis problem, i.e. the least interpretation of the predicates satisfying the ALFP constraints.
- The overall time complexity is polynomial time in the size of the universe which is linear in the size of the protocol.

Protocol scenarios

So far:
• One initiator
• One responder
• One server

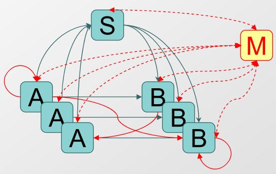


Generally:
• Many initiators
• Many responders
• One server



A scenario in LySa

- There are n principals
- Each of them can play the A-role (initiator) and the B-role (responder)
- Each principal shares two master key with the server (one for each role)
- A principal can initiate the protocol with any other principal
- Only the server can play the S-role
- The attacker can take on any role



A scenario for WMF protocol

Principals: 1..n
Attacker: 0

The initiator can start the protocol with any other legitimate principal

The responder is ready to interact with any principal (including the attacker)

The server can handle messages from/to any principal (including the attacker)

A

$(\nu_{i=1}^{n} K_i^A)(\nu_{j=1}^{n} K_j^B)$

0. $|_{i=1}^{n}\ |_{j=1}^{n}\ !\,(\nu\, K_{ij})$

1. $\langle A_i, S, A_i, \{B_j, K_{ij}\}_{K_i^A}^{A_{ij}^1}[\text{dest } S_{ij}^1]\rangle.$

3. $(\nu\, m_{ij})\ \langle A_i, B_j, \{m_{ij}\}_{K_{ij}}^{A_{ij}^2}[\text{dest } B_{ij}^2]\rangle.0$

B

2′. $|\ |_{j=1}^{n}\ !\,(S, B_j;\, y_j).$

2″. $|_{i=0}^{n}$ decrypt y_j as $\{A_i;\, y_{ij}^K\}_{K_j^B}^{B_{ij}^1}$ $[\text{orig } S_{ij}^1]$ in

3′. $(A_i, B_j;\, z_{ij}).$

3″. decrypt z_{ij} as $\{;\, z_{ij}^m\}_{y_{ij}^K}^{B_{ij}^2}$ $[\text{orig } A_{ij}^2]$ in 0

S

1′. $|\ |_{i=0}^{n}\ !\,(A_i, S, A_i;\, x_i).$

1″. $|_{j=0}^{n}$ decrypt x_i as $\{B_j;\, x_{ij}^K\}_{K_i^A}^{S_{ij}^1}$ $[\text{orig } A_{ij}^1]$ in

2. $\langle S, B_j, \{A_i, x_{ij}^K\}_{K_j^B}^{S_{ij}^2}[\text{dest } B_{ij}^2]\rangle.0$

Variants of WMF

A → S: A, E[Ka](B, K) S → B: E[Kb](A, K) A → B: E[K](m)	A → S: A, E[Ka](B, K) S → B: A, E[Kb](K) A → B: E[K](m)	A → S: A, B, E[Ka](K) S → B: E[Kb](A, K) A → B: E[K](m)
no errors reported	reports errors (A, B) ($\ell_\bullet$, B)	reports errors (A, B) (A, $\ell_\bullet$) ($\ell_\bullet$, B)

corresponds to real attacks!

Attacks on WMF variant

A → S: A, E[Ka](B, K)
S → B: A, E[Kb](K)
A → B: E[K](m)

(A, B) ∈ ψ

A → S: A, E[Ka](B, K)
S → M(B): A, E[Kb](K)
M(S) → B: A', E[Kb](K)
A → B: E[K](m)

B believes he is talking to A'

($\ell_\bullet$, B) ∈ ψ

M → S: M, E[KM](B, K)
S → M(B): M, E[Kb](K)
M(S) → B: A', E[Kb](K)
M → B: E[K](m)

B believes he is talking to A'

Attacks on WMF variant

A → S: A, B, E[Ka](K)
S → B: E[Kb](A, K)
A → B: E[K](m)

(A, ζ) ∈ ψ

A → M(S): A, B, E[Ka](K)
M(A) → S: A, M, E[Ka](K)
S → M: E[KM](A, K)
A → M(B): E[K](m)

(A, B) ∈ ψ

A → M(S): A, B, E[Ka](K)
M(A) → S: A, B', E[Ka](K)
S → B': E[Kb'](A, K)
A → M(B): E[K](m)
M(A) → B' : E[K](m)

(ζ, B) ∈ ψ

A → M(S): A, B, E[Ka](K)
M(S) → S: A, M, E[Ka](K)
S → M: E[KM](A, K)
M(A) → S: A, B, E[Ka](K)
S → B: E[Kb](A, K)
M → B: E[K](m)

How well are we doing?

- We compare ourselves against a selection of classical authentication protocols
- **Question 1**: robustness of protocol narrations:
  - is it important to distinguish initiator/responder roles for a principal?
  - is it important to have distinct master keys shared with the server for each role?
- **Question 2**: vulnerability in case of leaking old session keys

Question 1

distinguish between roles

same master keys for the two roles

parallel session attack

protocol $1 \le i, j \le n, i \ne j$	$A \ne B$ $\wedge_{i=0}^{n} K_i^A \ne K_i^B$	$A = B$ $\wedge_{i=0}^{n} K_i^A \ne K_i^B$	$A \ne B$ $\wedge_{i=0}^{n} K_i^A = K_i^B$	$A = B$ $\wedge_{i=0}^{n} K_i^A = K_i^B$
Wide Mouthed Frog	$\emptyset$	$\emptyset$	$\emptyset$	$(A_i, B_i), (S, S)$
with nonces	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Needham-Schroeder	(A_i, A_i)	(A_i, A_i)	(A_i, A_i)	(A_i, A_i)
with flaw corrected	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Amended Needham-Schroeder	(A_i, A_i)	(A_i, A_i)	(A_i, A_i)	(A_i, A_i)
with flaw corrected	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Otway-Rees	$\emptyset$	$\emptyset$	$(B_i, S), (S, B_i)$	$(B_i, S), (S, B_i)$
Yahalom	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
with BAN optimisation	$\emptyset$	$\emptyset$	$\emptyset$	$(A_i, B_i), (S, A_i), (S, B_i)$
Paulson's amendment	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Andrew Secure RPC	$(A_i^3, B_j^1), (B_i^2, A_j^4)$	$(A_i^3, B_j^1), (B_i^2, A_j^4)$	$(A_i^3, B_j^1), (B_i^2, A_j^4)$	$(A_i^3, B_j^1), (B_i^2, A_j^4)$
with BAN correction and flaw corrected	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

new flaw: B believ… he is talking to him…

Paulson's … h replay attack

Question 2: Leaking old keys

protocol	K_{12}^{old} is leaked
Wide Mouthed Frog	$(\ell_\bullet, B_2)$
with nonces	$\emptyset$
Needham-Schroeder with flaw corrected	$(B_2, \ell_\bullet), (\ell_\bullet, B_2)$
Amended Needham-Schroeder with flaw corrected	$\emptyset$
Otway-Rees	$\emptyset$
Yahalom	$(\ell_\bullet, B_2)$
with BAN optimisation	$\emptyset$
Paulson's amendment	$\emptyset$
Andrew Secure RPC with flaw corrected	$(A_1, \ell_\bullet)$
with BAN correction	$\emptyset$

Denning-Sacco attack

false positive!

OBS: Old keys and certificates are inserted explicitly in the attackers knowledge

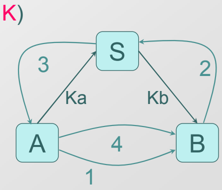
Yahalom protocol

Are both A and B convinced that K is fresh and known to both A and B both no others (except S)?

1. A → B: A, Na
2. B → S: B, E[Kb](A,Na,Nb)
3. S → A: E[Ka](B,K,Na,Nb), E[Kb](A,K)
4. A → B: E[Kb](A,K), E[K](Nb)

Does not mention Nb so the message could be a replay – or could it?

Nb is fresh and kept secret and so is K …



independent attribute analysis
versus
relational analysis

The nature of approximations

The analysis will occasionally report problems that are not really there

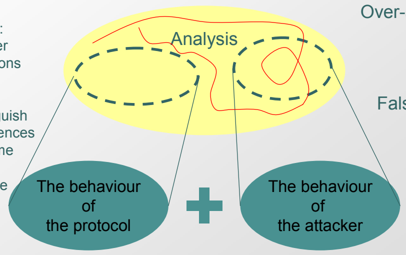
Independent attribute analysis

Flow insensitive: Ignores the order in which operations are performed

Does not distinguish between occurrences of the same name

Does not remove bindings when they are no longer relevant

…



Over-approximation

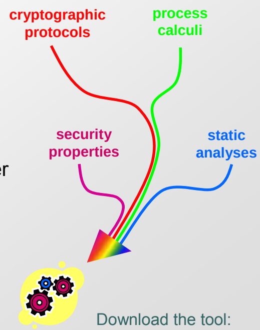
False positives

Process calculi

Language primitives for
- Symmetric cryptography
  - as presented here
- Asymmetric cryptography
  - including signatures
- Blinding
  - achieving anonymity
- Support scenarios with any number of principals

Fully automatic tool support with firm theoretical foundations

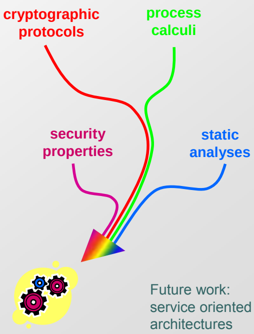
UML interface



Download the tool:
http://www2.imm.dtu.dk/cs_LySa/lysatool/

Security protocols and their properties

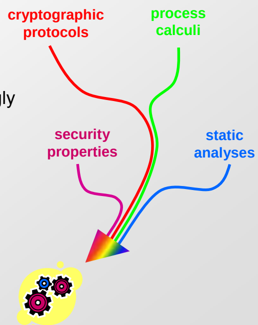
- Key exchange protocols
  - Authenticity, confidentiality, freshness
- Single Sign On protocols
  - Authenticity
- WiMAX protocols
  - Authenticity
- Voting protocols:
  - Verifiability: Voters can verify that their votes have been counted
  - Accuracy: No votes can be altered and only validated votes count in the final tally
  - Democracy: Only eligible voters can vote and they can only vote once
  - Fairness: No early results can be obtained from the voting
  - Privacy: Voters and their votes cannot be linked together..



Future work:
service oriented architectures

Static analysis

- The analysis is very simple:
  - Independent attribute analysis
  - Flow insensitive analysis
  - Context insensitive analysis
- But still, it correctly identifies surprisingly many flaws!!! ☺
- Future work:
  - More powerful analyses
    - Relational
    - Flow sensitivity
    - Context sensitivity
  - More security properties



Thank you for your attention!

Collaborators:
Flemming Nielson, DTU
Helmut Seidl, TUM
Pierpaolo Degano, Pisa
Mikael Buchholtz, DTU
Chiara Bodei, Pisa
Christoffer R. Nielsen, DTU
Han Gao, DTU

With special thanks to:
Michele Curti, Stephen Gilmore, Valentin Haenel, Jane Hillston, Carlo Montangero, Lara Perrone, Corrado Priami, Simone Semprini, Nikolaj Kaplan, Steffen Hansen, Jakob Skriver, Esben Heltoft Andersen, Ye Zhang, Ender Yuksel

The work has been supported by
- DEGAS (EU, 5th framework, IST-2001-32072)
- SENSORIA (EU, 6th framework, IST-2005-016004)
- LOST (Danish Natural Science Research Council)
- SiES (Danish Research Academy, IT research programme)

Selected publications (1)

- Bodei, Buchholtz, Degano, Nielson, Riis Nielson: *Static validation of security protocols.* Journal of Computer Security, 2005
- Bodei, Buchholtz, Degano, Nielson, Riis Nielson: *Automatic validation of protocol narrations.* CSFW 2003
- Bodei, Buchholtz, Degano, Nielson, Riis Nielson: *Control Flow Analysis can find new flaws too.* WITS, 2004
- Buchholtz, Nielson, Riis Nielson: *A calculus for control flow analysis of security protocols.* International Journal of Information Security, 2004

Selected publications (2)

- Nielsen, Riis Nielson: *Static Validation for Blinding.* Nordic Journal of Computing, 2006
- Nielsen, Andersen, Riis Nielson: *Static Validation of a Voting Protocol*. ARSPA, 2005
- Hansen, Skriver, Riis Nielson: *Using static analysis to validate the SAML Single Sign-on Protocol.* WITS, 2005
- Buchholtz, Montangero, Perrone, Semprini: *For-LySa: UML for Authentication Analysis.* Global Computing International Workshop, 2004
- Buchholtz: *Automated Analysis of Infinite Scenarios.* Trustworthy Global Computing 2005.