Securing Insecure Protocols

Y. Chevalier, M. Rusinowitch
Irit-UPS, Loria-Inria
Workshop IM&CTCPA

Plan

Research in Protocol Security

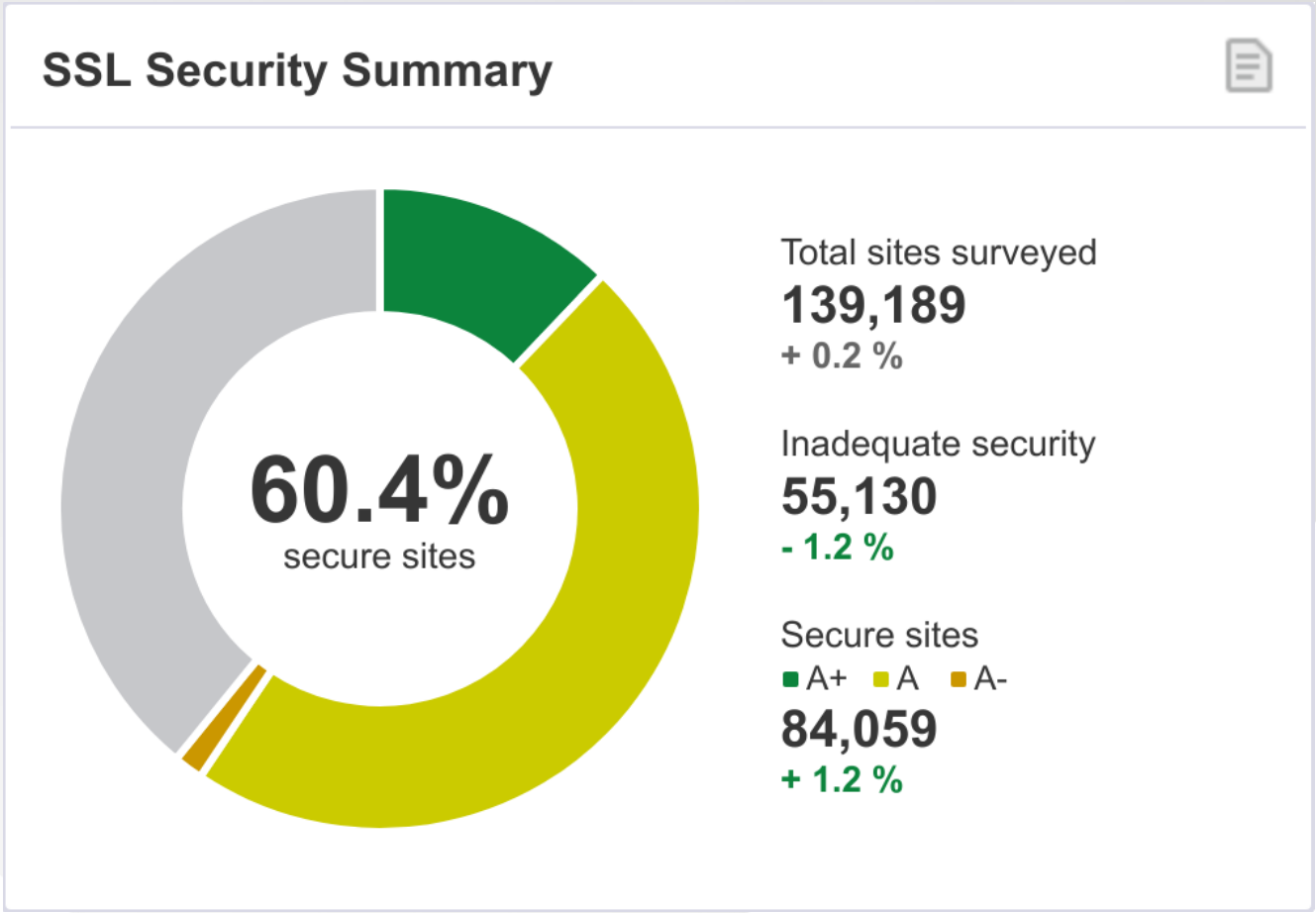
Main Topics

- Validation of the security of a protocol
- Find attacks on existing protocols
- Develop new secure protocols

Assumptions

- Flawed protocols are either patched or withdrawn
- Interoperability concerns are less important than security

Unrealistic Assumptions



Comments

- Interoperability often more important
- Patching an attack may lead to new attacks

Defense-in-Depth Approach

Defense-in-Depth

Intelligence + counter-measures:

- Use a monitor to get information on the current state of the system
- Deploy counter-measures against detected attacks

System protected from attacks even when components are insecure

Defense-in-Depth for Security Protocols

1. Identify attacks on deployed protocols
2. Identify a subset of agents that can cooperate to try to mitigate the attack
3. Monitor the executions of this protocol

   Abort an execution assessed to be an attack
4. This presentation: differentiate a normal execution from an attack

Defense-in-Depth Approach

Defense-in-Depth

Intelligence + counter-measures:

- Use a monitor to get information on the current state of the system
- Deploy counter-measures against detected attacks

System protected from attacks even when components are insecure

Defense-in-Depth for Security Protocols

1. Identify attacks on deployed protocols
2. Identify a subset of agents that can cooperate to try to mitigate the attack
3. Monitor the executions of this protocol

   Abort an execution assessed to be an attack
4. This presentation: differentiate a normal execution from an attack

Rest of this Presentation

Formalization

- Knowledge sharing defined by a monitor protocol
- Definition of an attack presentation
- Attack detectability decision problem

Existing Results

- Static equivalence: routinely employed to prove secrecy as indistinguishability
- Compilation of protocols computing the set of tests and constructions an implementation of the protocol should perform
- Apply the latter to detect attacks in practice

Plan

Strands and Traces

Strand

- Points of sequential execution in a distributed system
- Definition of protocol roles from a MSC presentation

Trace

Sequence of reception (?) and emission (!) of messages

Presentation of a distributed system

- A finite set of strands Σ
- A function tr mapping each strand to a trace

$$(\Sigma, \mathrm{tr})$$

Application

Protocol Definition

- Protocols defined by a MSC
- A protocol is a couple $(\Sigma_P, \mathrm{tr}_P)$
- The strands in Σ_P are the roles of the protocol

Protocol Execution

- Agents execute roles of the protocol
- Each agent can execute only one role
- Execution of a protocol $(\Sigma_P, \mathrm{tr}_P)$ defined by a tuple $(\Sigma_E, \mathrm{tr}_E, R_E)$ with:

$$R_E : \Sigma_E \rightarrow \Sigma_P$$

- Note: An agent playing several roles is represented using distinct strands

Monitor Protocol

Fact

All messages constructed by an agent can be constructed from its inputs

$\Rightarrow$ messages sent by an agent provide no additional information

Monitor

- A monitor protocol for $P = (\Sigma_P, \mathrm{tr}_P)$ is a protocol $(\Sigma_P, \mathrm{tr}_M)$
- Strands of the monitor are roles of the protocol
- Constraint: a trace of a strand in the monitor has the same input messages as the corresponding trace in P

  Limits available information to that available in one session
- Messages sent by all strands in the monitor protocol are received by a unique agent, the Monitor

  Ensures different executions have the same structure

Plan

Compilation of a protocol

Compilation Target: Active Frames

Sequential processes, 2 possible activities:

Receive x ($?x$): receives a message and store it in x

Check the messages received according to their expected pattern

Send x ($!x$): sends the value stored in x

Construct the value from available ones

Challenges

Inter-operability: Compute how a message sent is constructed from available values

Prudence: Compute a set of checks that is both minimal and as complete as possible

Modeling

Available constructions

- Functions in a library
- Abstract Data Type approach: semantics of functions defined by an equational theory

Deduction system

$(\{x_1, \ldots, x_n \rightarrow f(x_1, \ldots, x_n)\}_{f \in \mathscr{F}_p \subseteq \mathscr{F}}, \mathscr{F}, \mathscr{E})$:

- $\mathscr{F}$ is the set of function symbols and constants used to describe the library
- $\mathscr{F}_p$ is the set of functions in the library
- $\mathscr{E}$ is an equational theory

$$\mathrm{car}(\mathrm{cons}(x, l)) = x$$

Simple Example

Deduction system

$(\{x \to h(x)\}, \{h/1\}, \emptyset)$, model for perfect hash functions

Example commitment protocol

$(?h(M); ?M; !\mathrm{ok})$: a role receives a commitment on a message M, then later the message itself.

Target

$(?x_1; ?x_2; !x_3)$

1. On the first message, no check is possible
2. After receiving the second message the role must check $h(x_2) = x_1$
3. There is an infinite number of possible tests, but they are all entailed by this one
4. $x_3 = \mathrm{ok}$, a simple construction

Simple Example, Variation

Deduction system

$(\{x, y \to h(x,y)\}, \{h/2\}, \emptyset)$, HMAC of message x with key y

Example protocol

$(?K; ?M; !h(M,K))$: Initial knowledge is a shared secret K, a role receives a message M and replies with the HMAC of this message.

Target

$(?x_1; ?x_2; !x_3)$

1. No meaningful check between the first two messages
2. x_3 is constructed from x_1, x_2 with the equation $x_3 = h(x_2, x_1)$

More Complex Example (1/2)

Deduction System

The operations in the group $(\mathbb{Z}, 0, +, -)$, the equations in the theory of groups, terms are vectors of $\mathbb{Z}^n$, where n is the number of constants

$$a_1 + a_2 + a_2 + (-a_3) \longrightarrow (1, 2, -1)$$

Protocol

$(?M_1; \ldots; ?M_k)$ where M_i are vectors in $\mathbb{Z}^n$

More Complex Example (2/2)

Mathematical model

- trace $\varphi = (?M_1; \ldots; ?M_k)$ interpreted as:

$$\begin{array}{rccc} f_\varphi : & \mathbb{Z}^k & \rightarrow & \mathbb{Z}^n \\ & (x_1, \ldots, x_k) & \mapsto & x_1 \cdot M_1 + \ldots + x_k \cdot M_k \end{array}$$

- Every test satisfied on the trace can be reduced to $\vec{x} \in \mathrm{Ker} f_\varphi$
- A message M can be constructed iff $M \in \mathrm{Im} f\varphi$

Cryptographic Protocols case

- Similar to this mathematical model
- On terms (expressions) in a first-order signature not vectors!
- Almost always an infinite number of possible tests
- Difficulty: existence and computation of a “finite basis” for these tests

Interoperable Implementation

Interoperability

If the messages effectively received follow the pattern given in the protocol, the responses also follow this pattern

Notations

- input(t), the message received in the trace t presented as messages sent
- $\varphi \cdot t$: when t contains only sent messages, the trace of messages sent and received given the actions described in the active frame φ

Conform implementation

(Σ_P, ϕ) is a conform implementation of $(\Sigma_P, \mathrm{tr}_P)$ if:

$$\forall r \in \Sigma_P, \phi(r) \cdot \mathrm{input}(\mathrm{tr}_P(r)) = \mathrm{tr}_P(r)$$

Question: Can a recipe constructing a term from a set of terms be computed?

Ground reachability problem

Interoperable Implementation

Interoperability

If the messages effectively received follow the pattern given in the protocol, the responses also follow this pattern

Notations

- input(t), the message received in the trace t presented as messages sent
- $\varphi \cdot t$: when t contains only sent messages, the trace of messages sent and received given the actions described in the active frame φ

Conform implementation

(Σ_P, ϕ) is a conform implementation of $(\Sigma_P, \mathrm{tr}_P)$ if:

$$\forall r \in \Sigma_P, \phi(r) \cdot \mathrm{input}(\mathrm{tr}_P(r)) = \mathrm{tr}_P(r)$$

Question: Can a recipe constructing a term from a set of terms be computed?

Ground reachability problem

Interoperable Implementation

Interoperability

If the messages effectively received follow the pattern given in the protocol, the responses also follow this pattern

Notations

- input(t), the message received in the trace t presented as messages sent
- $\varphi \cdot t$: when t contains only sent messages, the trace of messages sent and received given the actions described in the active frame φ

Conform implementation

(Σ_P, ϕ) is a conform implementation of $(\Sigma_P, \mathrm{tr}_P)$ if:

$$\forall r \in \Sigma_P, \phi(r) \cdot \mathrm{input}(\mathrm{tr}_P(r)) = \mathrm{tr}_P(r)$$

Question: Can a recipe constructing a term from a set of terms be computed?

Ground reachability problem

Prudent Implementation

Context

- Term C build with public functions on variables $x_1, \ldots, x_n$ and names
- $C[x_1, \ldots, x_n] \cdot (t_1, \ldots, t_n) = C[t_1, \ldots, t_n]$

Constructibles Tests

Equalities $C = C'$ with C, C' contexts

Tests satisfied by a trace *tr*

$tr \models C = C'$ if $C \cdot tr =_{\mathscr{E}} C' \cdot tr$

Test set of a trace (Kernel)

The test set of *tr* is the set of constructible tests satisfied by *tr*

Refinement

Definition

t refines t' iff every test satisfied on t' is satisfied on t iff the test set of t' is included in the test set of t.

Finite Basis Property

For every trace t one can compute a finite set of tests T such that every trace t' that satisfies T also satisfies all equations in the test set of t

Usage

- Compute the finite basis for input messages in the prefixes of the trace of a role
- Tests are performed as soon as possible

Prudent implementation

Active frame that accepts only sequences of messages that pass all tests in the test set of the trace defining the role

Implementation of a Monitor

Principles

- A protocol monitor is a protocol
- We only need to compute a conform implementation

  Tests are performed in the implementation of the protocol
- The monitor receives the constructions on the messages received by the strands
- The constructions do not depend on an actual execution but their result does

Plan

Generic Approach

Attack Presentation

- An attack is an execution of the protocol
- It has to be given together with the execution expected for the same set of agents (strands in the execution) for any further computation to make sense

Attacks and Monitor

- The active frames of the monitor protocol compute messages sent to the monitor for the each execution
- Consider:
  - the sequence of messages received by the monitor in the normal execution
  - the sequence of messages received by the monitor in the attack
- The attack always refine the normal execution (for prudent implementations)
- The attack is detectable if it is not refined by the normal execution

Effective Computation of tests

Case of Undetectable Attacks

- If the attack is not detectable, we know there is no test that can differentiate an attack from a normal execution
- The monitor protocol has to be changed so that participants provide more information to the monitor

Note on possibility of detection

An orwellian monitor that receives all the information of all participants can guarantee that the information sent by one is indeed received by the other

Effective Computation of tests

Case of Detectable Attacks

- The monitoring of an attack can be implemented just like a normal protocol specification
- The execution of the monitored protocol is aborted when this implementation of the attack monitor executes correctly

  i.e., if all tests are satisfied
- Free generalization:

  all refinements of the attack are detected

Plan

Extensions

Conclusion

- Computing the implementation of the monitor protocol is automated
- But devising the monitor protocol is too protocol-specific or even business-case specific that cannot be securely guessed
- A trust relationship between strands in the protocol is needed to automate further

Focus on a set of properties

- Goal is to amend protocols automatically when a flaw has been discovered by an analysis tool
- We plan to rely on the analysis to construct the protocol monitor and the tests automatically

Get rid of the monitor

- Information sharing among participants seems easier to automate
- We plan to provide a decentralized monitor for the usual functions
- Needs trust relationship between strands in the execution

Extensions

Conclusion

- Computing the implementation of the monitor protocol is automated
- But devising the monitor protocol is too protocol-specific or even business-case specific that cannot be securely guessed
- A trust relationship between strands in the protocol is needed to automate further

Focus on a set of properties

- Goal is to amend protocols automatically when a flaw has been discovered by an analysis tool
- We plan to rely on the analysis to construct the protocol monitor and the tests automatically

Get rid of the monitor

- Information sharing among participants seems easier to automate
- We plan to provide a decentralized monitor for the usual functions
- Needs trust relationship between strands in the execution

Extensions

Conclusion

- Computing the implementation of the monitor protocol is automated
- But devising the monitor protocol is too protocol-specific or even business-case specific that cannot be securely guessed
- A trust relationship between strands in the protocol is needed to automate further

Focus on a set of properties

- Goal is to amend protocols automatically when a flaw has been discovered by an analysis tool
- We plan to rely on the analysis to construct the protocol monitor and the tests automatically

Get rid of the monitor

- Information sharing among participants seems easier to automate
- We plan to provide a decentralized monitor for the usual functions
- Needs trust relationship between strands in the execution