Wanted: Safe or Alive

Mohammad Torabi Dashti
CWI, Amsterdam
Liveness aspects of security properties

Joint work with
J. Cederquist, U/Twente
Security properties as we know

• Your password does not leak.
• The access policy is not violated.
• The bad guy is not authenticated.
• Access to private resources is not publicly authorized.
• Your vote is not publicly disclosed.
• Something bad does not happen.
Properties of systems

• Safety: Something bad will not happen.
• Liveness: Something good will eventually happen.
• [Alpern, Schneider 1984] Any (trace) property of systems can be expressed as intersection of safety and liveness properties.
Basic liveness properties

• Guaranteed service: Each request is followed by the corresponding service.
• Termination: Each run of the program will finally terminate.
Security as liveness

• Fairness in exchange: Two (or more parties) can exchange their items in a *fair* manner.
• “Fair exchange” appears in fair payment, certified e-mail, non-repudiation and contract signing protocols, for instance.
• Related problems: Denial of service attacks, Distributed consensus, Program termination
Fair exchange protocols

- Players do not trust each other.
- Players can quit the protocol whenever they want.
- [Even, Yacobi 1980] Fair exchange without a TTP is impossible.
Fair exchange with TTP

• TTP Bottleneck.
Optimistic fair exchange protocols

• Resilient channels: Modeling vs. implementation.
Fair exchange properties

[Asokan 1998]:

• **Effectiveness** (a.k.a. sanity check)

• **Fairness**

  Upon termination, A has B’s item and B has A’s, or none of them has lost its item.

• **Timeliness**

  Honest players will eventually terminate.
Fair exchange properties

[Asokan 1998]:

• **Effectiveness**

• **Fairness: Safety property**
  Upon termination, A has B’s item and B has A’s, or none of them has lost its item.

• **Timeliness: Liveness property**
  Honest players will eventually terminate.
Does the liveness part matter?

• [Gurgens, Rudolph 2002] There are attacks against (a family of) fair exchange protocols that prevent the termination of the protocol, while the safety aspect is not violated.

• Non-termination: Deadlocks vs. Livelocks.

• Looping agents are prevalent in fair exchange protocols.
Formal verification of fair exchange properties

- Formal verification of security protocols:
  - $M$: A model of the protocol (in a process algebra, …)
  - $I$: A model of the intruder
  - $P$: The desired security property
  - Check whether the property $P$ holds when the protocol $M$ runs while the intruder $I$ is present.
Formal verification of security protocols –I

- [Dolev, Yao 1984] The Dolev-Yao model of intruder (DY):
  - Assumes perfect cryptography.
  - The intruder controls the entire network traffics.
  - Symbolic manipulation of messages: decomposing, pairing, replaying, removing, encryption and decryption (if the intruder knows the associated keys).

- [Cervesato 2001] DY is the most powerful intruder (under certain assumptions).
The Dolev-Yao intruder
(a process algebraic spec)

\[ DY(X) = \sum \sum_{m \in \text{Msg} \ p \in E} \text{recv}(p, m, \text{net}).DY(\{m\} \cup X) + \]

\[ \sum \sum_{m \in \text{Msg} \ p \in E} \text{send}(\text{net}, m, p).DY(X) \triangleleft \text{synth}(m, X) \triangleright \delta + \]

\[ \kappa.\delta \]
Formal verification of security protocols - II

• The Dolev-Yao intruder is too strong for proving liveness properties.
• Passive DY is too weak.
• Resilient communication channels (RCC) are required for liveness properties to hold.
• DY + RCC?
Definition of RCC - I

• RCC assumption says that each received message by the network has to be eventually delivered to its destination.

• $RCC_n$ says that if $t$ instances of a message are received by the network, $\min(t, n)$ number of them are guaranteed to be eventually delivered to their destination.
Definition of RCC - II

• RCC in modeling: Unbounded memory.
• $\text{RCC}_n$ in modeling: Bounded memory if finite number of different messages are produced by the agents.
• Theorems on $\text{RCC}_n$:
  – If a liveness property holds in $\text{RCC}_n$, it holds in RCC.
  – There could be violations of liveness in $\text{RCC}_n$, while the very same property actually holds in RCC.
Formalization of RCC

• There is an intruder that has absolute control over the communication media.
• Agents communicate via the controlled network.
• Which traces of the resulting model satisfy RCC?
  – Complete finite traces, infinite traces
Formalization of RCC – finite traces

• When the network is shut down:
  – All the received messages were already delivered. (of course, deliverable messages)
  – There is no message to be received from the agents.
Formalization of RCC – infinite traces

• If the network is shut down (by the intruder) at some point, it has to comply with the previous criterion.

• Infinite traces has to be *fair*, respect to a certain subset of action.

• Informally: No excessive collaboration from the network (or the intruder). It does not need to be fair regarding delivering of those messages which are not required to be delivered, according to RCC.
Off the track: 
Fairness constraints in modeling

• Fair scheduling: All the competing agents has to be fairly treated and get access to resources often enough.

• If an action (of an agent) is infinitely often available, it has to be infinitely often performed.
DY + RCC

• The DY intruder plays also the role of the network. It can disrupt communication and need not to comply with RCC.

• Fairness constraint: Those traces which do not respect RCC are pruned. The desired (liveness) property is then checked on the remaining traces.

• Can this fairness constraint be expressed in μ-calculus? No, as far as we know. However, we do not have a proof of it yet.
Simulating $DY + RCC = I(X,Y)$

$$I(X,Y) = \sum_{m \in \text{Msg}} \sum_{p \in \text{E}} \text{recv}(p,m,\text{net}).I(\{m\} \cup X, \{m\} \cup Y) + \sum_{m \in Y} \sum_{p \in \text{E}} \text{send}(\text{net}, m, p).I(X, Y \setminus \{m\}) + \sum_{m \in \text{Msg}} \sum_{p \in \text{E}} \text{send}^\uparrow(\text{net}, m, p).I(X, Y)$$

$\triangleleft \text{synth}(m, X) \land m \not\in Y \triangleright \delta$
I(X,Y): How it works?

- The role of resilient buffer $Y$.
- The role of $send^\uparrow$ actions.
- A fairness constraint $P_A$ which prunes daggered actions: easy to express in efficiently checkable fragments of $\mu$-calculus.
- Theorem: $I(X,Y)$ under $P_A$ constraint is equivalent to $DY+RCC$. 
More about $I(X, Y)$

- If only parts of network is resilient, it can be modeled by $I(X, Y)$, with controlling the messages which are added to $Y$.
- An intruder which removes messages from $X$, but is pushed to collaborate is not equivalent to $DY + RCC$. 
Verifying fair exchange protocols

• Divide the properties into safety and liveness.
• Check the liveness part with $I(X,Y)$ and $P_A$.
• Check the safety part with $DY$ (can also be checked with $I(X,Y)$ and $P_A$, but it would then be more expensive).
Case Studies

Joint work with
J. Cederquist, U/Twente
R. Corin, U/Twente
H. Jonker, TU/e
S. Nair, VU
A fair payment protocol

- Customer $C$ owns a smartcard.
- Customer exchanges a payment order (signed by its smartcard) for an item from Vendor $V$.
- Customer can reject exchanges that do not happen in time.
- Banking system plays the role of TTP.
- $C$ and $V$ do not trust each other.
- The exchange between $C$ and $V$ is verified to be fair.
A non-repudiation protocol- I

- A wants to send a message $m$ to B. A wants a receipt if B receives $m$. B needs a proof, associated to $m$, that A can not deny sending $m$. This exchange should be fair.
- There has been a design/attack/re-design/attack loop in the community.
A non-repudiation protocol - II

- We proposed a very light-weight non-repudiation protocol, mainly removing all the labels.
- A formal verification was performed to provide more “confidence” in the design.
- We found non-trivial attacks, amongst them liveness attacks, against the protocol when some parts of messages are omitted or slightly changed.
A DRM protocol - I

• [S. Nair et al 2004] Designed a DRM-protocol which provides content exchange between customers, while preserving DRM.
• The customers are assumed to own compliant devices (though there is a mechanism to detect circumvented devices in real implementations).
• Fair exchange was missing in the first proposal. We added an optimistic fair exchange sub protocols, which provide strong fairness.
A DRM protocol - II

- Fair exchange properties were model checked.
- The intruder can not corrupt the devices, but it has control over the (resilient) network and can deliberately turn the devices off.
- The protocol has certain advantages over other fair exchange protocols proposed for trusted agents... The paper is still in preparation.
Summary

• Liveness is an aspect of security.
• What is RCC and why it is needed.
• How we can efficiently simulate DY+RCC.
• In which contexts fair exchange finds importance.
• How to formally verify fair exchange properties.