Verifying Second-Level Security Protocols

G.Bella, C.Longo, L.C.Paulson

UNIVERSITÀ di CATANIA
Dipartimento di Matematica e Informatica

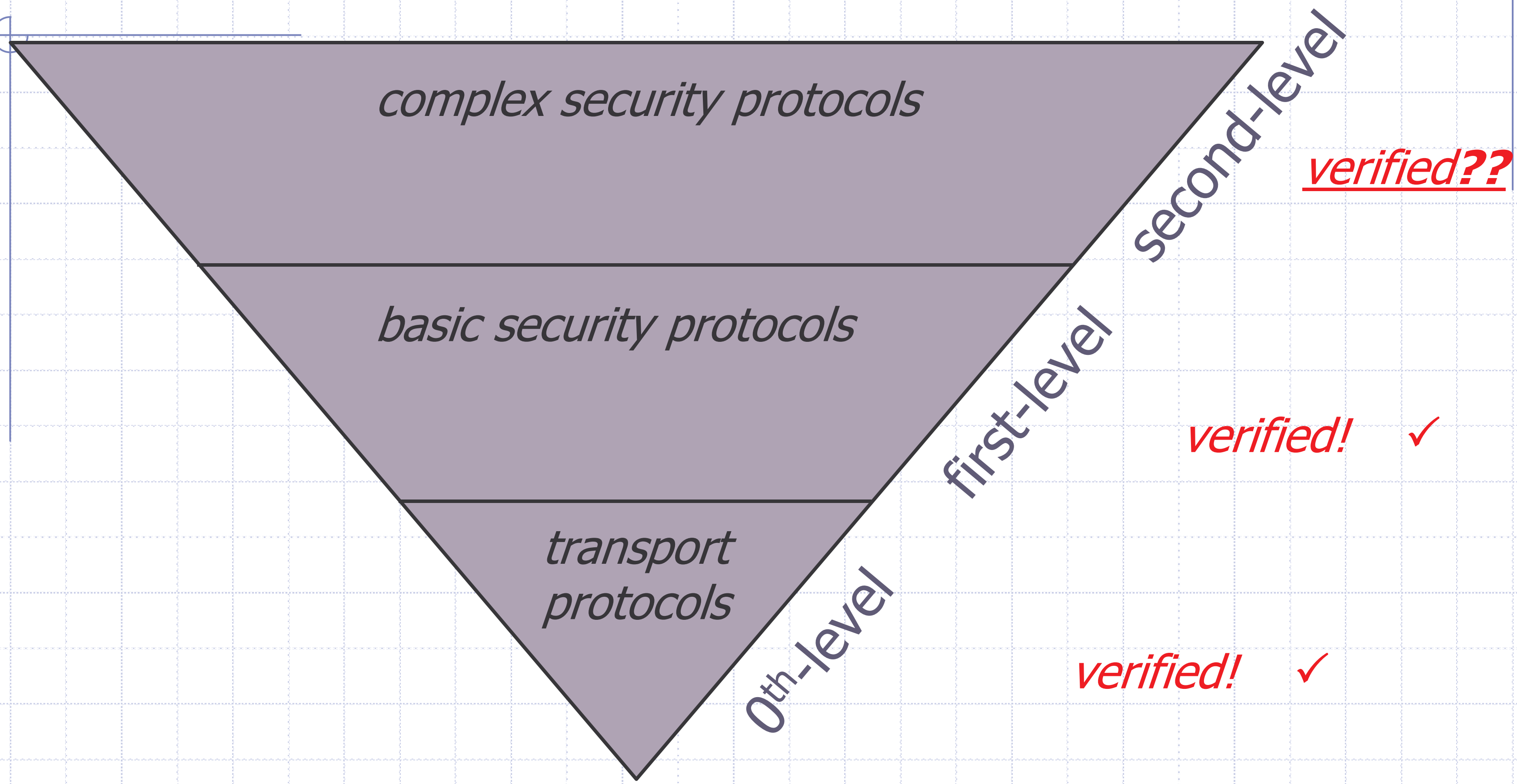
UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Goals in distributed systems

- Complex security goals: certified e-mail, contract-signing, non-repudiation, delegation…
- Basic security goals: confidentiality, authentication, integrity.
- Basic communication goals: routing, transmission of raw byte streams…

Different goals require different kinds of protocol.

A hierarchy of protocols



Each protocol relies upon underlying protocols.

Certified e-mail delivery

Hmm, must send him an e-mail…

… but in such a way that he can't claim I didn't…

OK, I'll send it using that certified e-mail protocol…

Then I'll get a receipt when he sees the message!

Certified e-mail delivery

Hmm, an e-mail from her… what a weird protocol though…

… damn it! It means she now has a receipt that I have read her message!

At least she couldn't get a receipt until I opened her email!

Certified e-mail (Abadi et al.)

Abbreviations:

$$h_S = \text{Hash}(q, r, \{m\}_k)$$
$$h_R = \text{Hash}(q', r', em')$$
$$S2TTP = \{S, k, R, h_S\}_{(\text{pubEKTTP})}$$

Steps:

1.	S	$\longrightarrow$	R	:	TTP, $\{m\}_k, q, S2TTP$
2.	R	$\xrightarrow{\text{SSL}}$	TTP	:	$S2TTP'$, RPwd, h_R
3.	TTP	$\xrightarrow{\text{SSL}}$	R	:	k', h'_R
4.	TTP	$\longrightarrow$	S	:	$\{S2TTP''\}_{(\text{priSKTTP})}$

This is a *second-level protocol*: it refers to SSL.

How the protocol works

- Sender sends the message, encrypted using a session key, to Recipient.
- If R wants to proceed, R asks the Trusted Third Party for the key.
- The TTP releases the key to R and simultaneously gives a receipt to S.

Verifying second-level protocols

- Shmatikov and Mitchell have model-checked a contract-signing protocol
- Abadi and Blanchet have verified the certified e-mail protocol using Blanchet's verifier

Our contribution

- Identify the concept of second-level protocols
- Enrich our inductive approach to
  1. model the goals of first-level protocols (here, secure channels)
  2. adapt Dolev-Yao's threat model
  3. express and verify the protocol goals

Primitive events

- Says A B X: A tries to send message X to B
- Gets B X: B receives message X from network
- Notes A X: A stores message X as an internal state change

These primitives can model typical first-level goals: *secure channels*

Specifying a protocol inductively

Protocol DAP

1.	$A \longrightarrow B$	:	A, Na
2.	$B \longrightarrow A$	:	$\{Na\}_{Kb^{-1}}$

```
Nil:     "[] ∈ dap"

DAP1:    "⟦evs1 ∈ dap; Nonce Na ∉ used evs1⟧
          ⟹   Says A B {|Agent A, Nonce Na|} # evs1 ∈ dap"

DAP2:    "⟦evs2 ∈ dap;
          Gets B {|Agent A, Nonce Na|} ∈ set evs2⟧
          ⟹   Says B A (Crypt (priSK B) (Nonce Na))
             # evs2 ∈ dap"

Recp:    "⟦evsr ∈ dap; Says A B X ∈ set evsr⟧
          ⟹   Gets B X # evsr ∈ dap"

Fake:    "⟦evsf ∈ dap; X ∈ synth(analz(knows Spy evsf))⟧
          ⟹   Says Spy B X # evsf ∈ dap"
```

1. Modelling secure channels

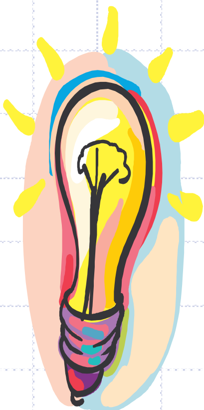
- **Authentication:** allow references to sender A in event $\mathsf{Says}\ A\ B\ X$, otherwise forbidden.
  Reception event $\mathsf{Gets}\ B\ X$ naturally hides sender.

- **Confidentiality:** use $\mathsf{Notes}\ A\ \{A, B, X\}$ followed by $\mathsf{Notes}\ B\ \{A, B, X\}$.
  Reception is not guaranteed in general.

- **Guaranteed delivery:** impose introduction of reception event $\mathsf{Gets}\ B\ X$.
  If also confidential, impose $\mathsf{Notes}\ B\ \{A, B, X\}$.

2. Adapting the threat model

What's the threat model for second-level protocols??



Simply Dolev-Yao, assuming that the first-level protocol works. The Spy can also use the protocol.

The formalisation of the goals just shown yields this threat model naturally.

Example: formalising message 2

```
CM2:
 "⟦evs2 ∈ certified_mail;
   Gets R {|Agent S, Agent TTP, em', Number AO,
            Number cleartext', Nonce q', S2TTP'|}
     ∈ set evs2;
   TTP ≠ R;
   hr = Hash {|Number cleartext', Nonce q',
                response S R q', em'|} ⟧
 ⟹ Notes TTP {|Agent R, Agent TTP, S2TTP',
                    Key(RPwd R), hr |}
          # evs2 ∈ certified_mail"
```

Query/response mechanism between sender and receiver. Hides a Hash.

R sends message to TTP on channel that is SSL protected and delivery guaranteed. The message "magically" reaches TTP.

Threat model: Spy sees message received by R but not that noted by TTP.

3. Modelling the new goals

Consider an e-mail *m*, its delivery receipt *d*, a sender *S*, an intended recipient *R*.

Goals of certified e-mail delivery (abstract version):

Let evs *be a generic trace of the protocol model;*
let Says $S\ R\ X$ *be an event in* evs *such that* X
features m*;*
then

$$m \in \text{analz}(\text{knows}\ R\ evs) \iff d \in \text{analz}(\text{knows}\ S\ evs).$$

Must be made precise given a specific protocol.

Example: sender's guarantee

```
"⟦Says S R {|Agent S, Agent TTP, Crypt K (Number m),
               Number AO, Number cleartext, Nonce q, S2TTP|}
    ∈ set evs;
  S2TTP = Crypt (pubEK TTP) {|Agent S, Number AO,
                  Key K, Agent R, hs|};
  Key K ∈ analz(knows Spy evs);
  evs ∈ certified_mail;
  S≠Spy⟧
  ⟹ R ∈ bad
    & Gets S (Crypt (priSK TTP) S2TTP) ∈ set evs"
```

If the Spy can see the message, then *R* is compromised; even then, *S* gets his receipt!

Other guarantees proved

- If neither peer is compromised, then the session key remains secure.
- The recipient (who may be the Spy) does not get the key until the sender gets his receipt.
- The recipient will get the key if the sender's receipt exists.

Differences from earlier proofs

- Distrust of peer, who may be dishonest
- Spy's knowledge no longer the main issue: new reasoning methods needed
- Subtle issues: for instance, only TTP can accept SSL connections
- Issues in the modelling of secure channels

Conclusions

- Second-level protocols are not difficult to verify
- A general-purpose proof tool (Isabelle) lets us modify the model without resorting to programming
- The use of logic lets us express properties abstractly and naturally