Tamarin Prover Tutorial

David Basin, Cas Cremers

Summer School on Real-world Crypto and Privacy

2023

About us

- David Basin
  - ETH Zurich since 2003. Heads Information Security Group
  - Research on Formal Methods for Security
    Tamarin, Monpoly, ActionGUI, VerifiedScion, CookieBlock, ...
  - Also applications, e.g., the SCION Internet
  - Enjoy both academic and industrial research
- Cas Cremers: Professor @CISPA

  He will tell you more himself!
- We are both looking for Postdocs interested in our topics.

Why attend this tutorial?

You are a protocol designer, quality assurance engineer, security researcher/grad student. But the sun is out and the water is warm.

- To learn how to:
  - Model cryptographic protocol
  - Model the adversary
  - Specify properties
- Understand verification and attack finding
- Gain experience with a state-of-the-art tool: **Tamarin**

Overall: deepen your knowledge of security protocols, their specification, and their machine-supported verification.

Tutorial's structure

Morning:

- Overview, motivation, basics (David)
- Modeling, demos (Cas)
- break
- Exercise I, Naxos (you)

Afternoon:

- More modeling, advanced primitives (Cas)
- EMV (David)
- break
- Exercise II (you)

Is this relevant the real world???

5G Authentication

EMV (Europay, Mastercard, Visa)

ETH-Forscher warnen

Sicherheitslücke bei Visa-Kreditkarten entdeckt

Dienstag, 01.09.2020, 11:49 Uhr

Dieser Artikel wurde **8**-mal geteilt.

- Forschende der ETH Zürich haben eine Sicherheitslücke bei Visa-Kreditkarten entdeckt.
- Damit könnten Betrügerinnen und Betrüger Beträge von Karten abbuchen, die eigentlich mit einem Pin-Code bestätigt werden müssten.
- Andere Unternehmen wie Mastercard oder American Express sind laut ETH nicht betroffen.

Den PIN-Code überlisten

01.09.2020 | News

Von: Felix Würsten

Will man an der Kasse grössere Beträge mit einer Kreditkarte bezahlen, muss man dies üblicherweise mit einem PIN-Code bestätigen. ETH-Forscher haben nun entdeckt, dass sich bei einigen Kreditkarten das System überlisten lässt.

Zahlen ohne PIN – Forscher knacken Visas NFC-Bezahlfunktion

Kontaktlos und ohne PIN bezahlten Forscher mit einer Visa-Karte quasi beliebig teure Produkte.

Lesezeit: **2 Min.** In Pocket speic

Security flaw allows bypassing PIN verification on Visa contactless payments

Experts demonstrate the PIN is useless in EMV contactless transactions

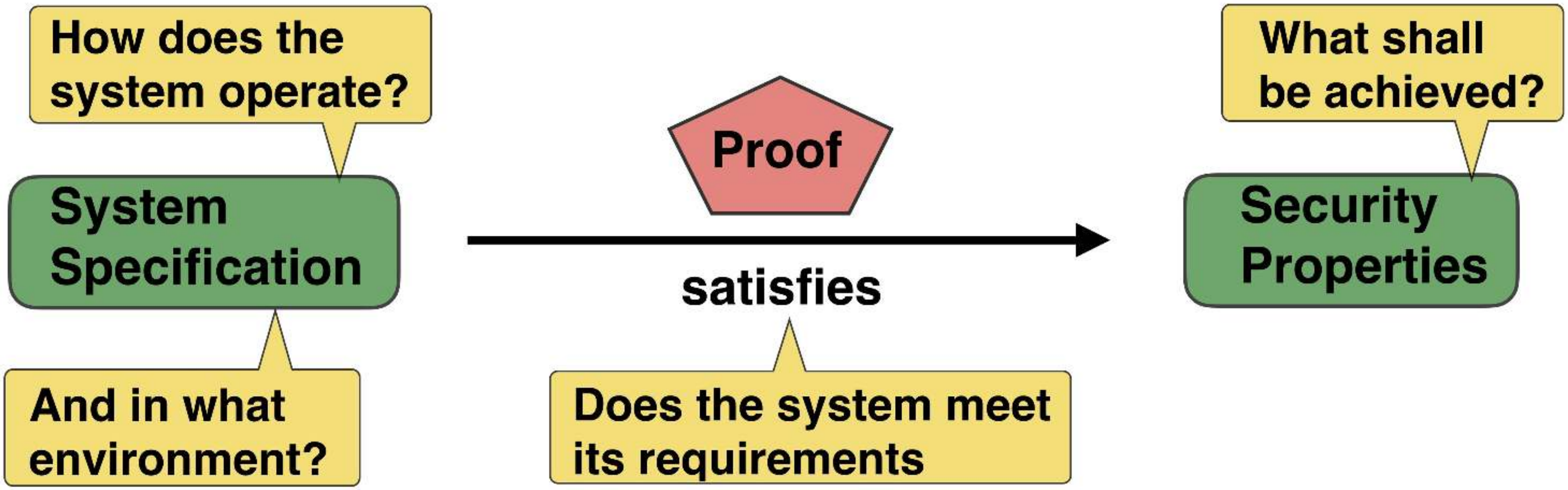
August 29, 2020 By Pierluigi Paganini

Researchers with ETH Zurich have identified vulnerabilities in the implementation of the payment card EMV standard that can allow bypassing PIN verification

Researchers David Basin, Ralf Sasse, and Jorge Toro-Pozo from the department of computer science at ETH Zurich discovered multiple vulnerabilities in the implementation of the payment card EMV standard that allow hackers to carry out attacks targeting both the cardholder and the merchant.

Where is the difficulty?

Where is the difficulty?



- Design documents are incomplete and imprecise
- Unclear adversary model

- Undecidability
- Even restricted cases intractable

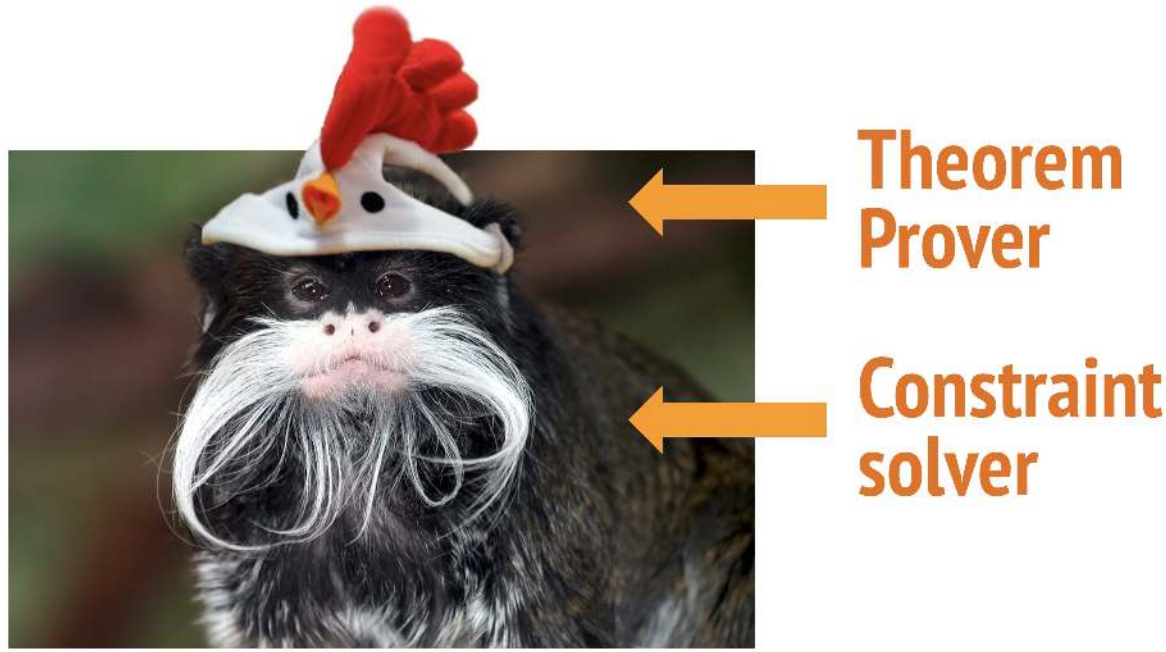
- Properties implicit or imprecise. E.g. "authenticate"

Weapon of choice

Constraint solver

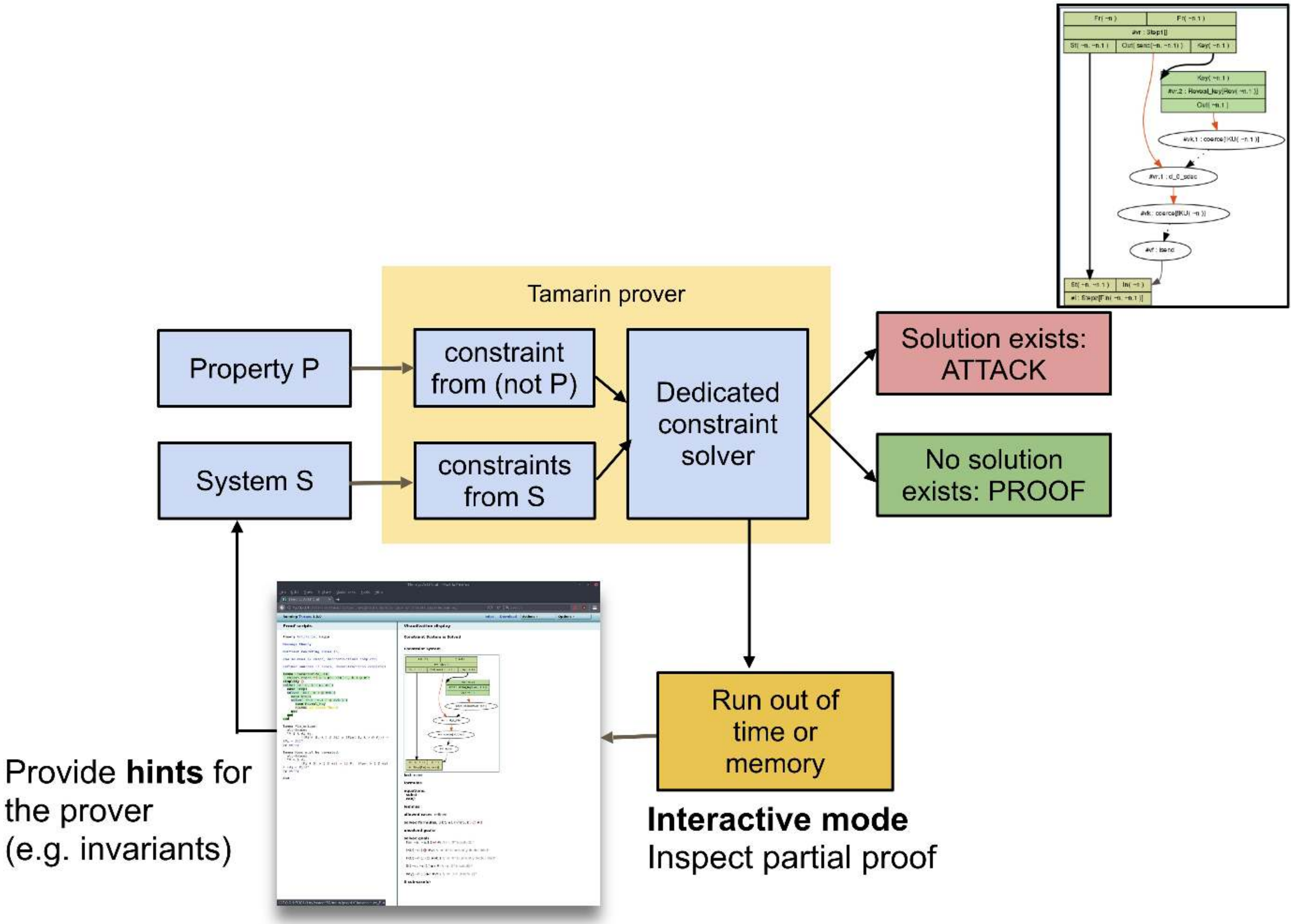
Tamarin prover

Weapon of choice



Tamarin prover

Tamarin Prover



What can Tamarin do for you?

- Rapid prototyping
- Finding attacks before you start a proof effort
- Provide a symbolic proof
- Explore alternative designs/threat models quickly

Contributors (partial)

ETH zürich

UNIVERSITY OF OXFORD

Inria

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Simon Meier

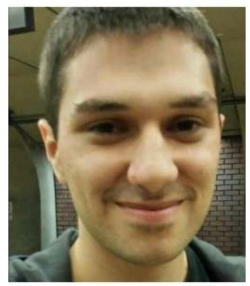
Benedikt Schmidt

Cas Cremers

David Basin

Robert Kunneman

Steve Kremer



Cedric Staub

Jannik Dreier

Ralf Sasse

Sasa Radomirovic

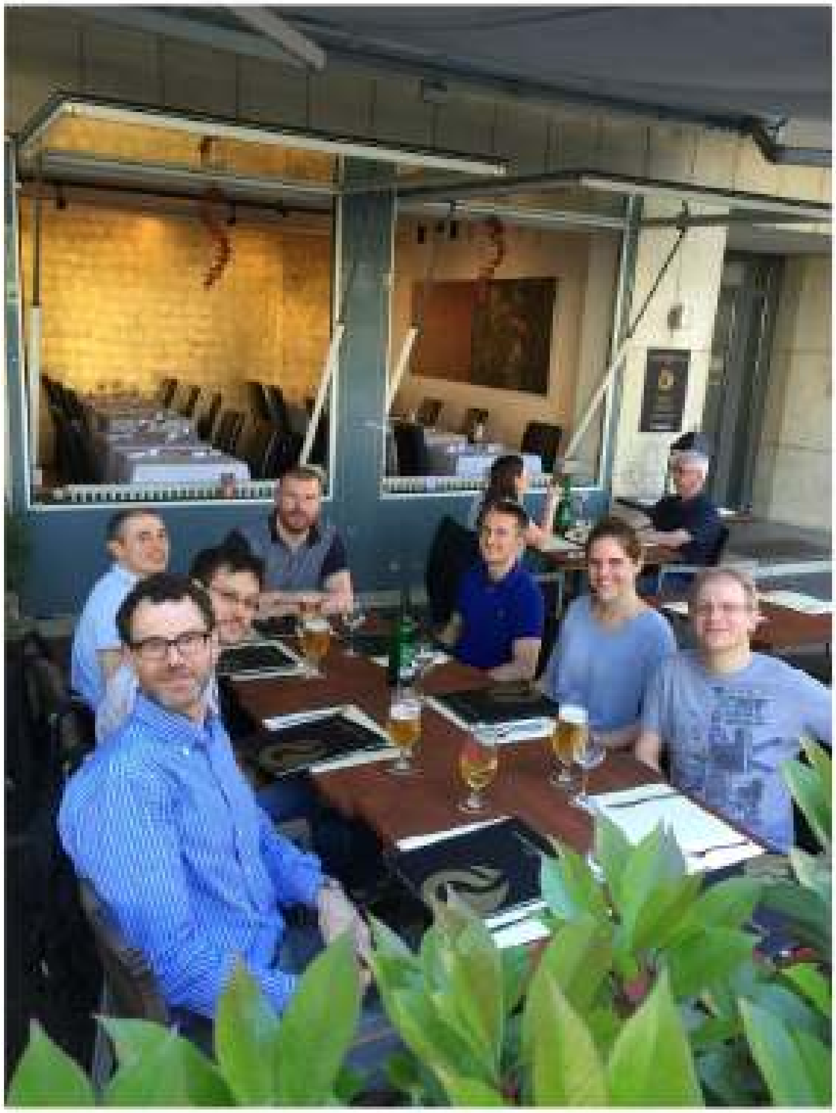
Lara Schmid

Charles Dumenil

Kevin Milner

Lucca Hirschi

Resources and documentation



- Sources on github
- 100+ page manual
- Plenty of examples/case studies
- Algorithm details in theses, papers
- We're writing a book!

Case Studies (examples)

Selected case studies

- Key exchange protocols
  - Naxos, Signed DH, KEA+, UM, Tsx
- Group protocols
  - GDH, TAK, (Sig)Joux, STR
- Identity-based KE
  - RYY, Scott, Chen-Kudla
- Loops
  - TESLA1 & 2
- Non-monotonic global state
  - Keyserver, Envelope, Exclusive secrets, Contract signing, Security device
- PKI and friends
  - ARPKI, DECIM
- E-Voting
  - Alethea, Selene, bulletin boards
- Detailed cryptographic primitives
  - WS-Security, X509, Scuttlebut, Let's Encrypt ACME, Bluetooth KE, Tendermint
- More complex analyses:
  - TLS 1.3
  - EMV (Chip and pin)
  - 5G-AKA, 5G handover
  - 802.11 WPA2 (Wifi)
  - TPM 2.0 direct anonymous attestation
  - DNP3 SAv5 (power grid)
  - Noise protocols
  - YubiKey/YubiHSM

Security protocols

- A **protocol** consists of rules describing how messages are exchanged between principals.

$$
\begin{array}{lll}
1. & A \to B: & \{A, N_A\}_{K_B} \\
2. & B \to A: & \{N_A, N_B\}_{K_A} \\
3. & A \to B: & \{N_B\}_{K_B}
\end{array}
$$

  I.e. a **distributed algorithm** with emphasis on communication.

- A **security** (or **cryptographic**) protocol uses cryptographic mechanisms to achieve security objectives.

- In practice, descriptions combine prose, data types, diagrams, ad hoc notation, and message sequences as above.

Message constructors (sample)

Names: A, B or *Alice*, *Bob*, ...

Asymmetric keys: A's public key K_A and private key K_A^{-1}.

Symmetric keys: K_{AB} shared by A and B.

Encryption: asymmetric $\{M\}_{K_A}$ and symmetric $\{M\}_{K_{AB}}$.

Signing: $\{M\}_{K_A^{-1}}$.

Nonces: N_A. Fresh data items used for challenge/response.

Timestamps: T. Denote time, e.g., used for key expiration.

Message concatenation: M_1, M_2. (Or $M_1||M_2$)

Example: $\{A, T_A, K_{AB}\}_{K_B}$.

Communication

- Fundamental notion: communication between principals (agents).

$$A \rightarrow B : \{A, T_A, K_{AB}\}_{K_B}$$

- A and B name **roles**.

  Can be instantiated by any principal playing the role.

- Communication usually modeled as being asynchronous.

$$A \rightarrow \ : \{A, T_A, K_{AB}\}_{K_B}$$
$$\rightarrow B : \{A, T_A, K_{AB}\}_{K_B}$$

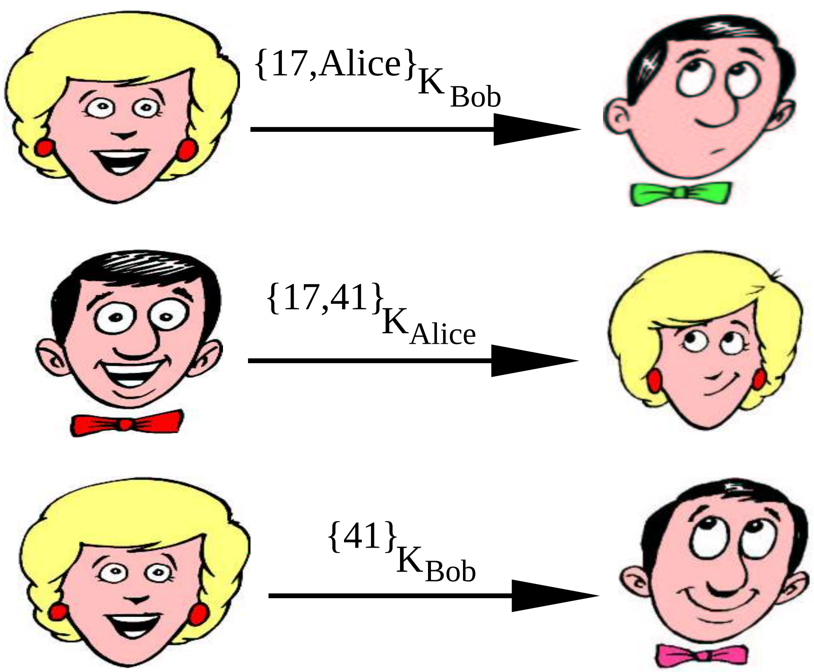
- Protocol specifies actions of principals in different protocol roles.

  It thereby also defines a set of event sequences (traces).

An authentication protocol (NSPK)

$$
\begin{aligned}
&1.\quad A \rightarrow B: \ \{A, N_A\}_{K_B}\\
&2.\quad B \rightarrow A: \ \{N_A, N_B\}_{K_A}\\
&3.\quad A \rightarrow B: \ \{N_B\}_{K_B}
\end{aligned}
$$

Here is an instance (a protocol run):



Execution in presence of attacker

Aliases: intruder, adversary, spy, Mallory, ...

How do we model the attacker? Possibilities:

- He knows the protocol but **cannot break crypto**. (Standard)

  Separates concerns: attacks on crypto versus communication.

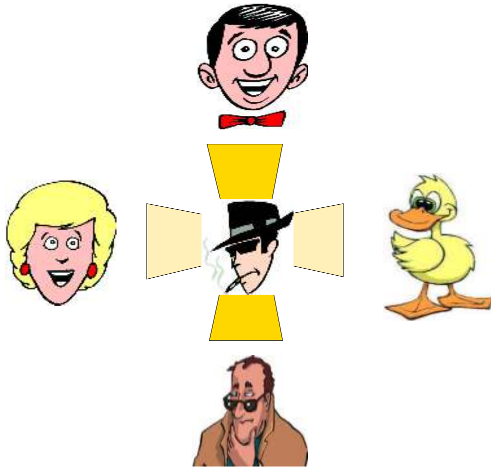
- He is **passive** but overhears all communications.

- He is **active** and can intercept and generate messages.

  "Transfer 20 CHF to Alice" $\leadsto$ "Transfer 10,000 CHF to Bob"

- He can compromise parties running the protocol, or perhaps learn some of their secrets (like their long-term keys).

Standard symbolic attacker model (Dolev-Yao)



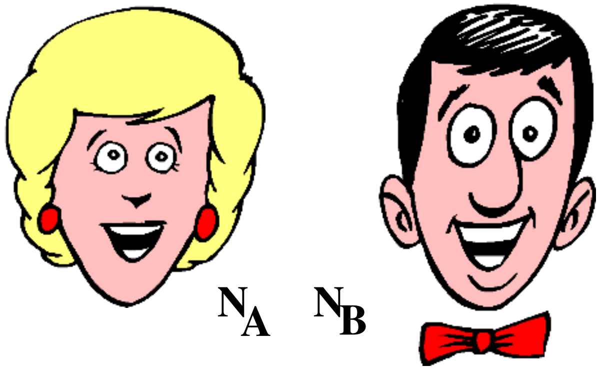
- An active attacker who controls the network.
  - He can **intercept** and **read** all messages.
  - He can **decompose** messages into their parts.
    But cryptography is "perfect": decryption requires inverse keys.
  - He can **construct** and **send** new messages, any time.
  - He can even **compromise** some agents and learn their keys.

- A protocol should ensure that communication between **non-compromised** agents achieves objectives (next slide).

- Strong attacker $\Longrightarrow$ protocols work in many environments.

  Note: symbolic model idealizes cryptographic model based on bit-strings and probabilistic polynomial-time attackers.

Example: NSPK



$$1. \quad A \to B: \quad \{A, N_A\}_{K_B}$$
$$2. \quad B \to A: \quad \{N_A, N_B\}_{K_A}$$
$$3. \quad A \to B: \quad \{N_B\}_{K_B}$$

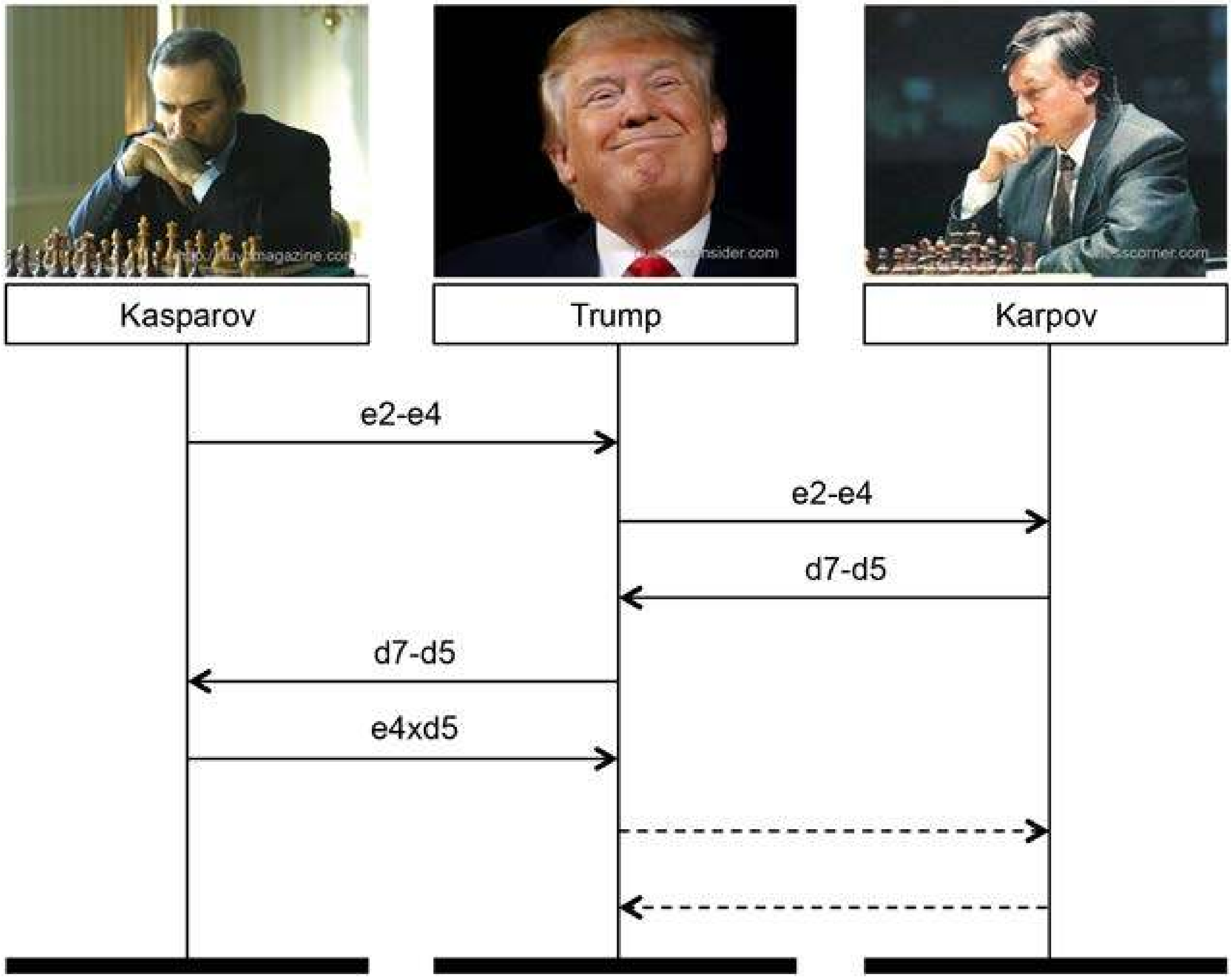
- **Objective:** Upon completion, A and B have been running the protocols in the right role and possess the same nonces, which are shared secrets between them, i.e., not known to the attacker.
  (We see later how to state this formally.)

- Correctness argument (informal).

1. This is Alice and I have chosen a nonce N_{Alice}.
2. Here is your Nonce N_{Alice}. Since I could read it, I must be Bob. I also have a challenge N_{Bob} for you.
3. You sent me N_{Bob}. Since only Alice can read this and send it back, you must be Alice.

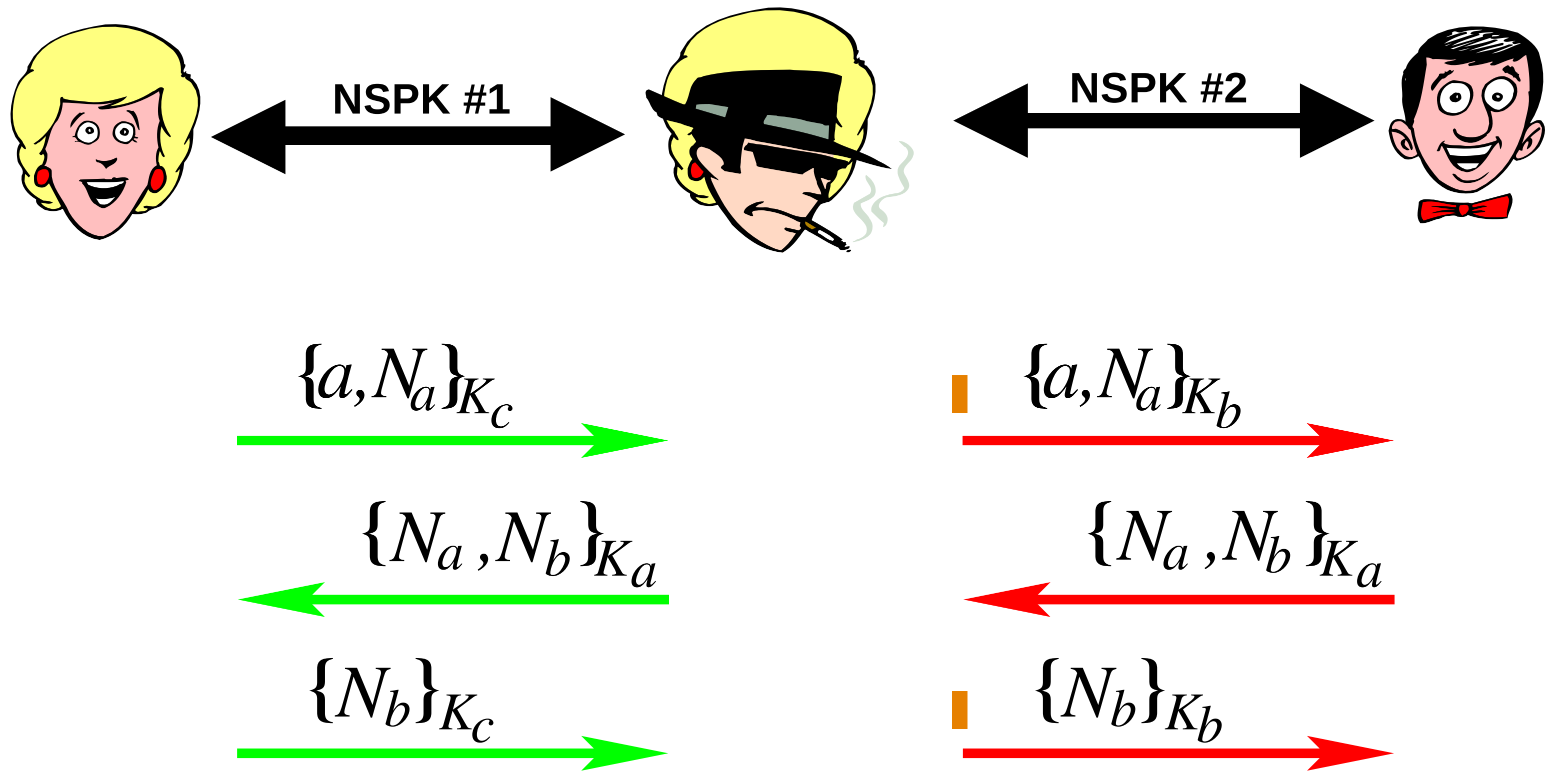
Protocol proposed in 1970s and used for decades.

Even Trump can beat a grandmaster



Attack on NSPK

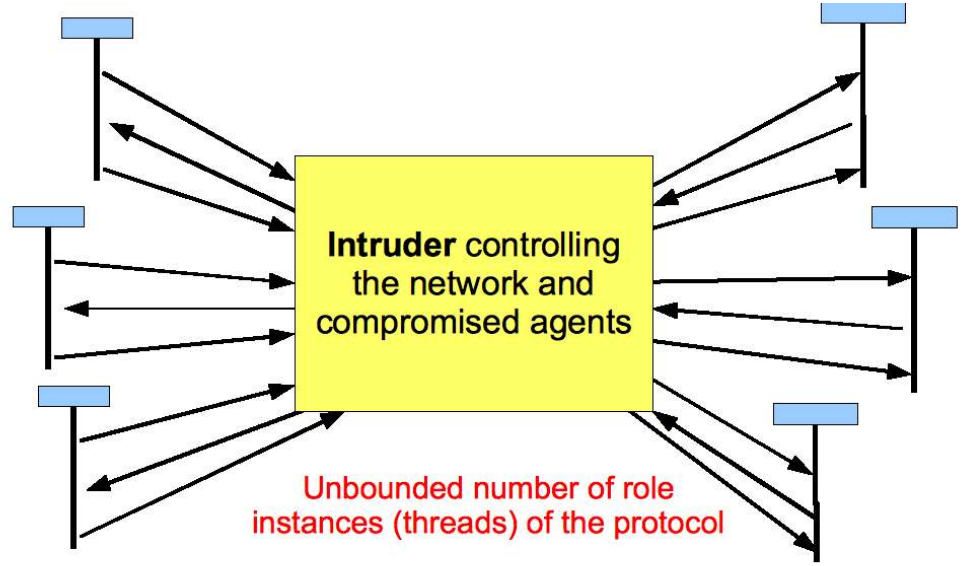
$$\begin{array}{lll} 1. & A \rightarrow B: & \{A, N_A\}_{K_B} \\ 2. & B \rightarrow A: & \{N_A, N_B\}_{K_A} \\ 3. & A \rightarrow B: & \{N_B\}_{K_B} \end{array}$$



$b(ob)$ believes he is speaking with $a(lice)$!

How might you protect against this attack?

Why are such attacks so difficult to spot?



(It took 20 years to find attack.)

- Assumptions are unclear.

  Is the intruder an insider or an outsider?

- Complex underlying model despite the suggestion of simplicity.

- Humans poor at envisioning all possible interleaved computations.

- And real protocols are **much** more complex!

We humans need help in modeling and reasoning about protocols and their properties.