CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Formal Analysis of Real-World Security Protocols

Lecture 0: Organization and Motivation

Aleksi Peltonen (aleksi.peltonen@cispa.de) | 21.10.2024

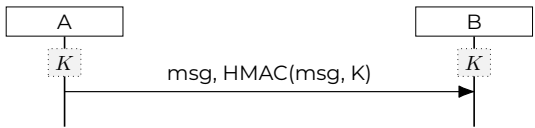
What is a Protocol?

What is a protocol?

- A **protocol** is a set of rules that determine how two or more parties can achieve a common goal by exchanging messages
- Protocols that use *cryptography* to achieve security-related goals are called **security** or **cryptographic** protocols

What is a protocol?

- A **protocol** is a set of rules that determine how two or more parties can achieve a common goal by exchanging messages
- Protocols that use *cryptography* to achieve security-related goals are called **security** or **cryptographic** protocols



Why should we care about protocol security?

New things are constantly connected to the Internet - security **threats** are everywhere

Nokia Threat Intelligence Report finds malicious IoT botnet activity has sharply increased[1]

Number of IoT devices (bots) engaged in botnet-driven DDoS attacks rose from around 200,000 a year ago to approximately 1 million devices.

Researchers find 36 new security flaws in LTE protocol[2]

South Korean researchers apply fuzzing techniques to LTE protocol and find 51 vulnerabilities, of which 36 were new.

The tech flaw that lets hackers control surveillance cameras[3]

Hackers Can Clone Millions of Toyota, Hyundai, and Kia Keys[4]

Encryption flaws in a common anti-theft feature expose vehicles from major manufacturers.

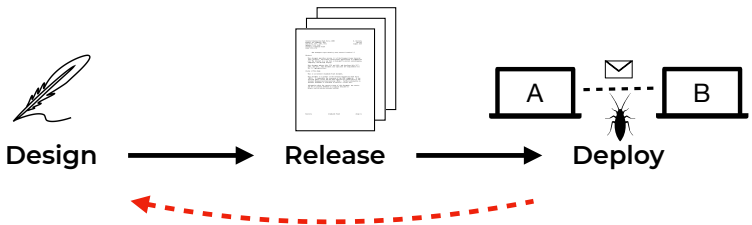
Massive HTTP DDoS Attack Hits Record High of 71 Million Requests/Second[5]

[1] https://www.nokia.com/about-us/news/releases/2023/06/07/nokia-threat-intelligence-report-finds-malicious-iot-botnet-activity-has-sharply-increased/
[2] https://www.zdnet.com/article/researchers-find-36-new-security-flaws-in-lte-protocol/
[3] https://www.bbc.com/news/technology-65975446
[4] https://www.wired.com/story/hackers-can-clone-millions-of-toyota-hyundai-kia-keys/
[5] https://thehackernews.com/2023/07/massive-http-ddos-attack-hits-record.html

How do we know a protocol is secure?

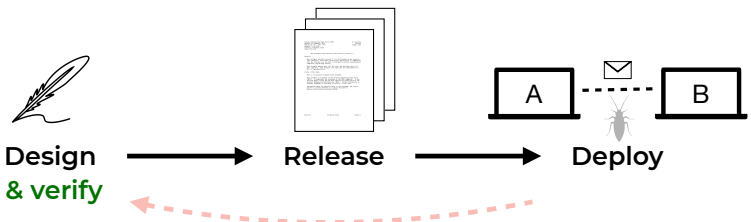
Before 1970

- **No proofs**
- Trial and error – we're not quite sure what a clever attacker could do
- Security: break - patch - repeat



1970-1980

- **Let's prove security** (Goldwasser, Micali, Yao)
- Attacker is any polynomial-time Turing machine
- Reduce security problem to generically solving a hard problem (e.g., discrete logarithm, factoring)



1980-1990

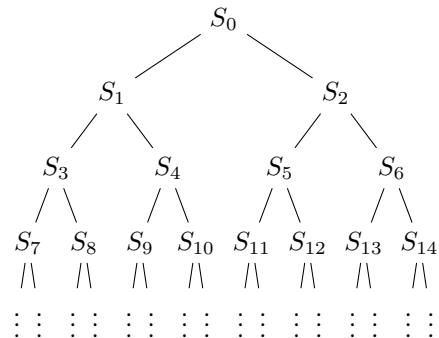
- **Computational proofs**
  - Proofs scale from primitives (“a signature scheme”) to small protocols (“a simple key exchange protocol”)
  - Typically pen-and-paper proofs, very error prone
  - *Not the focus of this lecture*
- **Symbolic proofs**
  - 1983: symbolic attacker model (Dolev-Yao)
  - Reason about abstract terms instead of bitstrings
  - “Perfect cryptography” assumption: Attacker can only decrypt an encrypted message if they know the key
  - Example property: Show that attacker can (not) learn a secret term

1990-2000

- **Model checkers become more widespread**
  - Analyze small finite cases
  - Found many flaws on basic designs
    - 1995: MitM attack on the NSPK protocol discovered with automated verification
- **Proliferation of tools & methods**
  - CSP/FDR, Prolog, NRL Protocol Analyzer, . . .
  - BAN Logic and variants
  - Isabelle (Paulson)

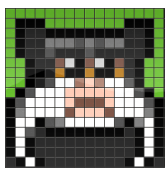
2000-2010

- **Bounded analysis matures**
  - AVISPA/Avantssar (OFMC, SAT-MC, CL-Atse)
  - Hit scaling barrier: State space explosion
- **Unbounded analysis?**
  - Undecidable.. Yet workable in practice?
  - ProVerif, Maude-NPA, Scyther, CPSA, Athena



2010-

- Tool situation stabilizes
  - ProVerif
  - **Tamarin prover**
- Increased expressiveness
  - Equational theories
  - Loops / stateful protocols
- **Real-world analysis**
  - Scaling to real-world protocols
  - Stronger threat models
  - Visibility to security engineers and standards bodies



Course Overview

This lecture

- What is formal analysis of security protocols?
- Learn about one state-of-the-art tool: the **Tamarin prover**
  - Learn how Tamarin works
  - Learn how to use Tamarin
  - Analyze protocols!
- Understand the **guarantees** and **limitations** of such analyses
- How have they been applied to **real-world protocols**?
- What **research** still needs to be done to make protocols secure?

Course staff

Cas Cremers

Aleksi Peltonen

Alexander Dax

Niklas Medinger

Course organization

Format

- Weekly lectures (Monday, 14-16)
  - 12-13 lectures in total
- Weekly exercises
  - 6 exercise sheets in total
- Midterm exam (2.12.204)
  - Covers lectures 0-5
  - Passing required to start the project

Grading

- Midterm exam (30%) + Project (70%)

IMPORTANT

- Make sure you are registered to the course
- Find a teammate to work with and register your group in CMS
- Regularly check CMS for news, updates, and other information

Tentative course plan

Lecture	Content	Other
0	Organization and Motivation	-
1	Terms and Equational Theories	Exercise 1
2	Protocols in the Symbolic Model	Exercise 2
3	Attacker Model and Trace Properties	Exercise 3
4	Verification Theory (Part 1)	Exercise 4
5	Verification Theory (Part 2)	-
	Midterm Exam (2.12.2024)	
6	Using Tamarin in Practice	Exercise 5
7	Advanced Security Properties and Threat Models	Exercise 6
8	Advanced Features (Part 1)	Project introduction
9	Advanced Features (Part 2)	-
10	Case Studies	-
11	Recent Research and Future Work	-
12	TBD	-

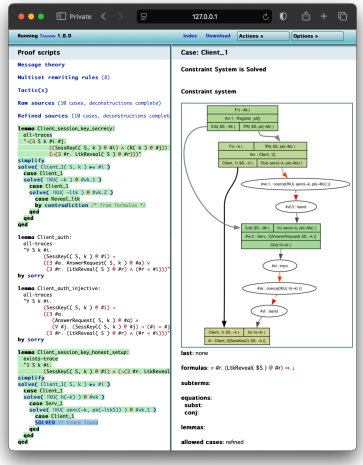
Course book

- David Basin, Cas Cremers, Jannik Dreier, and Ralf Sasse. **Modeling and Analyzing Security Protocols with Tamarin: A Comprehensive Guide**, 2024.
  - Unpublished; you can download the most recent draft on CMS
  - Each lecture will have a list of references to the relevant sections
  - Please note that the course will also cover topics that are not included in the book. **Attending the lectures is therefore highly recommended!**
- Other references are mentioned as optional reading for those who are interested

Analysis of Real-World Protocols

The Tamarin prover

- Symbolic analysis of security protocols
- First released in 2012, still under active development
- Simple protocols: Fully automatic
- Complex protocols: Interactive mode
- Considers an **unbounded number of sessions**, supports advanced features like stateful protocols and loops
- Open source on GitHub: `https://tamarin-prover.com/`



The Tamarin prover

Theorem prover

Constraint solver

Application domain examples

Key Exchange

- Naxos
- Signed DH
- Station-to-Station
- KEA+
- IKEv2
- Wireguard
- PQ-Wireguard
- Noise protocol family

Large Case Studies

- TLS 1.3
- IEEE 802.11 WPA2
- 5G-AKA
- 5G handover
- SPDM 1.2
- Apple iMessage PQ3

Payment

- EMV

Authentication

- WS-Security
- ACME

E-voting

- Alethea
- Belenios
- Selene

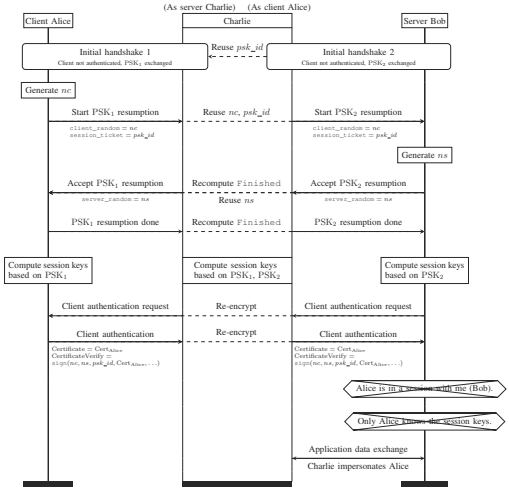
PKI

- ARPKI

Example 1 - TLS 1.3

- **What is it?**
  - **T**ransport **L**ayer **S**ecurity, version **1.3**
  - Likely the world's most used security protocol
- **Tamarin analysis**
  - Explicit support for the IETF during the development of TLS 1.3
  - Several person months of work: much of it in just understanding the standard
- **Results**
  - Tamarin finds complete break for the main proposal for Rev 10+'s "delayed authentication"
  - Minimal attack involves 3 modes & 18 messages
  - Motivated change to other mechanism
  - Proven core properties for final version

Example 1 - TLS 1.3



Re: [TLS] Revision 10: possible attack if client authentication is allowed during PSK
Eric Rescorla Sat, 31 October 2015 13:56 UTCShow header

Sam,

Thanks for posting this. It's great to see people doing real formal
analysis of the TLS 1.3 draft; this is really helpful in guiding the design.

As you say, the current draft-10 doesn't permit certificate-based client
authentication when PSK is used [0], but it's clear that we want to at least
be able to have this in the post-handshake case (since otherwise you
wouldn't be able to do on-the-fly client auth with a resumed handshake),
and this shows that we have to be very careful when we do that.
Note: It's less clear we want to allow this in the initial handshake, if
only
because it complicates the state machine.

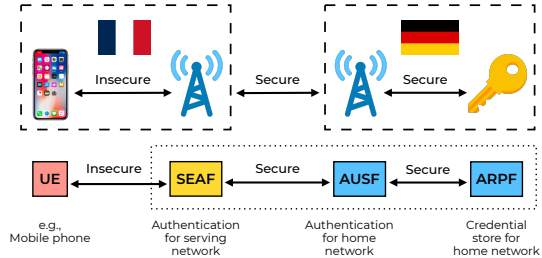
This result motivates and confirms the need to modify the handshake hashes
to contain the server Finished when we add post-handshake authentication
as is done in PR#316, which of course we'll be discussing in Yokohama.
I'd be very interested in learning of the results you get when you model
that.

Thanks again for your contribution here. It's really important to have
this kind of analysis, especially at this stage before the design is
completely
hardened.

Best,
-Ekr

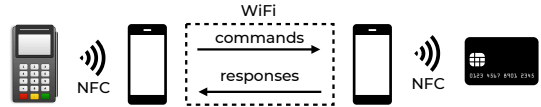
https://mailarchive.ietf.org/arch/msg/tls/TugB5ddJu3nYg7chcyeIyUqWSbA/

Example 2 - 5G-AKA



- **What is it?**
  - **5G A**uthentication and **K**ey **A**greement
  - The core key exchange of 5G
- **Tamarin analysis**
  - Analysis of the near-final standard (>1 000 pages)
  - Standard notably harder to parse than TLS 1.3's RFC, yet semi-stable
- **Results**
  - Tamarin finds that privacy is not guaranteed (linkable) – needs redesign
  - Tamarin proves other properties under some assumptions
  - (Prediction: many current wontfix'es will cause trouble later)

Example 3 - EMV



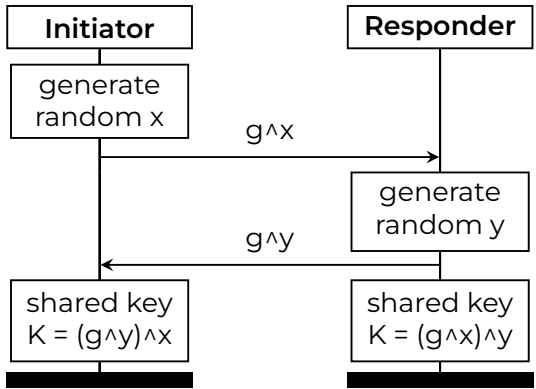
- **What is it?**
  - **E**uropay, **M**astercard, and **V**isa
  - Protocol between your bank, terminal, credit card
  - Modern versions have many different modes: Contactless, PIN, etc.
- **Tamarin analysis**
  - Standard documentation complex
  - Partial reverse engineering
- **Results**
  - Main attacks by researchers from ETH Zürich
    - PIN bypass: use man-in-the-middle to perform arbitrarily large transaction
    - Card mixup attack allow bypassing PIN codes
  - Later refined attacks by researchers from Birmingham and Surrey
    - Relay attacks and attacks on locked iPhones

Back to basics: Diffie-Hellman

Diffie-Hellman key exchange

One of the first proposals for secure protocols, for which we will see several variants.

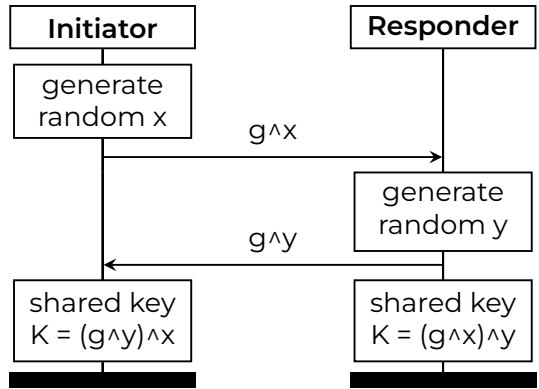
- “g^x” is shorthand for “gx mod P”, where g is the generator of a prime order group of order P. It is **efficient to compute g^x**
- If the group is sufficiently large, it is **infeasible to compute x from g^x**



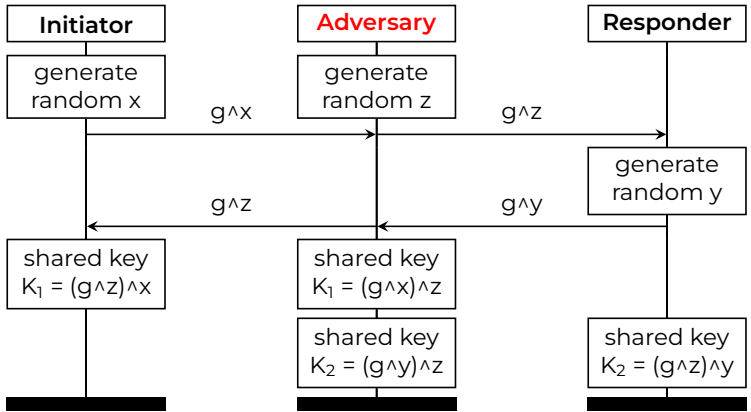
Diffie-Hellman key exchange

What should this protocol achieve?

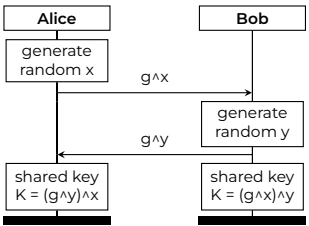
- If an adversary observes the network messages g^x and g^y, they cannot compute the shared key!
- What if the adversary can insert messages too?

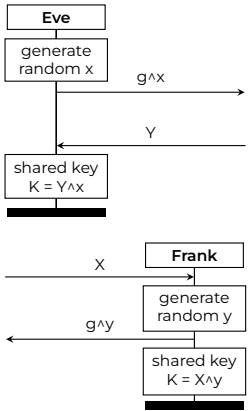


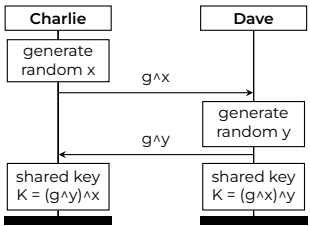
Man-in-the-middle attack



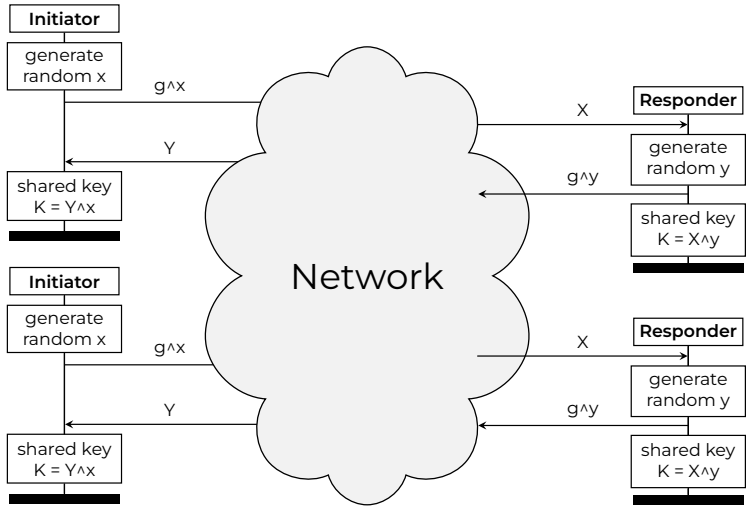
Execution model







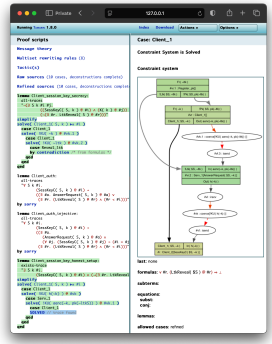
Adversary view



Summary

Next lecture

- Today, we learned that..
  - protocol security can be analyzed computationally or **symbolically**, and
  - we can automate the process using modern tools like **Tamarin**
- In the next lecture, we will learn more about modeling..
  - ..*messages* as **terms**
  - ..*cryptographic primitives* as **equations**



Reading material

Recommended reading: [Bas+24, Ch. 1-3]

[Bas+24] D. Basin, C. Cremers, J. Dreier, and R. Sasse. **Modeling and Analyzing Security Protocols with Tamarin: A Comprehensive Guide.** Draft v0.5. Sept. 2024.

Case studies i

Case studies:

TLS 1.3 [Cre+16; Cre+17],

5G-AKA [Bas+18; CD19],

EMV [BST21b; BST21a; Rad+22]

[Bas+18] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, and V. Stettler. **A Formal Analysis of 5G Authentication.** In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018.

Case studies ii

[BST21a] D. Basin, R. Sasse, and J. Toro-Pozo. **Card Brand Mixup Attack: Bypassing the PIN in non-Visa Cards by Using Them for Visa Transactions.** In: 30th USENIX Security Symposium (USENIX Security 21). 2021.

[BST21b] D. Basin, R. Sasse, and J. Toro-Pozo. **The EMV Standard: Break, Fix, Verify.** In: 2021 IEEE Symposium on Security and Privacy (SP). 2021.

[CD19] C. Cremers and M. Dehnel-Wild. **Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion.** In: 26th Annual Network and Distributed System Security Symposium, NDSS. 2019.

Case studies iii

[Cre+16] C. Cremers, M. Horvat, S. Scott, and T. van der Merwe. **Automated Analysis and Verification of TLS 1.3: 0-RTT, Resumption and Delayed Authentication.** In: 2016 IEEE Symposium on Security and Privacy (SP). 2016.

[Cre+17] C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe. **A Comprehensive Symbolic Analysis of TLS 1.3.** In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

[Rad+22] A.-I. Radu, T. Chothia, C. J. Newton, I. Boureanu, and L. Chen. **Practical EMV Relay Protection.** In: 2022 IEEE Symposium on Security and Privacy (SP). 2022.