A Rollercoaster Ride on the Formal Analysis of Attested TLS

Muhammad Usama Sardar[1], Arto Niemi[2], Hannes Tschofenig[3], Thomas Fossati[4]

[1]TU Dresden, Germany

[2]Huawei Technologies, Helsinki, Finland

[3]University of Applied Sciences Bonn-Rhein-Sieg and Siemens, Germany

[4]Linaro, Lausanne, Switzerland

March 28, 2024

Agenda

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
7. Formal Analysis of Attested TLS
8. Summary

Data-in-transit: Transport Protocols

- TLS[1]: world's most-used cryptographic protocol

[1] https://datatracker.ietf.org/doc/html/rfc8446

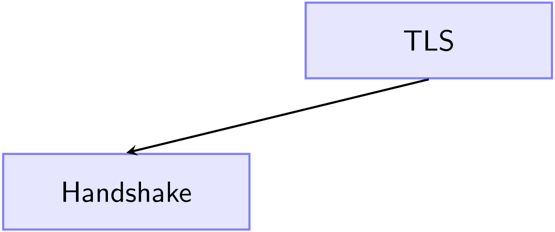
Data-in-transit: Transport Protocols

- TLS[1]: world's most-used cryptographic protocol
- Conceptually 2 main subprotocols:

[1] https://datatracker.ietf.org/doc/html/rfc8446

Data-in-transit: Transport Protocols

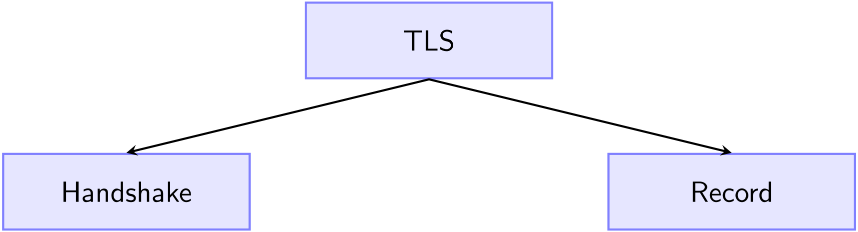
- TLS[1]: world's most-used cryptographic protocol
- Conceptually 2 main subprotocols:



[1] https://datatracker.ietf.org/doc/html/rfc8446

Data-in-transit: Transport Protocols

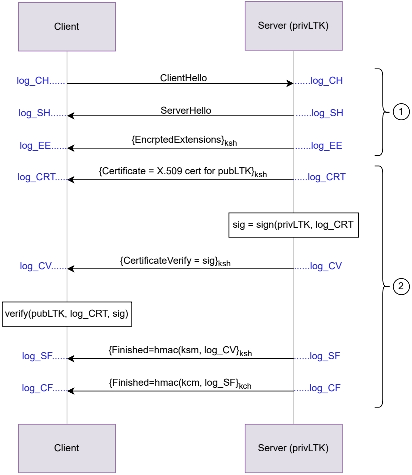
- TLS[1]: world's most-used cryptographic protocol
- Conceptually 2 main subprotocols:



[1] https://datatracker.ietf.org/doc/html/rfc8446

TLS Handshake Protocol

- Most complex part of TLS
  1. Unauthenticated key exchange (and parameter negotiation)
  2. Authentication (inc. key confirmation)



Problem in TLS

- No validation of security state of endpoint software and platform

Problem in TLS

- No validation of security state of endpoint software and platform
  - Need a more comprehensive set of security metrics in some use cases, e.g., CC

Problem in TLS

- No validation of security state of endpoint software and platform
  - Need a more comprehensive set of security metrics in some use cases, e.g., CC
- Very complex: at least 15 different exploits

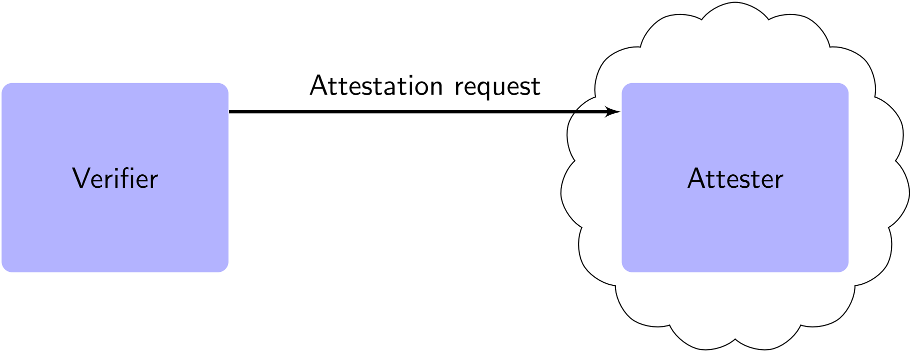
Problem in TLS

- No validation of security state of endpoint software and platform
  - Need a more comprehensive set of security metrics in some use cases, e.g., CC
- Very complex: at least 15 different exploits
  - Is all complexity (e.g., of key schedule) justified?

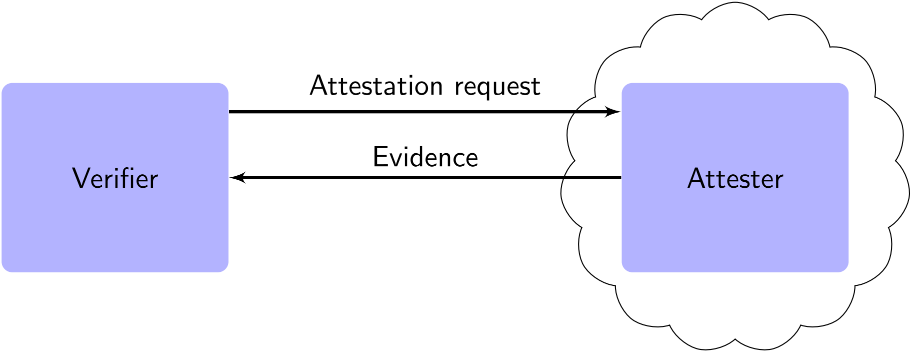
Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

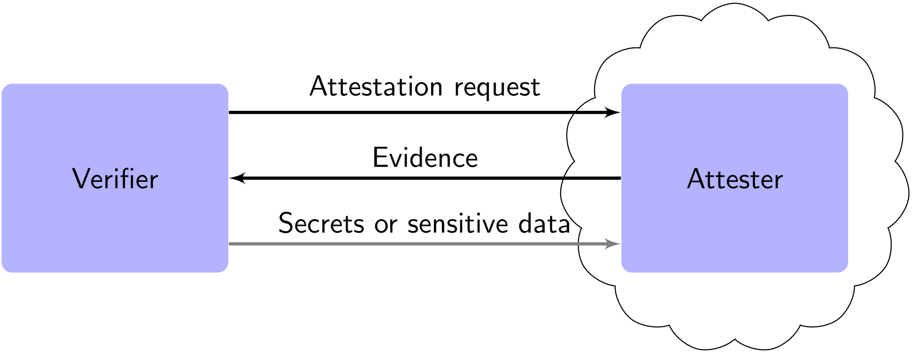
Architecturally-defined Attestation



Architecturally-defined Attestation



Architecturally-defined Attestation



Data-in-use: Architecturally-defined attestation[2]

- Intel TDX

	Integrity	Freshness	Confidentiality	Authentication
Intel's claimed TCB	×	×	×	×
Our proposed TCB	✓	✓	✓	×

- Arm CCA

Attester	Integrity	Freshness	Confidentiality	Authentication
Platform	✓	×	✓	×
Realm	✓	✓	✓	×

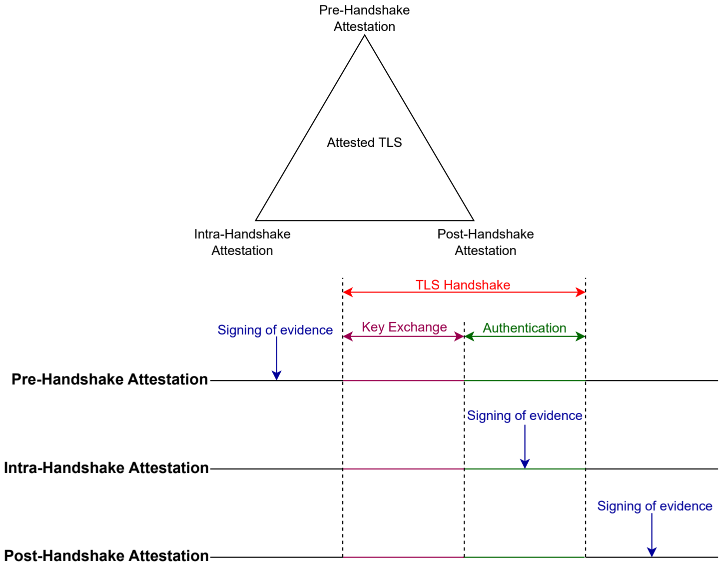
- Problem 1: No server authentication (e.g., misconfiguration)
- Problem 2: No standard way of implementation

[2] Sardar et al., *Formal Specification and Verification of Architecturally-defined Attestation Mechanisms in Arm CCA and Intel TDX*, 2023.

Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

Data-in-transit + Data-in-use



Intel's RA-TLS[3]

- Widely used protocol, e.g., in Gramine, RATS-TLS, Open Enclave Attested TLS, and SGX SDK Attested TLS

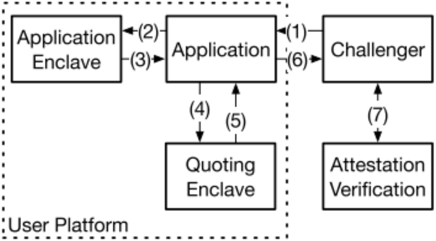


Figure 1: Remote Attestation Example. The challenger is off-platform with respect to the attester.

[3] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

Intel's RA-TLS[3]

- Widely used protocol, e.g., in Gramine, RATS-TLS, Open Enclave Attested TLS, and SGX SDK Attested TLS

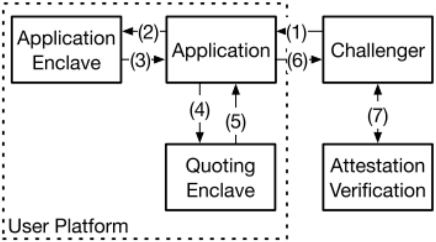


Figure 1: Remote Attestation Example. The challenger is off-platform with respect to the attester.

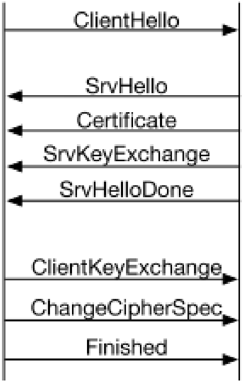


Figure 2: TLS 1.2 Handshake Messages.

[3]Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

Contributions

- Validation of formal model[4] of TLS 1.3 Key Schedule, revealing 3 major issues

[4] https://github.com/Inria-Prosecco/reftls

Contributions

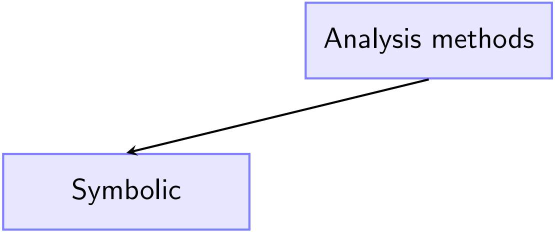
- Validation of formal model[4] of TLS 1.3 Key Schedule, revealing 3 major issues
- First formal analysis of attested TLS for TEEs

[4] https://github.com/Inria-Prosecco/reftls

Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

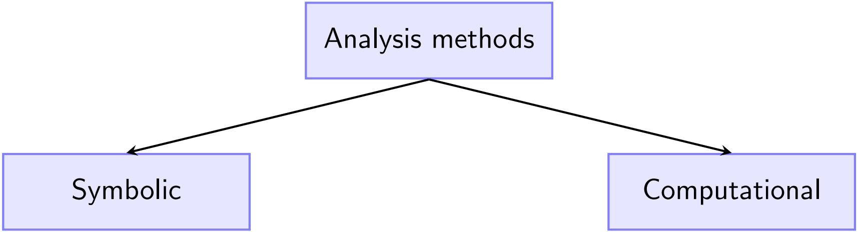
Analysis Approach[6] and Tool



[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

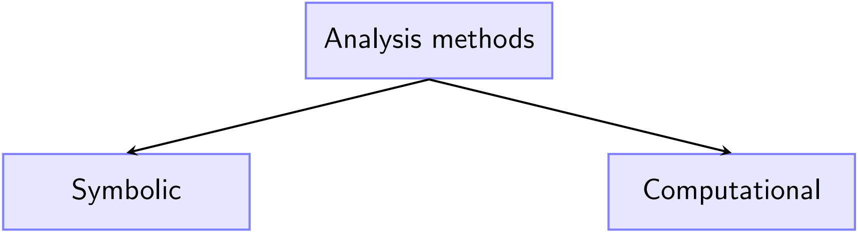


- Benefits of Symbolic

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

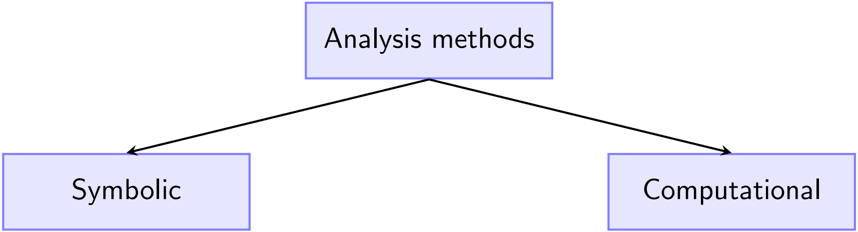


- Benefits of Symbolic
  - Reasonable level of abstraction

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

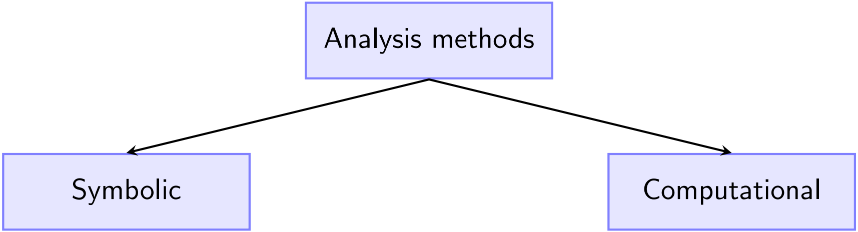


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

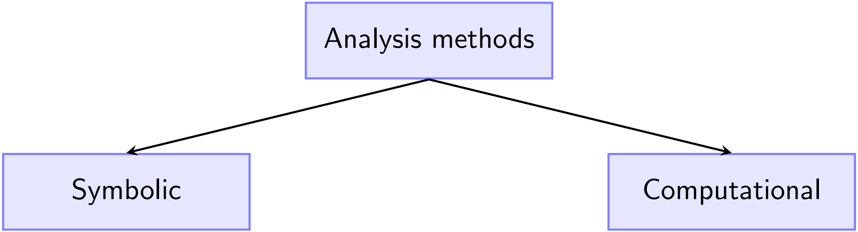


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

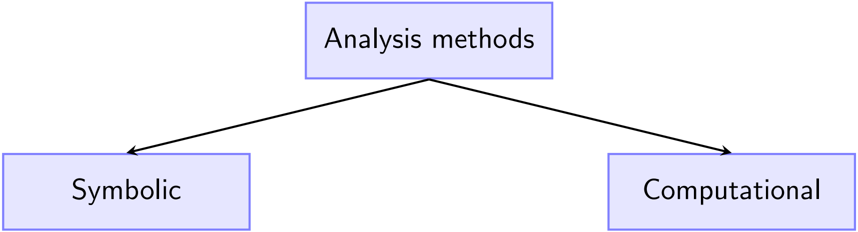


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic
  - Non-probabilistic model of cryptographic primitives

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

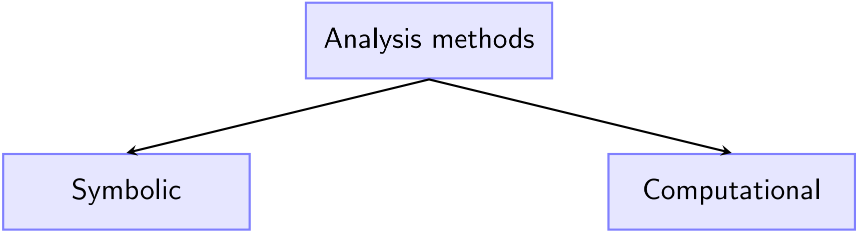


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic
  - Non-probabilistic model of cryptographic primitives
  - Side-channels out of scope

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

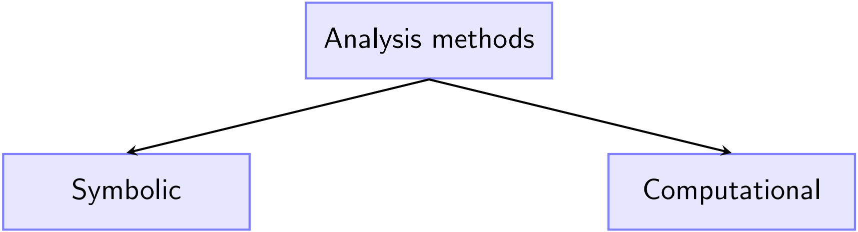


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic
  - Non-probabilistic model of cryptographic primitives
  - Side-channels out of scope
- Tool used: ProVerif[5]

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

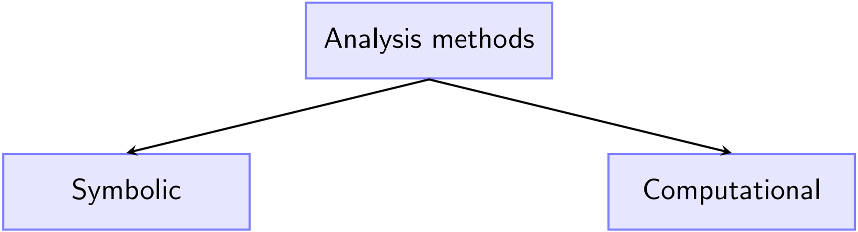


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic
  - Non-probabilistic model of cryptographic primitives
  - Side-channels out of scope
- Tool used: ProVerif[5]
  - Applied pi-calculus

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Analysis Approach[6] and Tool

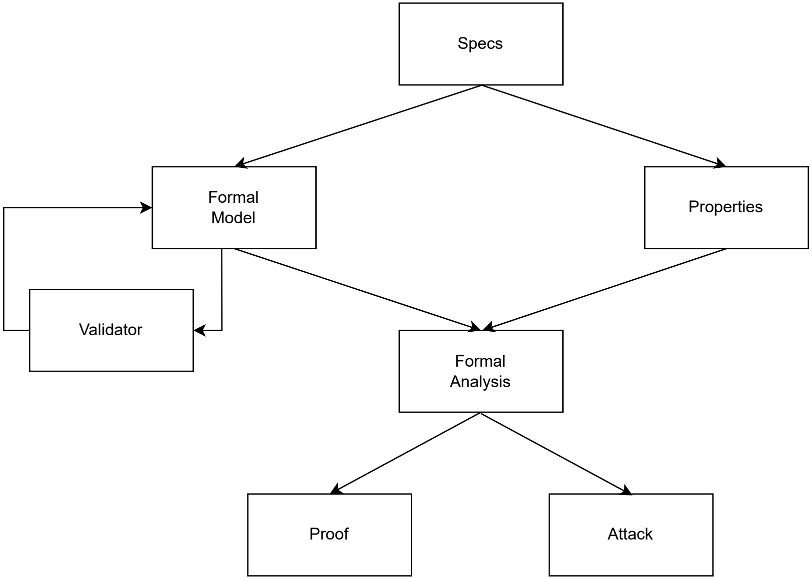


- Benefits of Symbolic
  - Reasonable level of abstraction
  - Unbounded number of sessions
- Limitation of Symbolic
  - Non-probabilistic model of cryptographic primitives
  - Side-channels out of scope
- Tool used: ProVerif[5]
  - Applied pi-calculus
  - Faster and extension to computational proofs (CryptoVerif)

[5] Blanchet, Cheval, and Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more", 2022.

[6] Barbosa et al., "SoK : Computer-Aided Cryptography", 2021.

Approach - Simplified



"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

“Rollercoaster”

- Incomplete and outdated specs for RA-TLS[7]
    - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
    - Fix: Used implementation and community input for formal model
- Very few comments in Inria’s TLS formal model[8]

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria's TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria's TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!
- Incomplete validation of draft 20 artifacts[9]

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria's TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!
- Incomplete validation of draft 20 artifacts[9]
  - Fix: Designed an automated validation framework for key schedule

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] https://github.com/Inria-Prosecco/reftls/tree/master/pv

[9] https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria's TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!
- Incomplete validation of draft 20 artifacts[9]
  - Fix: Designed an automated validation framework for key schedule
- A simple extension made the artifacts running for 1 month on high-end server (icelake)

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] https://github.com/Inria-Prosecco/reftls/tree/master/pv

[9] https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/

"Rollercoaster"

- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria's TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!
- Incomplete validation of draft 20 artifacts[9]
  - Fix: Designed an automated validation framework for key schedule
- A simple extension made the artifacts running for 1 month on high-end server (icelake)
  - Submitted to ProVerif developers for analysis

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] https://github.com/Inria-Prosecco/reftls/tree/master/pv

[9] https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/

“Rollercoaster”

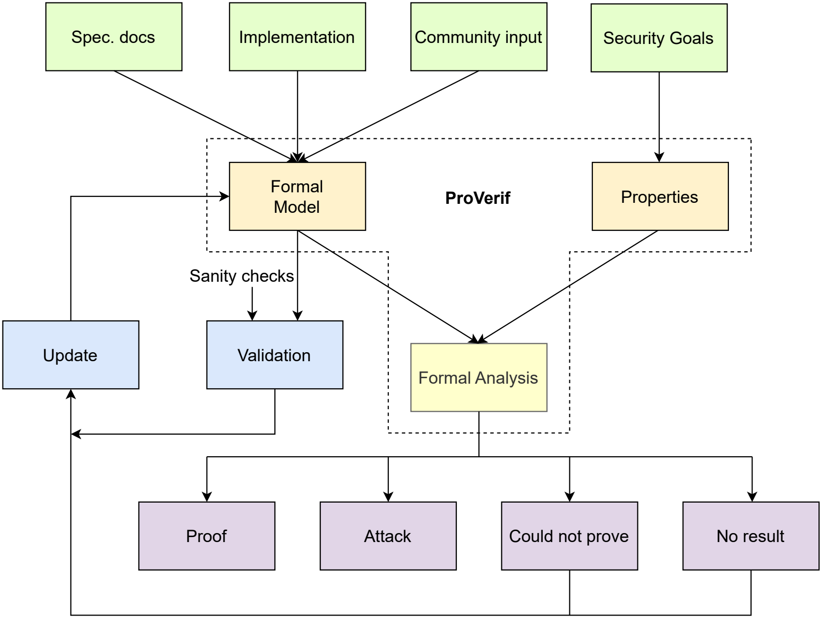
- Incomplete and outdated specs for RA-TLS[7]
  - Specs based on TLS 1.2 (TLS 1.3 is RFC since Aug 2018)
  - Fix: Used implementation and community input for formal model
- Very few comments in Inria’s TLS formal model[8]
  - Literally no comments at all in main processes (such as Client12, Server12, Client13, Server13, appData, channelBindingQuery and secrecyQuery)!
- Incomplete validation of draft 20 artifacts[9]
  - Fix: Designed an automated validation framework for key schedule
- A simple extension made the artifacts running for 1 month on high-end server (icelake)
  - Submitted to ProVerif developers for analysis
  - Fix: Formal model from scratch

[7] Knauth et al., *Integrating Remote Attestation with Transport Layer Security*, 2018.

[8] `https://github.com/Inria-Prosecco/reftls/tree/master/pv`

[9] `https://mailarchive.ietf.org/arch/msg/tls/-nFk9Eu7n-YFsFfGUe9X4JnrxX8/`

Approach



Spec. docs
Implementation
Community input
Security Goals
Formal Model
ProVerif
Properties
Sanity checks
Update
Validation
Formal Analysis
Proof
Attack
Could not prove
No result

Outline

1 TLS
2 Attestation (RA)
3 Attested TLS (RA+TLS)
4 Contributions
5 Approach and Tool
6 Validation of TLS 1.3
- Key Schedule
- Validation of Key Schedule

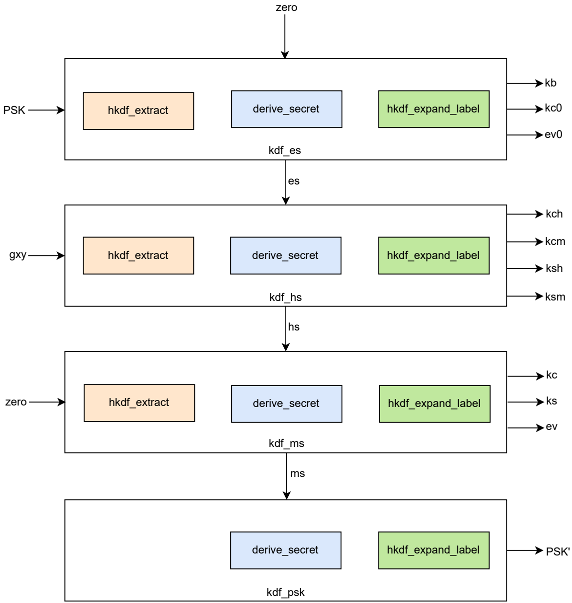
7 Formal Analysis of Attested TLS
- Flow
- Property

8 Summary

Agenda

6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule

Key Schedule - Overview

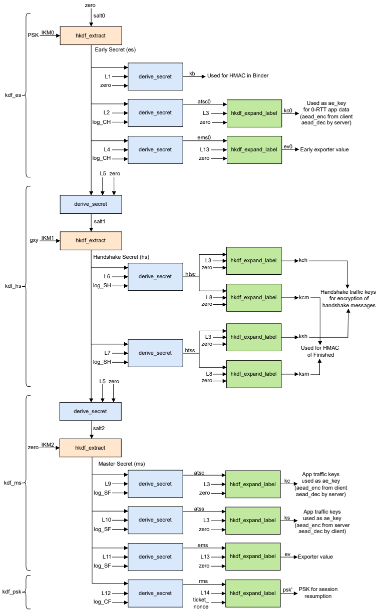


Key Schedule[10]

```
             0
             |
             v
   PSK ->  HKDF-Extract = Early Secret
             |
             +-----> Derive-Secret(., "ext binder" | "res binder", "")
             |                     = binder_key
             |
             +-----> Derive-Secret(., "c e traffic", ClientHello)
             |                     = client_early_traffic_secret
             |
             +-----> Derive-Secret(., "e exp master", ClientHello)
             |                     = early_exporter_master_secret
             v
       Derive-Secret(., "derived", "")
             |
             v
   (EC)DHE -> HKDF-Extract = Handshake Secret
             |
             +-----> Derive-Secret(., "c hs traffic",
             |                     ClientHello...ServerHello)
             |                     = client_handshake_traffic_secret
             |
             +-----> Derive-Secret(., "s hs traffic",
             |                     ClientHello...ServerHello)
             |                     = server_handshake_traffic_secret
             v
       Derive-Secret(., "derived", "")
             |
             v
   0 -> HKDF-Extract = Master Secret
             |
             +-----> Derive-Secret(., "c ap traffic",
             |                     ClientHello...server Finished)
             |                     = client_application_traffic_secret_0
             |
             +-----> Derive-Secret(., "s ap traffic",
             |                     ClientHello...server Finished)
             |                     = server_application_traffic_secret_0
             |
             +-----> Derive-Secret(., "exp master",
             |                     ClientHello...server Finished)
             |                     = exporter_master_secret
             |
             +-----> Derive-Secret(., "res master",
                                   ClientHello...client Finished)
                                   = resumption_master_secret
```

[10] https://datatracker.ietf.org/doc/html/rfc8446#section-7.1

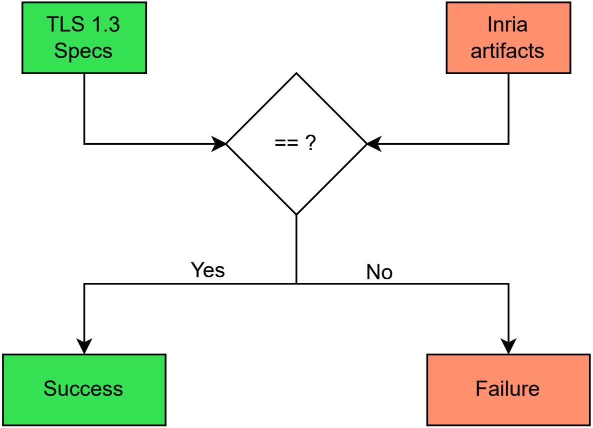
Key Schedule with 2nd Stage



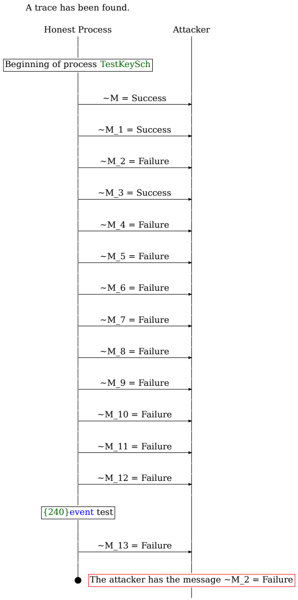
Agenda

6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule

Validation Framework



Validation Result



Issue 1: Salt for Handshake Secret[11]

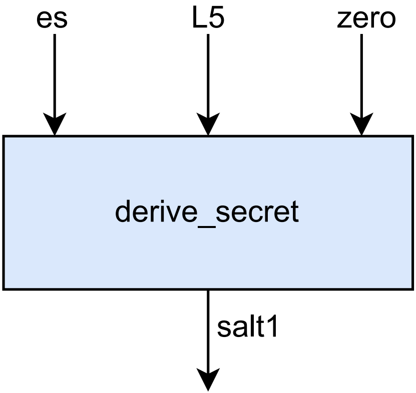


Figure: TLS 1.3 Specs

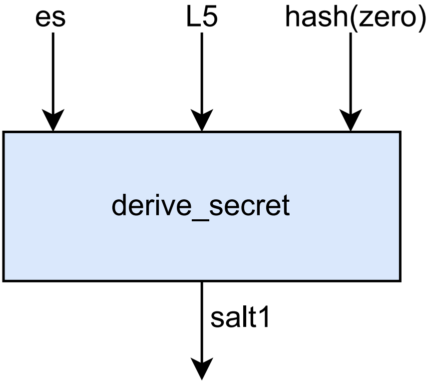


Figure: Inria artifacts

[11] https://github.com/Inria-Prosecco/reftls/issues/7

Issue 2: Salt for Master Secret[12]

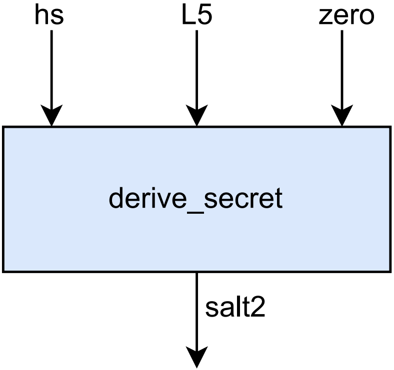


Figure: TLS 1.3 Specs

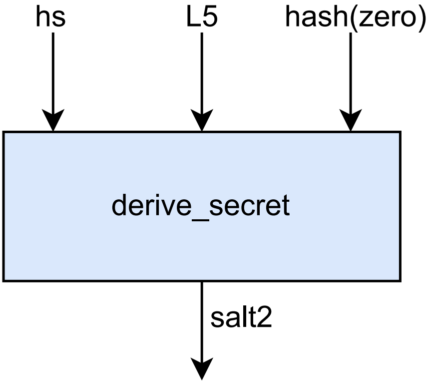


Figure: Inria artifacts

[12] https://github.com/Inria-Prosecco/reftls/issues/7

Issue 3: Master Secret[13]

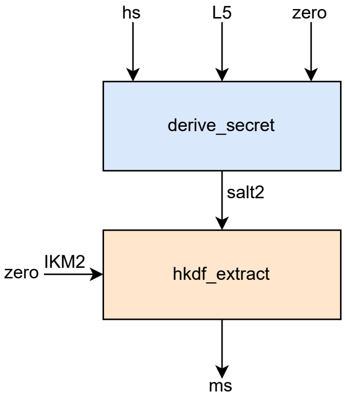


Figure: TLS 1.3 Specs

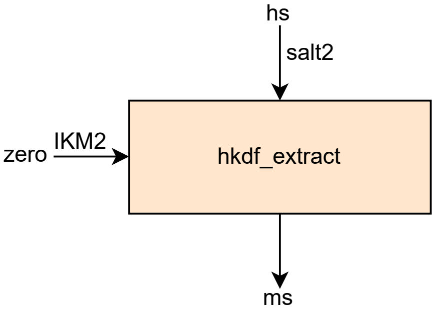


Figure: Inria artifacts

[13] https://github.com/Inria-Prosecco/reftls/issues/6

Ruling out Abstractions

- Ubuntu 20.04 LTS on an Intel Core i7-11800H processor with 64 GB of RAM

Code	**ProVerif 2.04**	**ProVerif 2.05**
Original	6 min 06.634 s	6 min 02.256 s
With issue 1 fixed	5 min 51.682 s	6 min 03.335 s
With issue 2 fixed	7 min 04.472 s	6 min 14.954 s
With issue 3 fixed	7 min 11.434 s	6 min 41.872 s
With all 3 issues fixed	6 min 40.010 s	6 min 31.887 s

Community input

- Paper authors[14]
  - Bruno Blanchet
  - Karthikeyan Bhargavan
  - Nadim Kobeissi
- LURK[15] authors
- IETF TLS WG[16]
- IRTF UFMRG chairs
- CCC attestation SIG[17]
- ...
- IETF 119 Hackathon[18]
- IRTF Crypto Forum RG @ IETF 119[19]

[14] Bhargavan, Blanchet, and Kobeissi, "Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate", 2017.

[15] https://github.com/lurk-t/proverif

[16] https://mailarchive.ietf.org/arch/msg/tls/ZGmyHwTYh2iPwPrirj_rkSTYhDo/

[17] https://github.com/CCC-Attestation/meetings/blob/main/materials/MuhammadUsamaSardar_Formal_RA-TLS.pdf

[18] https://wiki.ietf.org/meeting/119/hackathon

[19] https://datatracker.ietf.org/meeting/119/materials/slides-119-cfrg-formal-analysis-of-ra-tls-00

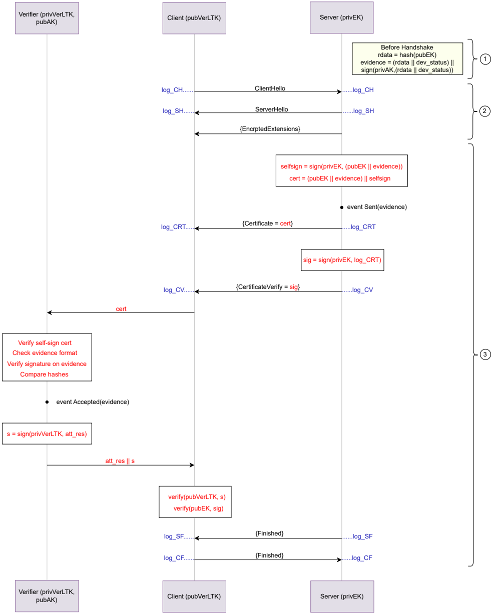
Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

Agenda

7. Formal Analysis of Attested TLS
   - Flow
   - Property

RA-TLS in BC Model



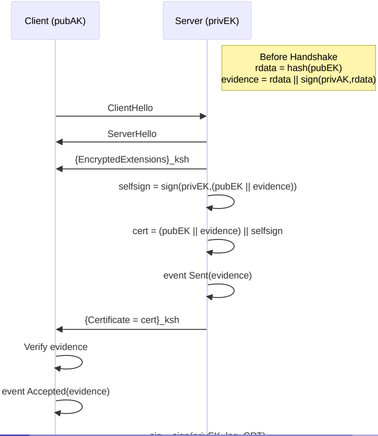
Agenda

7. Formal Analysis of Attested TLS
   - Flow
   - Property

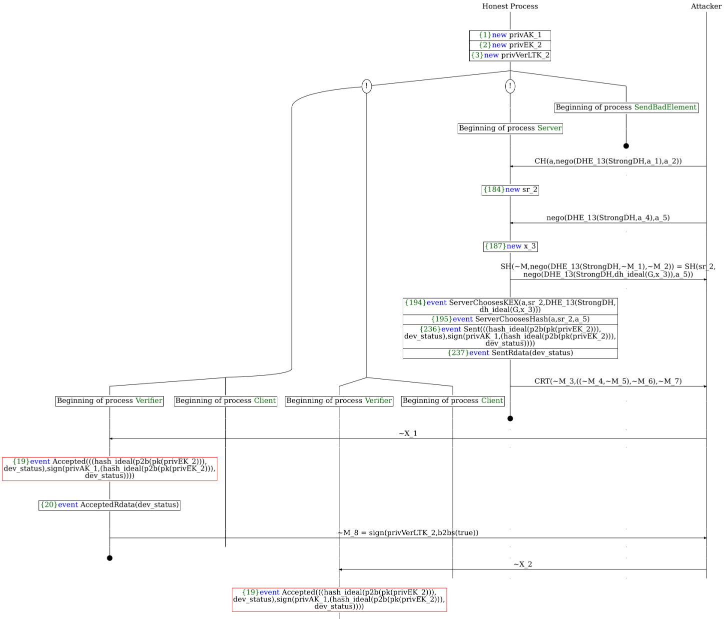
Replay Protection of Evidence

$$query\ ev : bitstring;$$

$$inj - event(Accepted(ev)) ==> inj - event(Sent(ev)) \quad (1)$$



Attack Trace (BC with one-way authenticated channel)



Outline

1. TLS
2. Attestation (RA)
3. Attested TLS (RA+TLS)
4. Contributions
5. Approach and Tool
6. Validation of TLS 1.3
   - Key Schedule
   - Validation of Key Schedule
7. Formal Analysis of Attested TLS
   - Flow
   - Property
8. Summary

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
    1. Comments in formal models (best practices)

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
    1. Comments in formal models (best practices)
    2. Validation of formal models
    3. Keep formal verification artifacts up to date (IRTF UFMRG)

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models
  3. Keep formal verification artifacts up to date (IRTF UFMRG)
  4. Usability of tools for formal analysis

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models
  3. Keep formal verification artifacts up to date (IRTF UFMRG)
  4. Usability of tools for formal analysis
- Plan

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
    1. Comments in formal models (best practices)
    2. Validation of formal models
    3. Keep formal verification artifacts up to date (IRTF UFMRG)
    4. Usability of tools for formal analysis
- Plan
    - Client-side attestation

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models
  3. Keep formal verification artifacts up to date (IRTF UFMRG)
  4. Usability of tools for formal analysis
- Plan
  - Client-side attestation
  - Propose and verify the fixed version for RA-TLS

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models
  3. Keep formal verification artifacts up to date (IRTF UFMRG)
  4. Usability of tools for formal analysis
- Plan
  - Client-side attestation
  - Propose and verify the fixed version for RA-TLS
- Call to action

Summary

- Intel's RA-TLS is potentially vulnerable to replay attacks
- Need for standardized and formally verified attested TLS
- Inria's formal model of TLS 1.3 draft-20 key schedule is wrong!
- Lessons learnt
  1. Comments in formal models (best practices)
  2. Validation of formal models
  3. Keep formal verification artifacts up to date (IRTF UFMRG)
  4. Usability of tools for formal analysis
- Plan
  - Client-side attestation
  - Propose and verify the fixed version for RA-TLS
- Call to action
  - anyone interested?

Key References

Barbosa, Manuel et al. "SoK : Computer-Aided Cryptography". In: *42nd IEEE Symposium on Security and Privacy*. 2021. URL: https://eprint.iacr.org/2019/1393.pdf.

Bhargavan, Karthikeyan, Bruno Blanchet, and Nadim Kobeissi. "Verified Models and Reference Implementations for the TLS 1.3 Standard Candidate". In: *2017 IEEE Symposium on Security and Privacy (SP)*. 2017, pp. 483–502. DOI: 10.1109/SP.2017.26.

Blanchet, Bruno, Vincent Cheval, and Véronique Cortier. "ProVerif with lemmas, induction, fast subsumption, and much more". In: *IEEE Symposium on Security and Privacy (S&P'22)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2022, pp. 205–222. DOI: 10.1109/SP46214.2022.00013.

Knauth, T. et al. *Integrating Remote Attestation with Transport Layer Security*. Tech. rep. Intel Labs, 2018. URL: https://arxiv.org/abs/1801.05863.

Sardar, Muhammad Usama et al. *Formal Specification and Verification of Architecturally-defined Attestation Mechanisms in Arm CCA and Intel TDX*. Nov. 2023. URL: https://www.researchgate.net/publication/375592777_Formal_Specification_and_Verification_of_Architecturally-defined_Attestation_Mechanisms_in_Arm_CCA_and_Intel_TDX.