Reactive and Proactive Standardisation of TLS

Thyla van der Merwe and Kenny Paterson

Real World Crypto
11 January 2018
Motivation

post-deployment-analysis vs analysis-prior-to-deployment

What can the security community learn? What standardisation model best fits critical protocols such as TLS?
NETSCAPE
SSL 2.0 (1995) → SSL 3.0 (1996)

IETF
**TLS 1.2**
- 2-RTT
- static RSA/DH
- HS not encrypted

**TLS 1.3**
- 1-RTT
- 0-RTT
- ephemeral DH
- HS encrypted

---

**IETF**

*No formal membership!

---

**Security Area**

*IP, TLS, HTTP,...*

---

**TLS Working Group (WG)**

**TLS Request for Comments (RFCs)**

---

**TLS Mailing List**
- Nov 26 2016
  - Re: [TLS] Certificate compression (a la QUIC) for TLS 1.3, Victor Vasiliev
  - Re: [TLS] Certificate compression (a la QUIC) for TLS 1.3, Eric Rescorla
  - [TLS] Certificate compression (a la QUIC) for TLS 1.3, Alessandro Gherardi

---

**Open model** for standards development
- no barriers to entry
- no financial barriers to adoption
TLS 1.2 and below - Design, Release, Break, Patch

- Development followed a **reactive** standardisation process
- An attack → releasing an extension OR making the change in the next version of the standard
Countermeasure to BEAST  
Long been known to be biased  
Deprecated in 2015 → phased out sooner?

Affects TLS 1.0 → exploits chained-IV vulnerability, recover plaintext  
Opened the floodgates → new techniques that made attack practical  
TLS 1.1 removes this vulnerability BUT implementations slow to react, TLS 1.0 is still popular today!

Affects SSL 3.0 → PKCS#1 v1.5 padding oracle uncovers the pre-master secret  
Briefly addressed in TLS 1.0 → mechanism to remove the padding oracle  
Re-enabled in many forms (Jager et. al, DROWN, ROBOT) → switch to PKCS#1 v2.1? No, backwards compatibility
<table>
<thead>
<tr>
<th>Attack</th>
<th>Damage</th>
<th>Fix</th>
<th>Resurrected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bleichenbacher</td>
<td>SSL 3.0, recover keys</td>
<td>Note in TLS 1.0 (1.1, 1.2)</td>
<td>Jager et al., DROWN, others</td>
</tr>
<tr>
<td>Vaudenay</td>
<td>TLS 1.0, recover plaintext</td>
<td>Addressed in TLS 1.1</td>
<td>Lucky Thirteen, POODLE (related)</td>
</tr>
<tr>
<td>Renegotiation</td>
<td>TLS 1.2 and below</td>
<td>Mandatory extension</td>
<td>Triple Handshake</td>
</tr>
<tr>
<td>BEAST</td>
<td>TLS 1.0, recovery plaintext</td>
<td>Addressed in TLS 1.1</td>
<td>Made practical with new techniques!</td>
</tr>
<tr>
<td>RC4</td>
<td>TLS 1.2 and below</td>
<td>Eventually deprecated</td>
<td>Old weakness</td>
</tr>
</tbody>
</table>
Contributing factors

- Backwards compatibility, wide deployment of TLS and time lags in adopting new versions hinder meaningful change
- Analysis tools not yet fully developed before TLS 1.2 release
- Lack of interaction with the academic community - reward came from producing high profile attacks
- Incentive model leaves users vulnerable to attack and imposes a patch action

Is a more cautious approach warranted for critical protocols?
TLS 1.3 - Design, Break, Fix, Release

- Development has followed a **proactive** standardisation process
- Working closely with the academic community, multiple drafts have been developed prior to official release
Academic community starts to get heavily involved!

**Removal of mechanisms** that aid attacks → compression, renegotiation, MEE

**Analysed** by Dowling et al. and Kohlweiss et al. → provides valuable feedback to WG on design
WG draws inspiration from secure designs and acknowledges the research community’s concerns!

Becomes highly influenced by OPTLS (Krawczyk and Wee)
Designed with TLS modes in mind
Uses secure primitives

Removal of mechanisms that aid attacks → compression, renegotiation, MEE
Analysed by Dowling et al. and Kohlweiss et al. → provides valuable feedback to WG on design
Analysed by Cremers et al. (automated, symbolic) and Li et al. Cremers et al. find a potential attack on post-handshake client authentication → informs fix for draft 11

Becomes highly influenced by OPTLS (Krawczyk and Wee)
Designed with TLS modes in mind
Uses secure primitives

Removal of mechanisms that aid attacks → compression, renegotiation, MEE
Analysed by Dowling et al. and Kohlweiss et al. → provides valuable feedback to WG on design
“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.”

Becomes highly influenced by OPTLS (Krawczyk and Wee)
Designed with TLS modes in mind
Uses secure primitives

Removal of mechanisms that aid attacks → compression, renegotiation, MEE
Analysed by Dowling et al. and Kohlweiss et al. → provides valuable feedback to WG on design
Analysed by Cremers et al. (automated, symbolic) and Li et al. Cremers et al. **find a potential attack** on post-handshake client authentication → informs fix for draft 11

Becomes highly influenced by **OPTLS** (Krawczyk and Wee)
- Designed with TLS modes in mind
- Uses secure primitives

**Removal of mechanisms** that aid attacks → compression, renegotiation, MEE

**Analysed** by Dowling et al. and Kohlweiss et al. → provides valuable feedback to WG on design

Much more analysis after **draft 10**...
“TLS Ready or Not?” (TRON 1.0) workshop in February 2016
  ○ showcased work by the academic community - computational analyses, symbolic analyses, implementations
  ○ brought the WG and the research community together
  ○ definition of properties - late in the game?
  ○ followed up by the less formal TRON 2.0

Huge amount of back and forth between the WG and the research community.
What’s changed?

Available Tools

Protocol analysis tools have matured since TLS 1.2
- primitives (HKDF, AEAD)
- modelling key exchange (ACCE, multi-stage KE)
- program verification (miTLS)
- automated tools (Tamarin, ProVerif)

Post-2008 a design-break-fix-release cycle can thrive!
What’s changed?

Available Tools

Protocol analysis tools have matured since TLS 1.2
- primitives (HKDF, AEAD)
- modelling key exchange (ACCE, multi-stage KE)
- program verification (miTLS)
- automated tools (Tamarin, ProVerif)

Impact and Incentives

- WG uses secure primitives and responds to research community’s needs, easing analysis
- research community appreciates the complexity of the protocol and use cases
- many top-tier publications prior to official release

Implementers and researchers seem to understand each other better.
Can we do *even* better?

- Many cooks in the kitchen brings conflict
- Rapidly moving target! Analyses become easily outdated
- TRON 1.0 - full set of requirements missing
Beyond TLS 1.3

- Is this newer, collaborative process unique to TLS?
- How does this process compare to ISO, NIST?
- What’s best for critical protocols such as TLS?
<table>
<thead>
<tr>
<th></th>
<th>IETF (TLS 1.3)</th>
<th>ISO</th>
<th>NIST (SHA-3)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
<td>Open</td>
<td>Closed</td>
<td>Open competition</td>
</tr>
<tr>
<td><strong>Organisation</strong></td>
<td>WG</td>
<td>WG</td>
<td>Teams</td>
</tr>
<tr>
<td><strong>Membership</strong></td>
<td>Individuals</td>
<td>National Bodies</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Contributions</strong></td>
<td>Many-to-one</td>
<td>Many-to-one</td>
<td>One-to-one</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>Free</td>
<td>$175</td>
<td>Free</td>
</tr>
<tr>
<td><strong>Analysis</strong></td>
<td>Prior-to-deployment</td>
<td>Post-deployment (sometimes pre)</td>
<td>Prior-to-deployment</td>
</tr>
</tbody>
</table>

**protocol**

**primitives**
Closing remarks

- Move from design-release-break-patch to design-break-fix-release enabled by better tools and greater engagement of the academic community
- Newer process allows for preemptive decision making and hopefully produces a stronger protocol, requiring less patching
- Perhaps requirements analysis-design-prove-release process would have been better
- Competition model as employed by NIST potentially suits TLS