Reactive and Proactive Standardisation of TLS

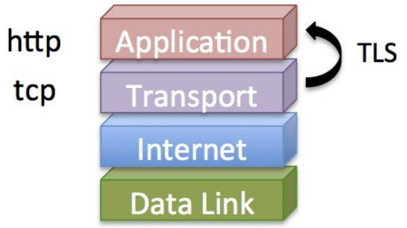
Kenny Paterson and **Thyla van der Merwe**
Royal Holloway, University of London

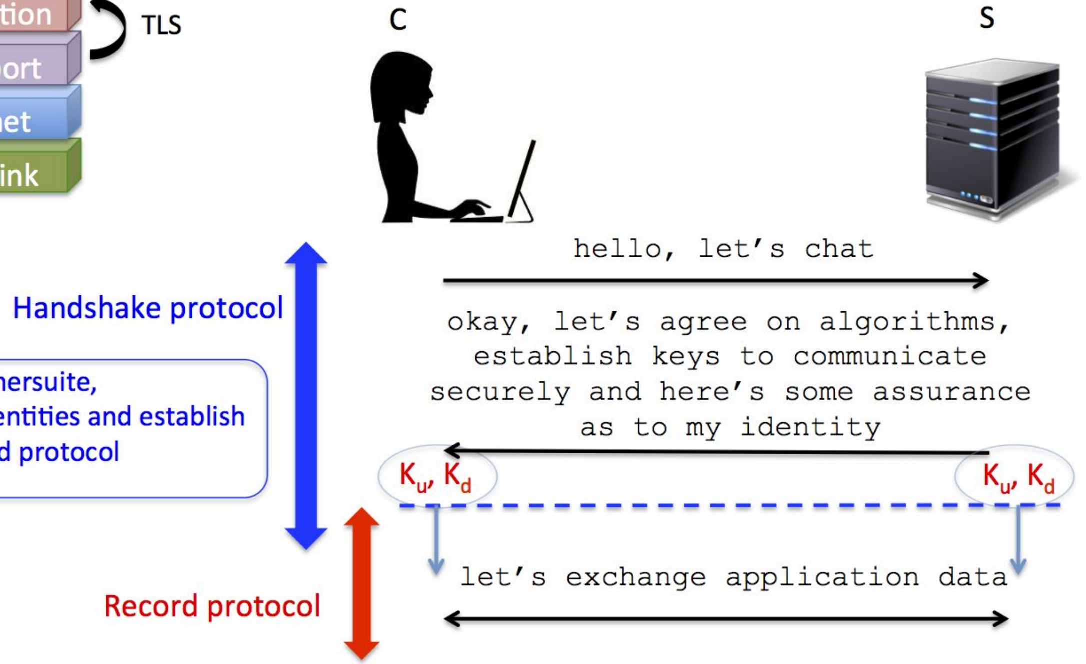
Security Standardisation Research (SSR)
5 December 2016

Motivation

- TLS is the *de facto* standard for securing communications on the Web
- Pressure to improve the protocol's efficiency and the need to address the many weaknesses identified in TLS 1.2 and below → TLS 1.3
- Analysis-prior-to-deployment design philosophy for TLS 1.3 vs post-deployment-analysis for TLS 1.2 and below. **Why?**
- What can the standards community learn? What standardisation model best fits critical protocols such as TLS?

TLS and the IETF




C
S
hello, let's chat
okay, let's agree on algorithms, establish keys to communicate securely and here's some assurance as to my identity
K_u, K_d
K_u, K_d
let's exchange application data
Handshake protocol
Record protocol

Negotiate ciphersuite, authenticate entities and establish keys for record protocol

Provide confidentiality and authenticity of application data using keys established in the Handshake protocol

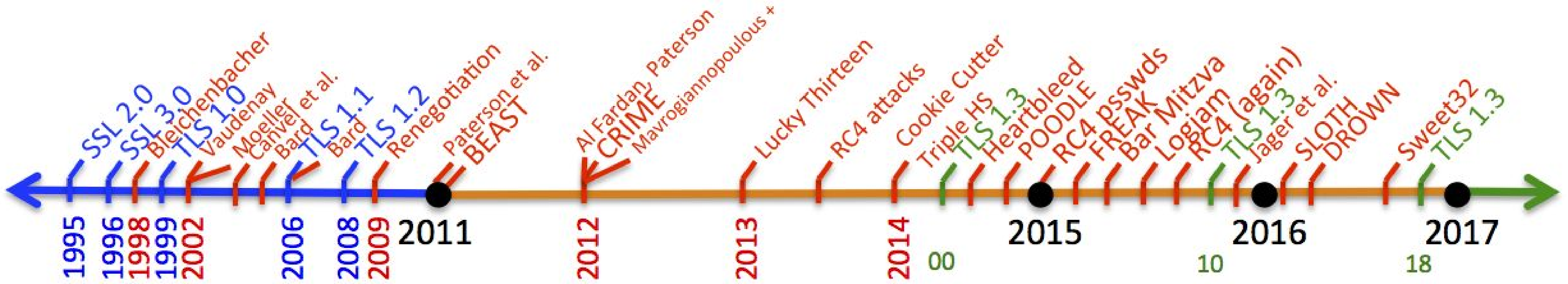
- Started life as the Secure Sockets Layer (SSL) protocol, developed by Netscape
- SSL 2.0 (1995) → SSL 3.0 (1996)

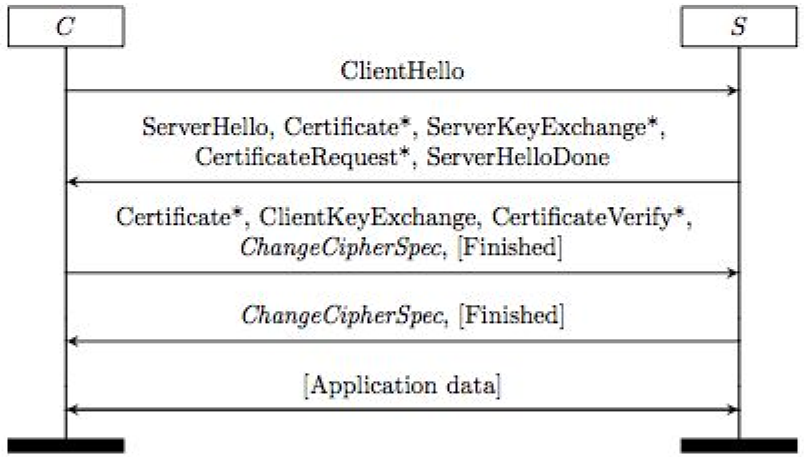
TLS 1.0 (1999) → TLS 1.1 (2006) → TLS 1.2 (2008)

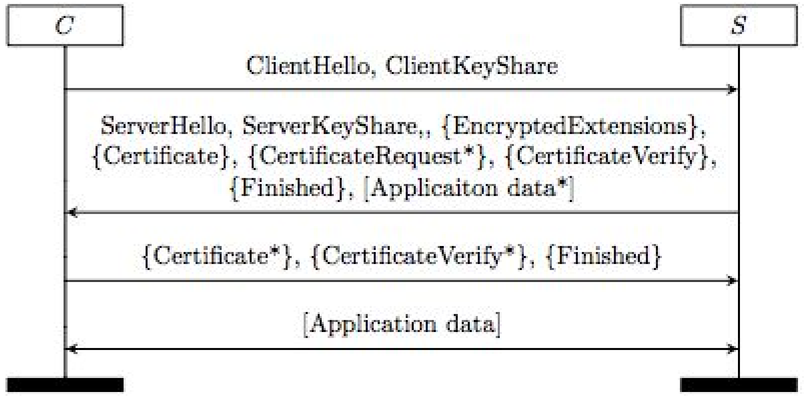
- Started life as the Secure Sockets Layer (SSL) protocol, developed by Netscape.
- SSL 2.0 (1995) → SSL 3.0 (1996)

IETF

TLS 1.0 (1999) → TLS 1.1 (2006) → TLS 1.2 (2008)




C
S
ClientHello
ServerHello, Certificate*, ServerKeyExchange*,
CertificateRequest*, ServerHelloDone
Certificate*, ClientKeyExchange, CertificateVerify*,
ChangeCipherSpec, [Finished]
ChangeCipherSpec, [Finished]
[Application data]

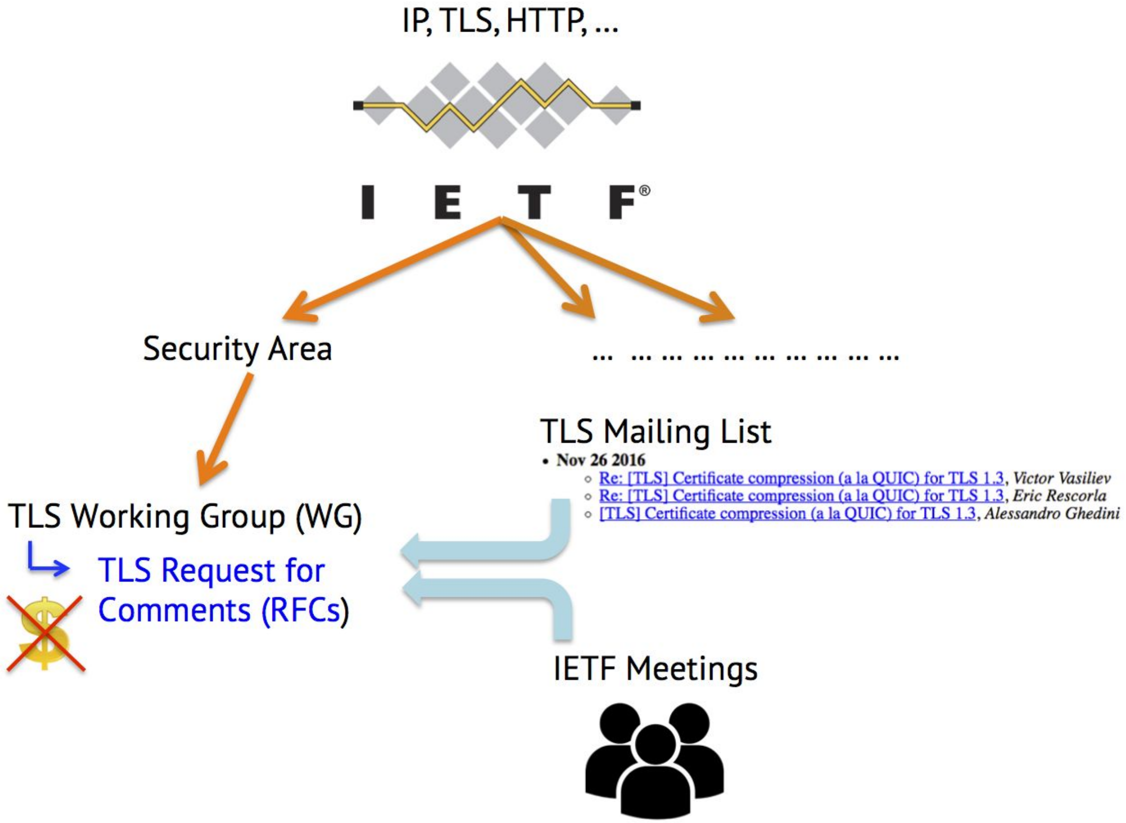

C
S
ClientHello, ClientKeyShare
ServerHello, ServerKeyShare,, {EncryptedExtensions},
{Certificate}, {CertificateRequest*}, {CertificateVerify},
{Finished}, [Applicaiton data*]
{Certificate*}, {CertificateVerify*}, {Finished}
[Application data]

15

Fig. 1. TLS 1.2 handshake

Fig. 2. TLS 1.3 (EC)DHE handshake

- More of the handshake encrypted, 1-RTT, only (EC)DHE
- Resumption replaced by Pre-Shared Keys (PSKs)
- Renegotiation removed, 0-RTT functionality, and AEAD only

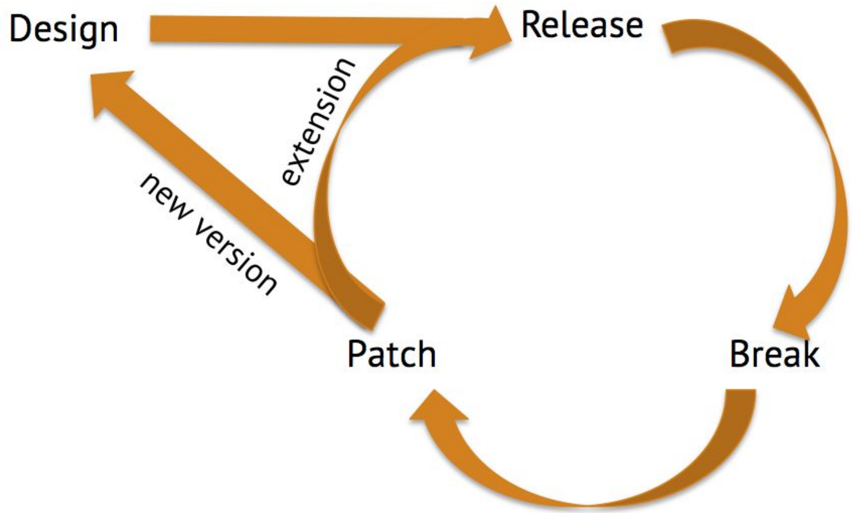


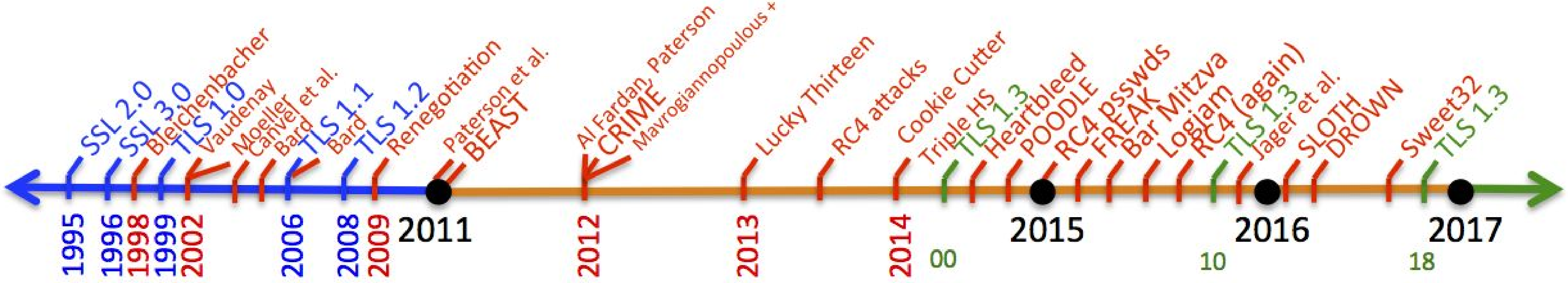
- Nov 26 2016
  - Re: [TLS] Certificate compression (a la QUIC) for TLS 1.3, *Victor Vasiliev*
  - Re: [TLS] Certificate compression (a la QUIC) for TLS 1.3, *Eric Rescorla*
  - [TLS] Certificate compression (a la QUIC) for TLS 1.3, *Alessandro Ghedini*

- No formal membership
- **Open model** for standards development - no barriers to entry, many-to-one development, and no financial barriers to adoption

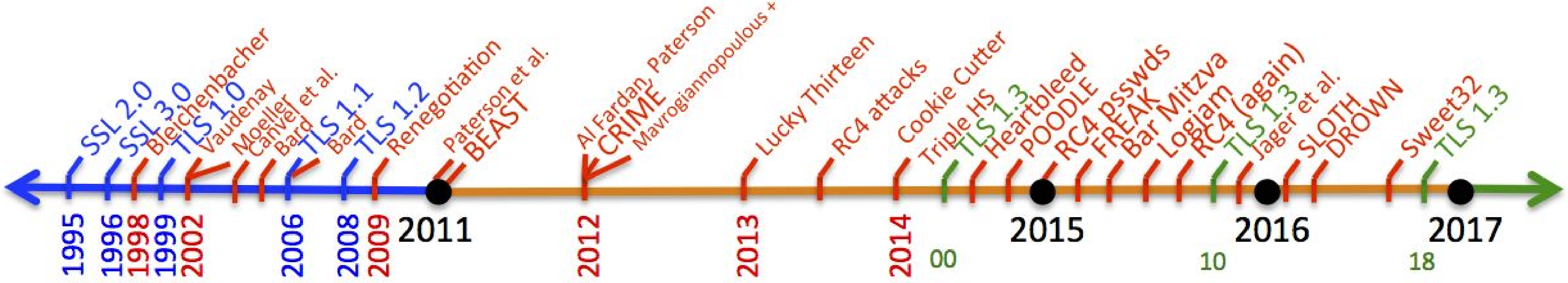
TLS 1.2 and below - Design, Release, Break, Patch

- Development followed a **reactive** standardisation process
- An attack → releasing a extension OR making the change in the next version of the standard

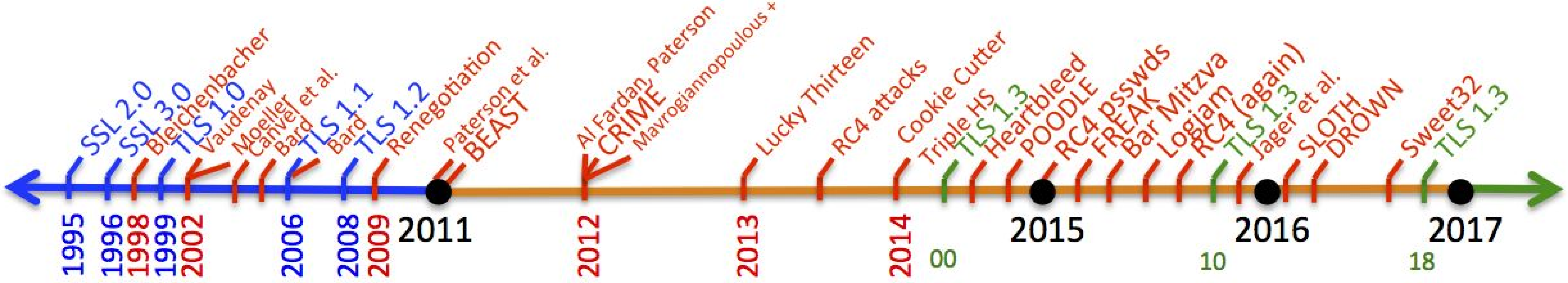




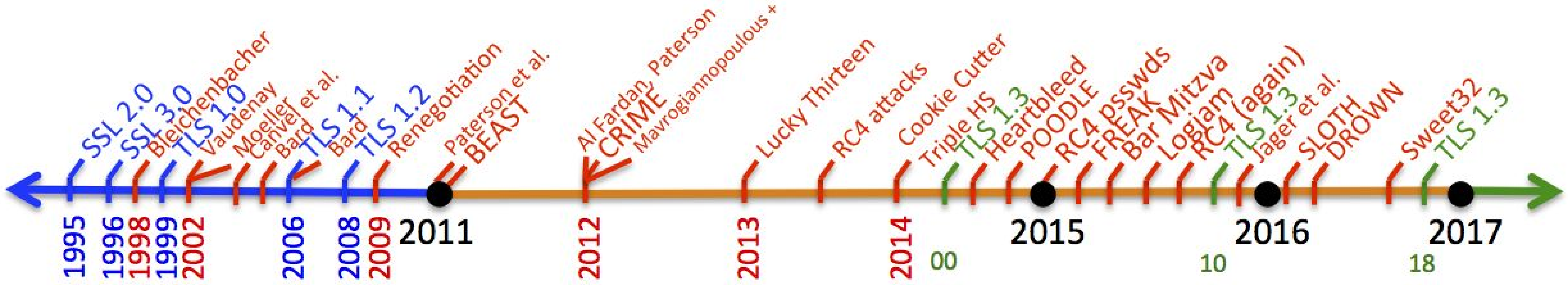
- **Bleichenbacher attack** affects SSL 3.0 - uses an RSA PKCS#1 v 1.5 padding oracle to uncover the pre-master secret
- Briefly addressed in TLS 1.0 - a mechanism that eliminates the oracle (TLS 1.1 and TLS 1.2)
- Attack re-enabled in various forms - Jager et. al, DROWN
- Better to switch to PKCS#1 v2.1? Not done to maintain backwards compatibility



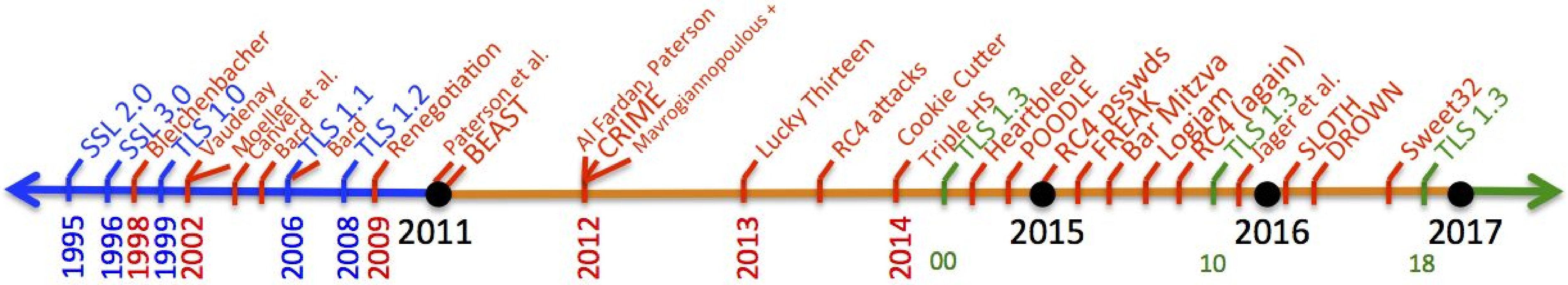
- **Vaudenay's padding oracle attack** affects TLS 1.0 - exploits CBC-mode padding format in the MEE construction to recover plaintext
- Addressed in TLS 1.1 - try to keep record processing time constant
- Left a small timing side-channel, not believed to be exploitable - cue Lucky Thirteen!
- Easier to replace the MEE construction at an earlier stage?
- Attacks need to be practical before a change is considered



- **Renegotiation attack** of Ray and Rex exploits the lack of binding between an attacker's initial handshake and a subsequent renegotiation handshake
- Attacker convinces the server that all data came from the client
- Mandatory extension - include `Finished` in renegotiation `Hello`
- Resurrected by the Triple Handshake attack
- Analysis tools premature prior to 2014?



- **BEAST attack** of Duong and Rizzo affects TLS 1.0 and exploits the known chained-IV vulnerability of Moeller and Bard to recover plaintext
- Opened the floodgates - introduced techniques that made the attack practical and everyone took notice.
- TLS 1.1 removed this vulnerability BUT deployed implementations did not move as quickly - TLS 1.0 is still popular today!



- **RC4** suggested as the counter-measure to BEAST
- RC4 keystream has long been known to be biased - using the new BEAST techniques, researchers started mounting increasingly practical attacks
- Deprecated by the IETF in February 2015
- Could have been phased out a long time ago, before attacks became so powerful, particularly with AES support present

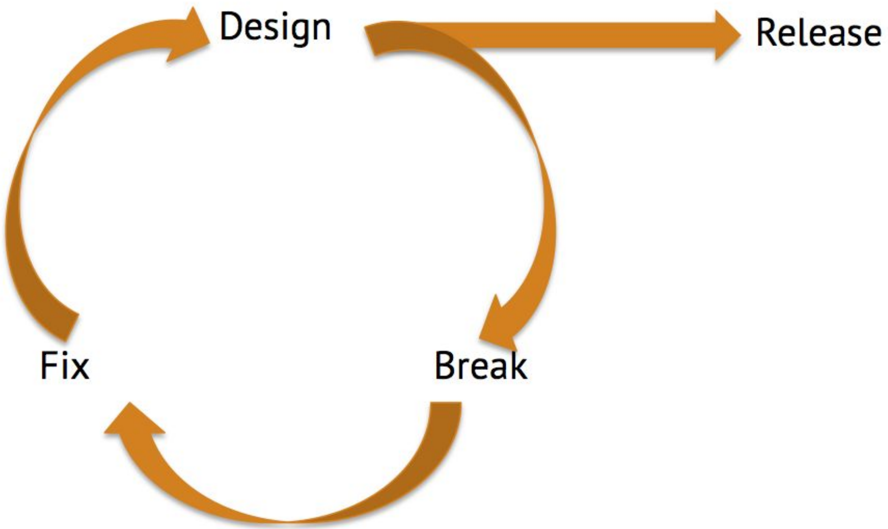
Attack	Damage	Fix	Resurrected
Bleichenbacher	SSL 3.0, recover keys	Note in TLS 1.0 (1.1, 1.2)	Jager et al., DROWN, others
Vaundenay	TLS 1.0, recover plaintext	Addressed in TLS 1.1	Lucky Thirteen, POODLE (related)
Renegotiation	TLS 1.2 and below	Mandatory extension	Triple Handshake
BEAST	TLS 1.0, recovery plaintext	Addressed in TLS 1.1	Made practical with new techniques!
RC4	TLS 1.2 and below	Eventually deprecated	Old weakness

Contributing factors

- Is a more cautious approach warranted for critical protocols?
- 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 engagement by the academic community - reward came from producing high impact attacks
- Incentive model leaves users vulnerable to attack and imposes a patch action

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



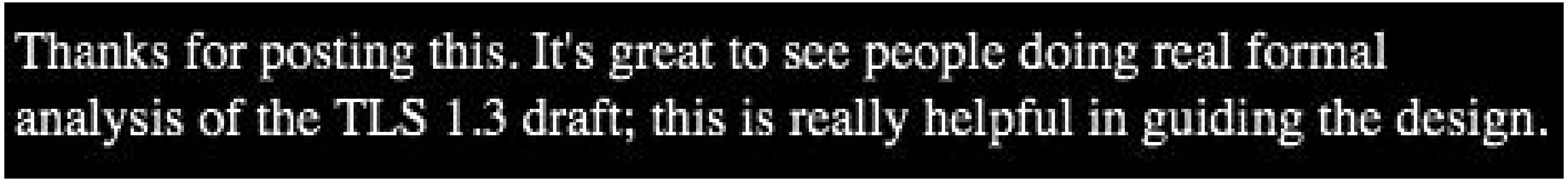
- `draft-00 - draft-05`
  - removal of compression (CRIME attack)
  - inclusion of a *session hash* (Triple Handshake attack)
  - removal of renegotiation (RENEGOTIATION attack)
  - removal of MEE (Lucky Thirteen attack)
  - Handshake and Record protocols no longer overlap
  - analysed by Dowling et al., as well as Kohlweiss et al. - provided valuable feedback to the WG on TLS 1.3 design

Academic community starts to get heavily involved!

- `draft-07`
    - becomes highly influenced by OPTLS of Krawczyk and Wee
    - OPTLS uses ephemeral DH and offers 0-RTT and PSK modes
    - key derivation is similar to OPTLS, using HKDF designed by Krawczyk
- `draft-08 - draft-09`
    - removal of SHA-1 and MD5 (SLOTH)

WG draws inspiration from secure designs and acknowledges the research community’s concerns.

- `draft-10`
  - Cremers et al. perform an automated analysis in the symbolic setting, looking at the interaction of the different handshakes
  - Li et al. develop a computational model and find `draft-10` to be secure
  - The work by Cremers et al. finds a potential attack in the newly proposed post-handshake authentication mechanism - communicated via the mailing list - fixed in `draft-11`


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.

- "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
    - cryptographic protocol analysis tools have matured since TLS 1.2
        - primitives - HKDF, authenticated encryption
        - modelling secure channels and key exchange - ACCE, multi-stage KE
        - program verification - miTLS
        - automated tools - Tamarin and ProVerif

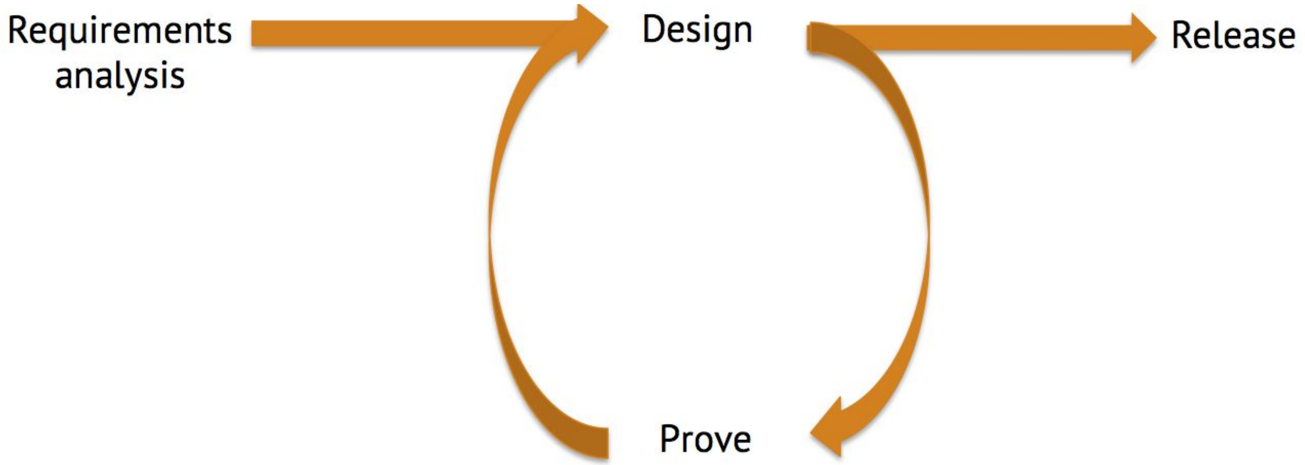
Post-2008 a design-break-fix-release cycle can thrive!

- Involvement, impact and incentives
    - WG has removed weak primitives and switched to secure designs
    - WG has responded to the academic community’s needs - easing analysis of the protocol
    - academic community appreciates the complexity and many use cases of the protocol
    - many top-tier publications prior to official release

Implementers and researchers seem to understand each other better.

Can we do 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?

IETF® vs ISO vs NIST

	IETF (TLS 1.3)	ISO	NIST (SHA-3)
Model	Open	Closed	Open competition
Organisation	WGs	WGs	Teams
Membership	Individuals	National Bodies	N/A
Contributions	Many-to-one	Many-to-one	One-to-one
Cost	Free	$ 175	Free
Analysis	Prior-to-deployment	Post-deployment	Prior-to-deployment

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