TLS Extractors

Eric Rescorla
Network Resonance
ekr@networkresonance.com
General Idea

• Other protocols want to use TLS negotiation
  – But for some reason access keying material

• Examples
  – SCTP Auth
  – DTLS-SRTP
  – TCP Auth
  – EAP
Example: DTLS-SRTP

Alice  

Offer fingerprint=XXX 

ServerHello use_srtp=SRTP_AES128_CM_SHA1_80, SRTP_AES128_CM_SHA1_32, ...

Certificate, CertificateRequest, ServerHelloDone 

Certificate, ClientKeyExchange, CertificateVerify

ChangeCipherSpecs, Finished

ChangeCipherSpecs, Finished

SIP

DTLS

Early Media

Normal Media

SRTP

Answer fingerprint=YYY

SIP

SRTP

Bob

Offer fingerprint=XXX

Offer fingerprint=XXX

ClientHello

use_srtp=SRTP_AES128_CM_SHA1_80, SRTP_AES128_CM_SHA1_32, ...

Certificate, CertificateRequest, ServerHelloDone 

Certificate, ClientKeyExchange, CertificateVerify

ChangeCipherSpecs, Finished

ChangeCipherSpecs, Finished

Alice Atlanta Biloxi Bob

Offer fingerprint=XXX

Offer fingerprint=XXX

Offer fingerprint=XXX
Simple Idea

- General technique for generating keys from TLS handshake (*Extractor*)

- Requirements
  - Each *exported keying material* (EKM) is unique
  - Infeasible to go from $EKM_1$ to $EKM_2$
  - Infeasible to go from $EKM$ to $MS$

- Algorithm is: $EKM = PRF(master\_secret, label, SecurityParameters.client\_random + SecurityParameters.server\_random)[length]$

- Labels MUST be registered
Changes From Last Version

- Labels no longer MUST have a fixed prefix
- Text that you should somehow indicate you’re doing this
  - Like with an extension
Where to from here?

• Comments?
• Should we accept this as a WG item?
• draft-rescorla-tls-extractor-01.txt