Bootstrapping Key Infrastructure over EAP

draft-lear-eap-teap-brski

E. Lear, O. Friel, N. Cam-Winget
Cisco
Related Draft

BRSKI over IEEE 802.11
draft-friel-brski-over-802dot11

O. Friel, E. Lear, M. Pritikin cisco
M. Richardson Sandelman Software Works
What problems are we trying to solve?

• What Wi-Fi networks support BRSKI?
• How to avoid the device onboarding against the wrong network?
• What credentials does the device use before and after BRSKI bootstrap against a Wi-Fi network?
• How long does it take / what signalling is required for the device to determine that the network is untrusted?
• How complicated is the device state machine when switching from candidate network A to candidate network B?
• How complicated is the device state machine during network onboarding?

draft-friel-brski-over-802dot11 outlines some possible solutions but does not make any final recommendations

draft-lear-eap-teap-brski focuses on one candidate solution: running BRSKI inside a TEAP tunnel
Refresher: ANIMA BRISKI

1a. $\text{SIG_P(requestvoucher)}$
1b. $\text{SIG_R(SIG_P(requestvoucher))}$
1c. $\text{SIG_M(voucher)}$
1d. $\text{SIG_M(voucher)}$

2a. simpleenroll + CSR
2c. LDevID
2b. Sign CSR

• Bootstrapping pledge trusts nothing except the manufacturer

• Pledge discovers registrar service on local domain (GRASP, mDNS, DNS options)

• Registrar is akin to a smart middlebox that proxies voucher requests to a manufacturer service that the device trusts

• Manufacturer issues a signed voucher instructing the pledge to trust the registrar
What we *could* do with current mechanisms

1. 802.11 connect and 802.1X using IDevID
2. DHCP, IP, DNS
3. BRSKI Voucher Request / Reject
4. Reboot
5. 802.11 connect and 802.1X using IDevID
6. DHCP, IP, DNS
7. BRSKI Voucher Request / Accept
8. Reboot
9. 802.11 connect and 802.1X using LDevID
10. DHCP, IP, DNS
11. Access resources
What we would like to do

1. 802.11 connect and 802.1X using IDevID with BRSKI messages tunnelled inside EAP TLS tunnel

2. 802.11 connect and 802.1X using IDevID with BRSKI messages tunnelled inside EAP TLS tunnel

3. DHCP, IP, DNS

4. Access resources
ANIMA BRSKI

1. Provisional TLS connection to Registrar
2. Establish Trust via Voucher
3. Verify TLS connection
4. Download Trust Anchors
5. Enrol to get a cert
EAP-TEAP is a good fit

1. Provisional TLS connection to Registrar
2. Establish Trust via Voucher
3. Verify TLS connection
4. Download Trust Anchors
5. Enrol to get a cert

TEAP supports Server Unauthenticated Provisioning
2. New TLVs can be transported in TLS tunnel
3. Device can verify server after TEAP Phase 2 completes
4. Trusted-Server-Root TLV exists
5. PKCS#7 and PKCS#10 TLVs exist
EAP-TEAP BRSKI Architecture

- TEAP server and BRSKI Registrar could be co-located
- BRSKI Registrar and CA could be co-located
EAP-TEAP BRSKI Flow

- New TEAP TLVs defined
  - VoucherRequest
  - Voucher
  - VoucherStatus*
  - EnrollmentStatus*
  - CSR-Attributes*
- BRSKI TLVs must be exchanged prior to Crypto-Binding
- BRSKI is not a new EAP Method
  - BRSKI exchange is not an inner method
  - No need for Channel-Binding

* Usage shown in detailed flows in draft
Summary

• Running BRSKI as part of 802.1X simplifies device onboarding state machine

• EAP TEAP is a good fit for BRSKI

• Defining new TEAP TLVs vs. a new EAP method seems simpler

• Request EMU adoption for draft-lear-eap-teap-brski
Discussion