Man-in-the-middle in Tunnelled Authentication
Cambridge Security Protocols Workshop, April 2003

N. Asokan, Kaisa Nyberg, Valtteri Niemi
Nokia Research Center
A tale of two protocols

• In the beginning..
  • an authentication method is designed and deployed for some need
  • user credentials are provisioned, at great expense

• ..then a framework protocol is developed;
  • to transparently support multiple authentication methods
  • authentication methods are plugged in to the framework

• .. new applications arise; framework doesn’t quite do the job
  • missing bits: session keys, mutual authentication, identity privacy
  • designing a new protocol is not a desirable option
  • provisioning new credentials is even less desirable

➢ Use it with another protocol that provides missing features
AKA and EAP/AKA: example authentication protocol

- AKA: authentication and key agreement protocol for 3GPP
  - mutual authentication, session key derivation

- EAP: an authentication framework
  - supports multiple authentication mechanisms

- EAP/AKA: plugging AKA into EAP
  - allows WLAN access authentication using cellular credentials
**PEAP: example of tunnelled authentication**

- **Ciphersuite (for link layer)**
- **Ciphersuite (for link layer)**

- **Client**
  - EAP API
  - EAP Method

- **NAS**
  - EAP conversation
  - over PPP or 802.11 link
  - Keys
  - Trust

- **Backend Server**
  - EAP API
  - EAP Method

**Server-authenticated tunnel**
Tunneled authentication

Client

Front-end authenticator

Tunnelling protocol
Server authenticated
secure tunnel establishment

secure tunnel

Backend Server (Tunnel authentication)

Backend Server (Legacy authentication)

Authentication protocol
Client authentication
The same tale in different guises

- PIC - ISAKMP and EAP: provisioning credentials based on legacy authentication
- IKEv2 Secure Legacy Authentication
- PANA over TLS: Authentication for Network Access
- HTTP Digest Authentication and TLS
PEAP with EAP/AKA

Client

NAS
(Front-end authenticator)

Backend Server
(TLS Authentication)

Establishing a PEAP tunnel (server authenticated)

EAP-Request/Identity

PEAP Part 1 – TLS based on server certificate

TLS(EAP-Request/Identity)

secured by PEAP TLS tunnel

TLS(EAP-Response/Identity(IMSI))

Rand, AUTN, XRES, IK, CK
[e.g., over DIAMETER]

TLS(EAP-Request/AKA-Challenge (RAND, AUTN))

WLAN master session key (based on TLS tunnel key)

Data traffic on secured link

IMS [e.g., over DIAMETER]
Stolen WLAN link

MitM against PEAP+EAP/AKA

Client

MitM (Front-end Authenticator)

NAS

Backend Server (TLS Authentication)

Backend Server (AKA Authentication)

Establishing a PEAP tunnel (server authenticated)
EAP-Request/Identity

PEAP Part 1 – TLS based on server certificate

TLS(EAP-Request/Identity)

secured by PEAP TLS tunnel

TLS(EAP-Response/Identity(IMSI))

IMSI Request

IMSI

RAND, AUTN

RES

TLS(EAP-Request/AKA-Challenge (RAND, AUTN))

TLS(EAP-Response/AKA-Challenge (RES))

WLAN master session key
(based on TLS tunnel key only)

IMSI [e.g., over DIAMETER]

RAND,AUTN,XRES,IK,CK
[e.g., over DIAMETER]
Conditions for failure

1. Same credential used in both tunnelled & untunnelled modes
2. Tunnelling protocol does not perform mutual authentication
3. Keys from authentication protocol not used for subsequent protection
Fixing the problem

1. Enforcing that same credential is not used in both modes
   - maybe feasible in some cases
   - not exactly “legacy authentication” anymore
   - server authentication brings in new problems
   - unnecessary restriction on strong authentication methods

2. Require mutual authentication in tunnelling protocol
   - if that is possible, no need for tunnelling in the first place

3. Cryptographically bind tunnelling and authentication protocol
   - binding can be explicit or implicit
   - requires authentication protocol to provide a key to be used in binding
   - requires changes to tunnelling protocol or framework
   - does not improve the security of weak authentication protocols
Current status

- Some authors of tunnel proposals informed in October 2002
- General agreement that this is indeed a problem
  - opinions differ on what the solution should be
- Subsequent changes to several proposals to reduce the impact of the problem
  - EAP/AKA (v-05)
  - PEAP (v-06)
  - IKEv2 (v-05)
  - PANA over TLS (v-01) → PANA (v-00)
  - EAP SIM GMM → EAP binding
  - ...
Are there any lessons here?

• This is all obvious, at least in hindsight

• So why did it happen?
  • re-use of credentials is unavoidable in practice
  • re-use of protocols is also unavoidable in practice
  • framework equalizes all authentication methods
    • mutual authentication, key agreement etc. not visible
  • tools for/knowledge of protocol validation not accessible to designers