Security analysis of PEAP protocol using NuSMV tool.

Martin Očenáš

Brno University of Technology, Faculty of Information Technology
Božetěchova 1/2. 612 66 Brno - Královo Pole

iocenas@fit.vutbr.cz

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

6. prosince 2017

Motivation

- Security protocols are vital in these days.
- Security flaw in such protocol may have serious consequences.
- PEAP is widely used security protocol.

Scope of research

- Analyse PEAP for security weaknesses.
- Create formal model of the protocol.
- Use NuSMV as formal model checker.
- Check that PEAP hold security requirements.

802.1X

- Used to authenticate device on the network, against authentication server.
- Send authentication data from peer to authentication server.
- Authentication server does not need to perform the authentication itself.

EAP

- Extensible authentication protocol.
- Authenticate user.
- Originally designed for P2P connections. LAN variant is called EAPOL.
- Possibly provide encryption key for L2 communication.
- Has many extensions: EAP-MD5, LEAP, PEAP,...
- Peer/Supplicant, authenticator, authentication server.

PEAP

- Procted EAP Protocol.
- Defined in RFC draft.
  - https://tools.ietf.org/html/draft-josefsson-pppext-eap-tls-eap-06.
- Used for 802.1X authentication.
- Protects EAP communication with TLS.

PEAP

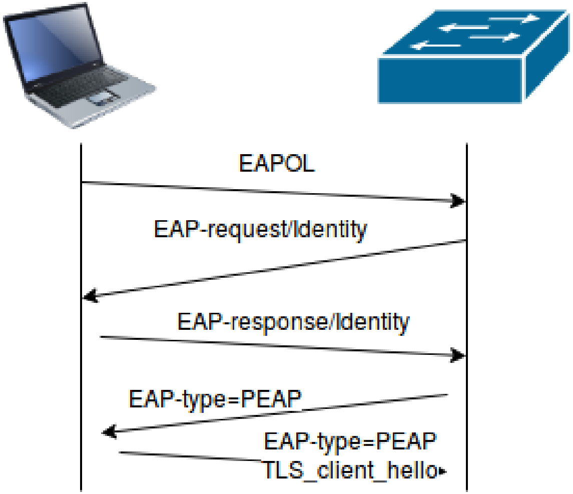
Phase 1

- Establish TLS session.
- Authenticate server.

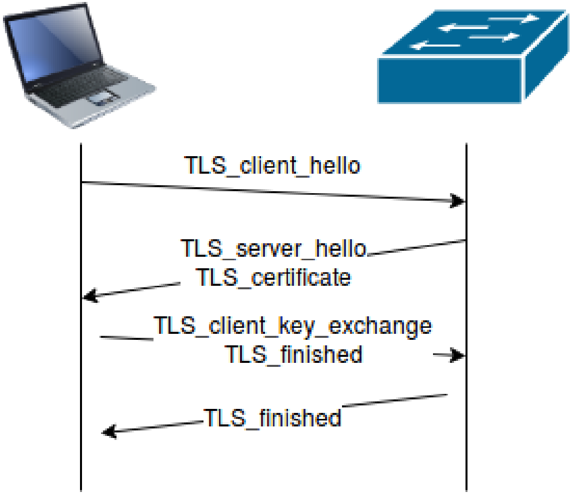
Phase 2

- Communication protected by TLS.
- Authenticate peer.

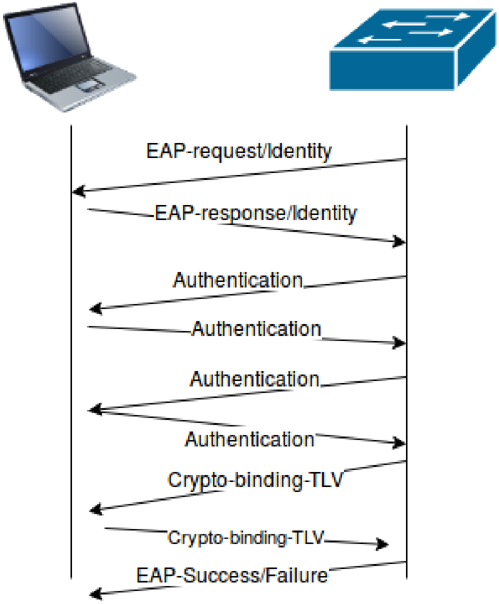
PEAP Initiation



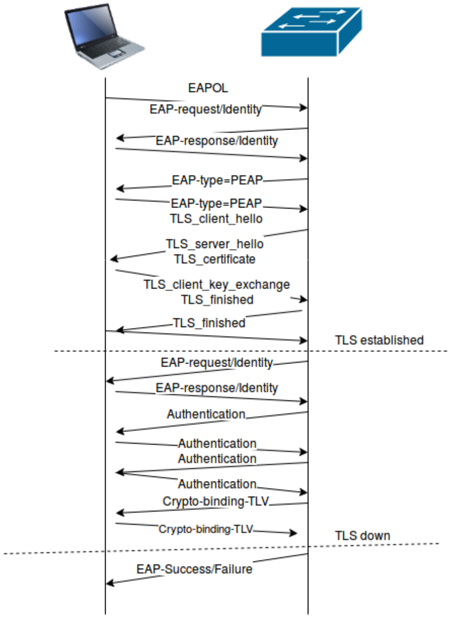
PEAP TLS handshake



PEAP Phase 2


EAP-request/Identity
EAP-response/Identity
Authentication
Authentication
Authentication
Authentication
Crypto-binding-TLV
Crypto-binding-TLV
EAP-Success/Failure

PEAP


EAPOL
EAP-request/Identity
EAP-response/Identity
EAP-type=PEAP
EAP-type=PEAP
TLS_client_hello
TLS_server_hello
TLS_certificate
TLS_client_key_exchange
TLS_finished
TLS_finished
TLS established
EAP-request/Identity
EAP-response/Identity
Authentication
Authentication
Authentication
Authentication
Crypto-binding-TLV
Crypto-binding-TLV
TLS down
EAP-Success/Failure

Formal verification and NuSMV

Formal verification

- Has formal model of system.
- Has formal assertions.
- Validates that specified assertions holds in the system.
- Produces a proof that assertions holds, or a counter example when it does not hold.

Model checking

- Variant of formal verification.
- Expands all possible states of the system.
- Validates that assertions holds in all states.

NuSMV

- Symbolic Model Checker.
- Extension of SMV.
- Tool for formal verification, based on model checking.

Primitives of NuSMV

- Modules - contains logic of one module.
- Variables might be module local or shared.
- Uses next-state logic to generate
- CTL or LTL formulas to specify assertions.

Processes in NuSMV

- Independently run modules.
- NuSMV chooses the process that will run.
  - Simulates scheduler of the operating system.
- NuSMV supports fairness.

NuSMV program example

```
MODULE main
  VAR
    semaphore : boolean;
    proc1 : process user(semaphore);
    proc2 : process user(semaphore);
  ASSIGN
    init(semaphore) := FALSE;
  SPEC AG ! (proc1.state = critical
        & proc2.state = critical)
  SPEC AG (proc1.state = entering
        -> AF proc1.state = critical)

MODULE user(semaphore) ...

FAIRNESS
  running
```

NuSMV program example

```
MODULE user(semaphore)
  VAR
    state : {idle, entering, critical, exiting};
  ASSIGN
    init(state) := idle;
    next(state) :=
      case
        state = idle : {idle, entering};
        state = entering & !semaphore : critical;
        state = critical : {critical, exiting};
        state = exiting : idle;
        TRUE : state;
      esac;
    next(semaphore) :=
      case
        state = entering : TRUE;
        state = exiting : FALSE;
        TRUE : semaphore;
      esac;
```

NuSMV output

```
--specification AG !(proc1.state = critical
       & proc2.state = critical)  is true
--specification AG (proc1.state = entering
       -> AF proc1.state = critical)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 1.1 <-
    semaphore = FALSE
    proc1.state = idle
    proc2.state = idle
  -> Input: 1.2 <-
    _process_selector_ = proc1
    running = FALSE
    proc2.running = FALSE
    proc1.running = TRUE
  -- Loop starts here
...
```

Verification of PEAP

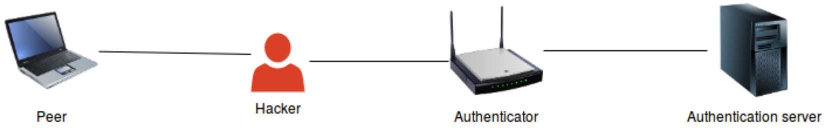
Security goals

- Secrecy of peer's identity,
- secrecy of peer's credentials,
- impossibility of pretending to be auth. server,
- impossibility of downgrade attack,
- secrecy of the communication key.

Security assumptions

- Attacker is in the Man in the Middle position.
- Up to date cryptography is flawless.
- Variably: client and server are well configured.
    - Latest version of TLS.
    - Up to date cypher suites.
    - Client can validate the server's certificate.

Hacker position



Nodes in the systems

- Peer
- Hacker.
- Authenticator.

Behavior of the nodes:

- Peer and authenticator behave deterministically based on the protocol.
- Variable - client can verify server's certificate.
- Protocol abstracted.
- Hacker is non deterministic, may do anything with the communication.

Implemented security goals

```
EF (peer.state = finished & server.state = finished);
EF (peer.state = TLS_ESTABILISHED
        & server.state = TLS_ESTABILISHED);
AG (server.state = finished
        -> server.peer_ID_known = TRUE);
AG (peer.state = TLS_ESTABILISHED
        -> peer.Certificate = SERVER_CERT);
AG (hacker.server_pretended = FALSE);
AG (hacker.user_id_known = FALSE);
AG (peer.state = TLS_ESTABILISHED
        -> peer.AGREED_VERSION = TLS12);
AG (server.state = TLS_ESTABILISHED
        -> server.AGREED_VERSION = TLS12);
```

Results

- With proper configuration of peer and authentication server, no attack found.
- With peer unable to validate server's certificate, hacker can pretend to be the server.
- NuSMV is not the right tool for validation of security protocol.

Thank You For Your Attention !