Things to consider for a new version of 802.1X

LinkSec suggestions

09/23/03

Paul Congdon

PAR Issues to consider

- Primary purpose is to support requirements of LinkSec
- EAP would expect this to be a transport for EAP exchanges (assuming we care to support EAP – i.e. EAP has requirements for us)
- Do we specify the Specific Authentication protocol to be used (e.g. PEAP, EAP-???)
- Compatibility with current 802.1X versions or totally new protocol?

High-level concepts to consider

- EAP independence
- Symmetric operation (true controlled port on both sides)
  - Still want mode configuration (requestor/responder)
- Allowing key machines to run at anytime, not sequenced with EAP-Success
- True shared media support (e.g. 802.3 multi-user)
  - Is there still a port now or something else?
  - Instantiate per VLAN?
- New EAPOL-Hello messages or Start messages being sent by both sides (don't overload initial EAP-Request to determine if someone is out there).
  - Include network discovery in this new message so supplicants can select identity. (do it this way instead of via LLDP).
  - Consider supporting Bob M.'s AUP requirements in this message.
  - Model after 802.11 Probe/Beacon capability, with some extensibility for messages (AUP) and discovery/identity selection

High-level concepts to consider (cont)

- Protected EAPOL control frames
  - EAPOL-Logoff, EAPOL-Hello, EAPOL-Start, etc (are we sure?)
- Include a 4-way handshake key machine
  - Perhaps interpretation of messages is done by external entities
- Separate ‘EAP authorized’ state, ‘key exchange’ state and ‘port in use’ state into three indications that need to be tested independently. Valuable for pre-auth ideas
- Insert a frame mux/de-mux above the MAC interface instead of the broadcast scheme that exists today (see Mick’s SecY diagrams)
- Indicate that EAPOL frames below the mux shall not be encrypted using the encryption schemes of the SecY (again see Mick’s diagram)

Details of Consideration Notes

Considerations – EAP Independence

- Certain EAP methods supports many (if not all) of the requirements for MAC Sec Authentication and Key agreement
  - Credential flexibility
  - Confidentiality and Integrity for key agreement
  - Freshness (EAP-TLS, EAP-TTLS, PEAP)
  - Mutual Authentication (EAP-TLS, EAP-TTLS, PEAP)
  - Identity Hiding (EAP-TTLS, PEAP)
- EAP does NOT inherently support the following MAC Sec Authentication and Key agreement requirements:
  - Fast rekey and fast handoff
  - EAP key framework working group maybe addressing this
  - DOS resistance – a specific non-objective, but some improved support in rfc 2284bis.
- EAP standards are still evolving
  - Recent changes in RFC 2284bis
  - EAP working group

Considerations – Symmetric

- Desire a true controlled port on both sides (802.1aa is pretty much there now)
- Can we implement a single state machine for both supplicant and authenticator roles?
- Certain authentication protocols (e.g. EAP) still expect an initiator and responder

Considerations – Independent Key Exchange

- Currently we define a specific sequence of key exchange and EAP-Success message transmission
- Need to support re-keying without re-authentication
- Interpretation of key message should be extensible to support new information
- Currently key machines are replaceable, but defined by other standards (e.g. 802.11i).
- Consider supporting both replaceable machines with extensible message content.

Considerations – Shared Media

- How are multiple simultaneous independent authentications managed on shared media?
  - Do we need a new definition of a port?
  - Do we need to use uni-cast addresses?
- Provider bridging model may be similar to the shared media problem
- Pre-authentication appears as a shared media problem on the DS side

Consideration – New Start-up Sequence

- Currently we overload EAP-Req/ID by authenticator to find a supplicant
- Need to address discovery of network identities to select user identity
- Need to address AUP requirements before starting authentication exchange.
- Consider a new EAP-Hello message to initiate a peer conversation

Consideration – Protected EAPOL

- Currently state machines are reset or restarted by certain unprotected frames (EAPOL-Start, EAPOL-Logoff)
- Do we really want to establish our own protection below the authentication protocol?
  - EAP is responsible for authenticating identity today
  - EAP protects itself with certain methods (EAP-TTLS, PEAP)
- MAC Sec encryption hardware could be used to sign certain frames or at least MAC Sec keying material could be used

Consideration – Key Machines

- Current key machines are not secure nor reliable
- 4-way handshake by 802.11i is secure, but not ‘owned’ by 802.1X
- Possible to define message exchange protocol, without decoding all message content

Consideration – Independent signals

- Three distinct signals to consider
  - Authentication protocol authorized
  - Keys exchanged and installed
  - Port is in use (not just pre-authenticated)
- Controlling the ‘operStatus’ of the controlled port should combine the above concepts.
- Consider separate machines or processes to assert and control each signal.

More backup

More details

- Do we need to specify a demux/mux function for frames on the controlled and uncontrolled port? Today, frames are broadcast to both ports and the filtering is done above that level (see Mick's SecY diagrams) – assume yes for now.
- If we have this mux, we need to redefine the PAE to live on top of both the controlled and uncontrolled port because sometimes it would have to receive encrypted EAPOL frames (i.e. during re-auth). This multi-port PAE would need some policy that knows when to use which port (both sending and receiving). Alternatively – never send EAPOL encrypted and let EAP take care of the protection of the conversation.