BTNS Core

draft-ietf-btns-core-00.txt

Nicolas.Williams@sun.com

PAD and BTNS Extensions

- PAD extensions
  - Wildcard matching extension
    - Match any peer public key
  - Last PAD entry should use this wildcard to match BTNS peers (if desired)
  - Peer IDs for BTNS peers are coerced to a new ID type (PUBLICKEY) whose value is a peer's public key as used to authenticate it
    - Specific to KEs and peers that use public keys for auth.
- BTNS extension: "BTNS_OK' flag

Example

- Node A generates a private/public key pair and a self-signed cert for that public key
- Node A initiates IKE exchange with node B using that key and cert
- Node A matches BTNS catch-all PAD entry in node B's PAD
- Assuming node B matches a PAD entry on node A and all works out then an SA pair is installed in the two nodes' SADBs

Example

- Node A's traffic to node B needs to match BTNS_OK SPD entries (searched for according to the matching PAD BTNS entry) in order to move
- Node B may or may not be a BTNS node, but it must be authorized by node A's PAD accordingly

Connection Latching

draft-ietf-btns-connection-latching-00.txt

Nicolas.Williams@sun.com

What it is

- Binding of logical packet flows (“connections”) to series of SAs which share common criteria, particularly peer IDs, but also algorithms, etc..
- Latching happens on send/receive of a packet flow’s first packet
- Specified in very general terms: who (IPsec, IP, ULP or application) passes what to whom

How it Works

- Optional: app may provide some parameters to bind to (e.g., ESP/AH, algs, key lengths to use)
- ULP passes a latch template down with initial packet on output to IP/IPsec
  - IPsec initiates KE if need be (according to policy and latch template), fills in the latch template (shared with or passed back to ULP)
- On inbound IPsec passes SA# up to ULP
  - ULP fills in a latch on initial packet based on SA characteristics or checks that the SA matches existing latch, else drop