MINUTES -- SMUG MEETING
OSLO, NORWAY
July 11, 1999

Notes by David Balenson

PARTICIPANTS

1. N. Asokan, Nokia, n.asokan@nokia.com
2. David Balenson, NAI Labs, balenson@tislabs.com
3. Mark Baugher, Intel, mbaugher@intel.com
4. Brad Cain, Nortel Networks, brad@nortelnetworks.com
5. Ran Canetti, IBM, canetti@watson.ibm.com
6. Roger Kermide, Motorola Labs, ark008@email.mot.com
7. Amit Kleinman, NDS, akleinmann@ndsisrael.com
8. Isidor Kouvelas, cisco, kouvelas@cisco.com
9. Eric Harder, US DoD, ejh@tycho.ncsc.mil
10. Thomas Hardjono, Nortel, thardjono@baynetworks.com
11. Hugh Harney, Sparta, hh@sparta.com
12. Michael Luby, Digital Fountain, luby@dfountain.com
13. Hilarie Orman, Novell, ho@novell.com
14. Theo Pagtzis, UCL, ucactxp@cs.ucl.ac.uk
15. Mohammed Payravian, IBM, payravn@us.ibm.com
16. Colin Perkins, UCL, c.parkins@cs.ucl.ac.uk
17. Bob Quinn, Stardust.com, rcq@stardust.com
18. Tony Speakman, Cisco, speakman@cisco.com
19. Gene Tsudik, USC/ISI, gts@isi.edu
20. Joe Wesley, Sun, joe.wesley@sun.com
21. Brian Whetten, Talarian, whetten@talarian.com

AGENDA

09:00 - 10:30 Presentations
09:00 - 09:30 Hugh Harney (Sparta), Security Policy for Groups
09:30 - 09:40 Thomas Hardjono (Nortel), Moving GIO Box for Reference Framework
09:40 - 09:55 Ran Canetti (IBM), Host arch for group member - update
09:55 - 10:15 Ran Canetti (IBM), Source authentication: current techniques and challenges
10:15 - 10:30 Gene Tsudik (ISI)
10:30 - 12:30 Building block discussions
12:30 - 01:30 Lunch
01:30 - 04:30 Building block discussions

HUGH HARNEY (SPARTA) - SECURITY POLICY

Security policy
- a description of relevant:

- mechanisms
- characteristics
- behaviors

Mechanisms
- policy establishment
- assumed
- dynamic
- group establishment
- peer SA mechanisms (exact or compliant set)
- peer PKI (exact or set)
- key download
- GSA for security
- compromise and/or rekey
- group data transmission
- one or more security profiles per application

Characteristics
- PKI definition and cross matches
- data access control
- compromise rules
- at what point do we declare a compromise?
- one might value communication more than security
- or, security may be paramount
- rekey rules
- additional security parameters
- permission field entries
- alternative permission sites
- environmental parameters

Behaviors
- delegated action authorization parameters
- key server (initial establishment)
- policy server
- compromise recovery server (rekey)

* server may not be separate; may be individual member; or distributed service

1-N vs. N-N (straw positions re. security policy for each type of communications)
- 1-N
- single source -> static policy
- peer SA ? receiver authentication only
- rekey vs. compromise recovery
- single PKII/root
- no source authentication
- N-N
- N-N ? negotiation (more likely)
- peer SA ? mutual suspicion
- compromise recovery
- multiple PKI/roots
- source authentication

THOMAS HARDJONO (NORTEL), MOVING GIO IN REFERENCE FRAMEWORK

Hardjono: If have GIO, small box with long lines, or tall box with short lines?
Balenson: GIO also interacts with receivers.
Baugher: What is the GIO?
Hardjono: Have GIO, or not?

RAN CANETTI (IBM), SOURCE AUTH: CURRENT TECHS AND CHALLENGES

Source authentication mechanisms for multicast

The problem: how to authenticate the data when nobody but the source (and maybe the key server) is trusted

Parameters for evaluating solutions:
- security
- length / bandwidth overhead
- time at sender
- time as receive
- latency of transmission
- reliability requirements

Overview of schemes:
1. signing large blocks / stream signatures
   + efficient, low bw overhead
   - requires reliability, has latency
2. tree-hash+sign (Wong, et. al., McCarthy)
   + efficient, medium bw overhead
   - latency, requires state
3. (2) + one-time signatures (Rohatgi)
   + very efficient, no latency
   - high bw overhead
4. sign each packet
   + no latency, no state
   - inefficient, high bw
5. multi-MAC's (CG/MNP 99)
   + efficient, low bw, no latency, no state
   - secure only against small coalitions, complex key management

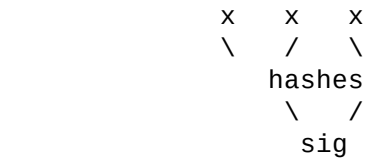
Discussion of Schemes

1. Signing large blocks
- use public key signatures
- very secure
- very easy key management
- need verification key from trusted party
- relatively efficient: small signature relative to large data
- big problem is reliability and latency since need all data to verify

2. Sign each packet

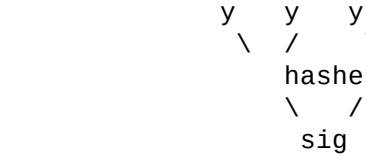
- very secure
- highly inefficient
- easy key management
- each signature is 40/128 bytes long
- can generate at most 50 signatures/second, so
  only 50 packets/second

3. Tree hash and sign
- Wong, et. al.
- McCarthy draft describing, available on home page?



- send pkt w/ hashes and sig (periodically)
  x hashes sig
  x hashes sig
- can verify each packet individually
- can cache verified signature
- still some latency problems

4. Use of one-time signatures
- Draft sent out few weeks ago
- using one-time signatures
- prepare in advance:
- one-time keys, independent of message,
  compute in advance
  tree hash and sign



- can process each packet individually:
- append one-time sig, one-time key, hashes, sig
- verify each packet
- no latency
- pay price of appending data (about 260 bytes--non-trivial) to
  each packet

5. Multi-MACs

- have many MAC keys
- each receiver has subset of keys
- sender has all keys
- sender MACs each message with all keys
- each receiver verifies with keys known to him (say random

half of keys)

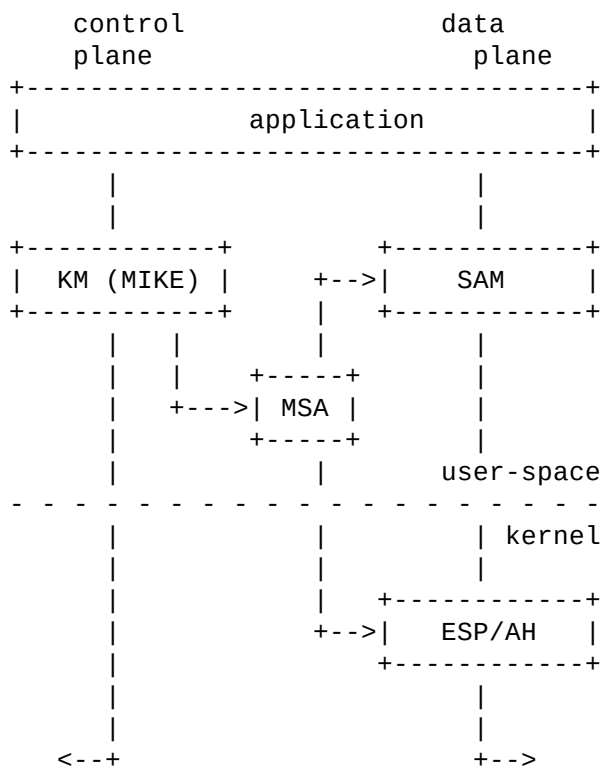
```
                k1,   k2, ...,  kn
sender:       x m1,   m2, ...,  mn
```

user 1: verifies ones for which he knows keys
user 2: verifies ones for which he knows keys
- no user can forge packets for other users
- main drawback: only secure against small size coalitions;

RAN CANETTI: AN IPSEC-BASED HOST ARCH FOR SECURE MULTICAST (UPDATE)

Ran Canetti
P-C Cheng
Frederique Giraud
Dimitris Pendarakis
JR Rao
Pankaj Rohatgi
Debanjan Saha

Basic architecture: reminder



SAM source authentication module
MSA holds group key, authentication key, signs information?

Implementing/testing this using FreeSWAN

Update:
- successfully combined multicast with IPsec (FreeSWAN on Linux)
- additional concern: intra-host access control (for multi-user hosts)
- need an additional module?
- in/out of kernel
- SPI assignment (old stuff)
- need entity to assign & distribute SPI
- SPI based on multicast address, for consistency
  w/ IPsec architecture
- sequence numbers (old stuff)
- for n-n, doing multiple 1-n's? IPsec-specific question
  Do we want IPsec?

GENE TSUDIK (ISI), CLQ_API, GROUP KEY AGREEMENT API BASED ON CLIQUES PROTOCOL SUITE

Underlying group communication subsystem must provide for reliable synchronized event notification
- group joins (single member)
- group leaves (single member, voluntary)
- partitions (leaves "en masse")
- node failures or disconnects (involuntary)
- merges (heals)
- NOTE: the above is not an absolute requirement!

CLQ_API
- clq_join — new member calls after getting context from GC
- clq_proc_join — called by GC to hand over group context after updating key share
- clq_update_ctx — each member calls as the last step of JOIN, LEAVE, MERGE or REFRESH; updates key
- clq_leave — each member calls after receiving a LEAVE event
- clq_refresh_key — any member can call (usually GC) to update its share
- clq_update_key — GC and new member call during MERGE 1st stage)
- clq_factor_out — all members call during MERE (2nd stage)
- clq_merge — new GC calls to merge groups (3rd stage)
- clq_first_member only called by "founding" member

CLQ_API
Diagram depicting GC, New member, and All members calls for JOIN and LEAVE

CLQ_API
Diagram depicting GC, New member, and All members calls for REKEY and MERGE

Last slide
- Home page http://www.isi.edu/div7/CLIQUES
- API code available on discretionary basis
- Will be unconditionally released soon
- ID ready this week
- Implementing centralized scheme as a point of comparison

- Integration with SPREAD nearing completion
- TOTEM integration underway
- Experimenting with different groups sizes, geo. distribution
- Results to be published soon

OTHER SMUG MATTERS

(a) SMuG will be getting a page of the upcoming/new
    IRTF web server

    Currently: http://www.irtf.org/charters/secure-multicast.htm

(b) Naming of SMuG Drafts (in the light of recent IETF decisions)

    Should drafts use individual names or irtf-smug names?
    Allow both:

    (1) draft-irtf-smug- .txt (SMuG work product; requires permission
        from SMuG chairs)

    (2) draft-<individual>-smug- .txt (anyone can do)

BUILDING BLOCKS DISCUSSION

(Note: these notes capture some, but not all of the discussion.)

Hardjono:
What do we mean?
Can we define them?
Related to reference framework
Interfaces?

Canetti:
slide w/ sketchy suggestions on building blocks
raises questions for discussion

Suggested build blocks: sketch

(Ia) Data encryption
- assumes shared key among users
- participants: group members (sender(s) and receivers)

(Ib) Data authentication
- same as (Ia)

(II) Key server(s) to member communication (includes KS to KS
communication)
- use both point-to-point communication (login, logout) and multicast

(III) KS(s) to Policy server(s) communication

- report join/leave requests
- obtain PS decision

(IV) KS to Group I/O
- get group information from group I/O

May want to make some of these more modular.
Building block roughly a "protocol" that solves part of problem independent of others.
Breaks things up by functionality, rather than by entity.

Harney: Not supposed to be an architecture for design.

Quinn: PWG as separation between policy decision and enforcement points. III becomes COPS.

Baugher: so not communication, per se, but definition, based on policy framework.

Balenson: very broad view of secure group management policy, encompassing both crypto security and group communications mechanisms, characteristics, and behaviors

Baugher: application policy vs. IP multicast policy?

Challenge to have framework and building blocks general enough to support both application layer and network layer solutions?

Canetti: doubt can do it w/o incompatibilities among components?

Baugher: Can envision combination of AH authentication and application layer confidentiality.

Canetti: Can use ESP w/o modification; authentication will require work.

Hardjono: go through an example or two?

Conception
    |
Announce / Create group
    |
Decide to join
    |
Registration (contact to join, user authentication, pair-wise, long-term security reln)
II, KS-->member
III, KS-->PS
    |
Establish group SA
(rekey arrow to Leave)
II, KS-->members, KSs, members
    |
Participate in secure multicast group

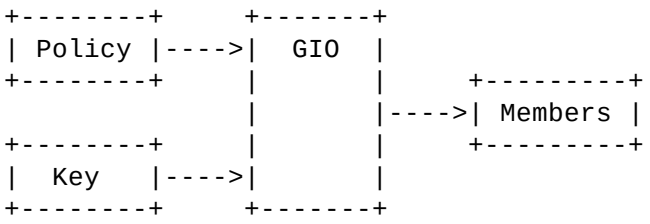
(rekey arrow back to establish group SA)
Ia, Ib, senders, receivers
    |
Leave

So, now which boxes apply to each of these steps?

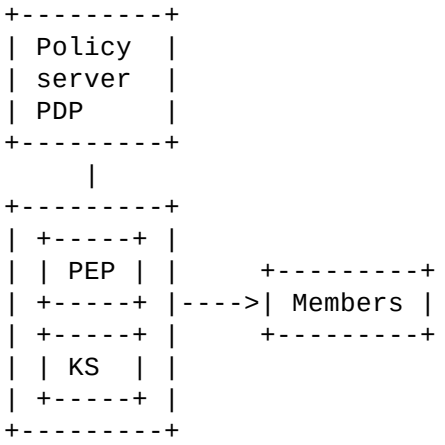
Participate box is Ia and Ib
Establish group SA box is II, policy too?
Registration box is II and III

Much discussion about PS
- does it communicate with member?
- placement of PWG notions of PDP and PEP
- Harney: some members perform PEP?
-
-

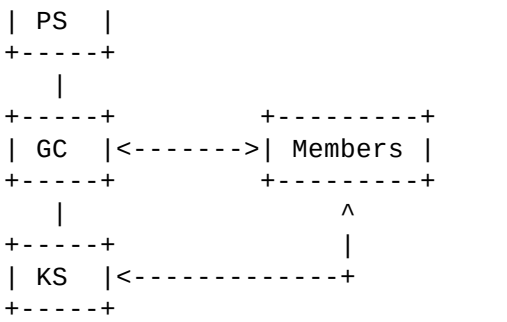
One variation considered:



Another variation considered:



Final variation:



GC <--> member
- member authentication
- policy negotiate
- learning attributes
- monitoring (does group exist?)
- register / de-register

GC <--> KS
- add member
- evict member
- create group
- all via auth'd channel

KS <--> member
- distribute group keying material
- key updates
- member revocation

GC <--> PS
- PEP
- COPS

PS <--> PS
- COPS

GC <--> GC
- ???

KS <--> KS
- ???

Orman: different trust models for
Orman: Group <name, policy, GCs, KSs, Members>

Orman: make sure the GCs are administering the same policy

Balenson/Quinn: PDP in PS; PEP is combination of GC and KS

Whetten: Not heard convincing argument why need separate GC and KS?

Orman: Again, may need trust separation between GC and KS

Balenson: e.g., security officer manages classified employees but does not have access to classified material

Baugher: Have not changed the original very much

BREAK

Canetti recap discussion:
- two building blocks in data plane
- data encryption
- data authentication
- control plane
- 1. GC <--> member
- 2. GC <--> KS
- 3. KS <--> member
- Note: from member viewpoint, same whether GC and KS separate or combined
- GC <--> GC replication protocol
- KS <--> KS replication protocol
- GC <--> PS uses COPS
- PS <--> PS uses COPS
- can combine some of the interfaces by taking union of protocols

---

BRIAN WHETTEN (TALARIAN), RMTP-II REQUIEMENTS

Internet Draft

Example requirements for a RM transport

Issues
1. Division of responsibility
2. Key management / distribution

Functions/Layers
1. Sender authentication
  - its the only thing which needs asymmetric
  - can we do this w/ IPsec?
2. Data encryption (--> group authentication)
  - Requirement that DR (repair-node) cannot decrypt/access data
  - Requirement that retransmissions authentic from DR
  - IPsec --> hop-by-hop --> violates DR requirement
    --> but this is not inviolate
3. Transport protection
  - need to do group authentication on control packets
  - tied closely to network level
  - needs transport header authentication

Conclusions/Recommendations
1. DoS - control packet authentication
   only transport responsibility

2. DoS ties to IPsec
3. Use application security for sender authentication & data encryption
4. Unless DR requirement is not needed
   --> then can use IPsec for all

DR retransmission points for data within the network

Key Division
1. Asymmetrical key for sender authentication
2. symmetric key for data encryption
   --> doesn't have to go to control nodes
3. control traffic uses multiple keys (symmetric)
  - key for each DR to receivers/senders
  - key for inside the control tree
  --> provides protection between different enterprises

SMUG NEXT MEETING?

Meeting in September?

Washington IETF in November

Australia IETF in March?

