Efficient Security

for

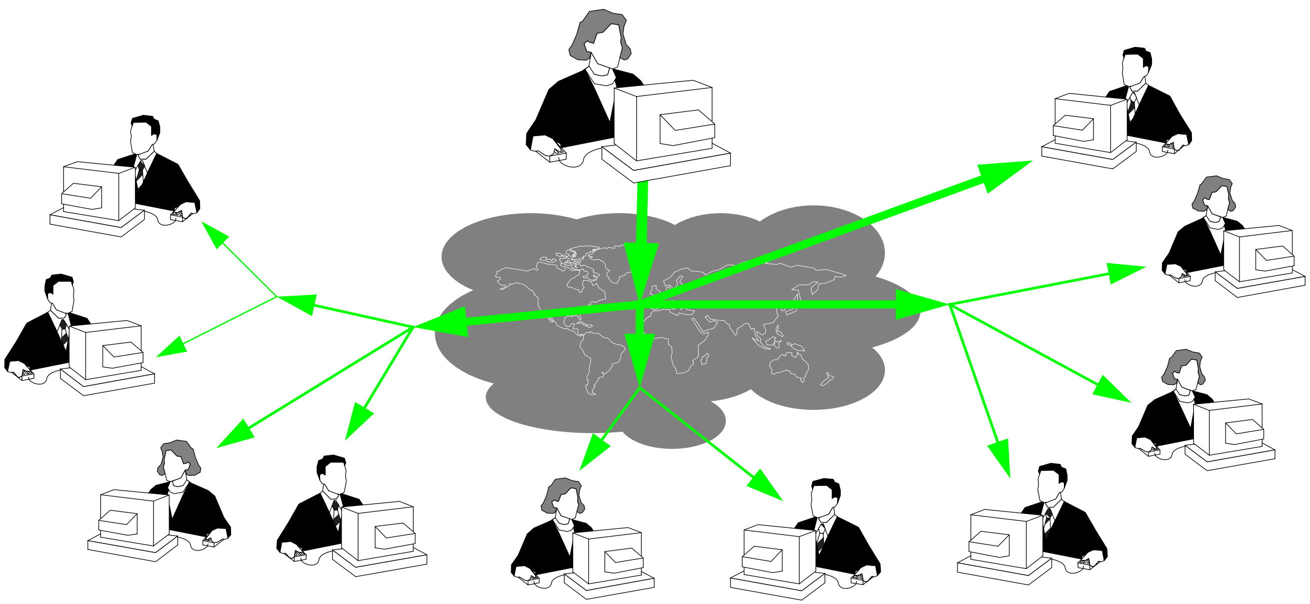
Multicast Groups

Germano Caronni, Marcel Waldvogel, et al.

Computer Engineering and Networks Lab, ETH Zürich
Internet Commerce and Security, Sun Microsystems, Inc.

Contents

- Multicasting applications and security issues
- IPSEC solutions, traditional applications
- New approaches: The SMC-Tree, and others
- Secure IP multicasting in the future



Multicast Functionality

- MBone (IP Multicasting)
- Sender unaware of recipients
- Recipients subscribe with the network (routers)
- Global distribution possible – on demand

- Unreliable unordered transmission of datagrams
- End system transparency, network handles message duplication on router level.

- Related issues: Multicast in ATM, QoS, Flow ID's in IPv6, ToS, IETF WG's

Multicast Applications

Broadcasting

Audio, video, news bulletins, stock exchange quotations, usenet news, ...

Conferencing

Large anouncments, teleteaching, distributed consulting, panel meetings, ...

Distributed Computing

Shared efforts with unknown memberships, redundancy, distributed storage, ...

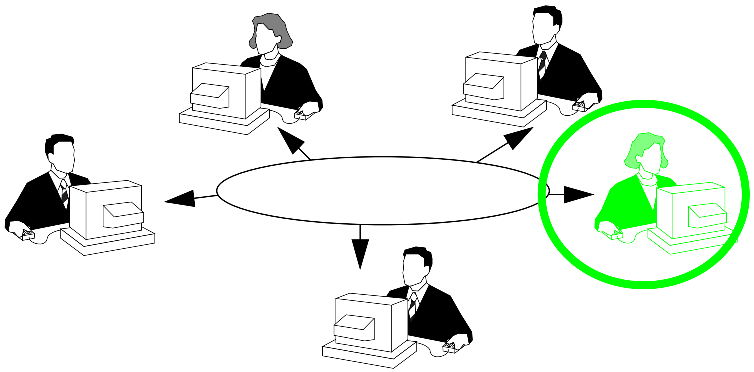
Security Issues (I)

- Focus on privacy
- Traditional security (unicast or centralized variants) cause implosion with large groups
- Trust in intermediate components (routers, gateways) is to be minimzed
- Perfect forward privacy must be possible
- Multicasting may be used in one-way environment, solution must honor this property

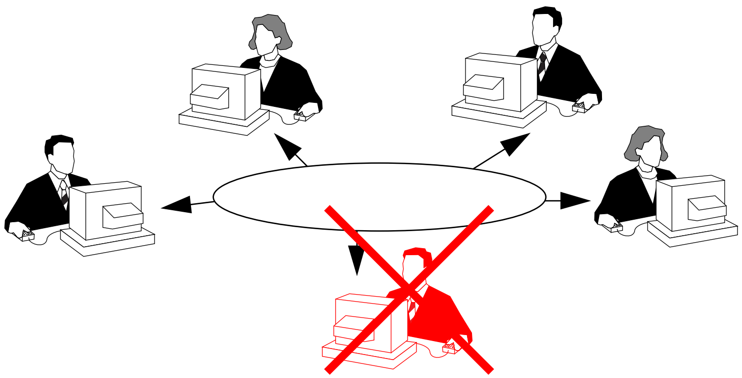
Security Issues (II)

Efficient support of dynamic group operations

Join
Past traffic must be safe, minimal involvement of group



Leave
Future traffic must be safe. Exclusion must be possible 'on the fly'



Existing Solutions

IOLUS (SIGCOMM '97)

- Uses network components to build autonomous sub-groups.
- Multiple stages for messages
- Complex network infrastructure
- Trust in 3rd party components required

Mbone tools

- Manual configuration

NSA Internet Draft (Key Management for Multicast)

Using 'traditional' Secure IP

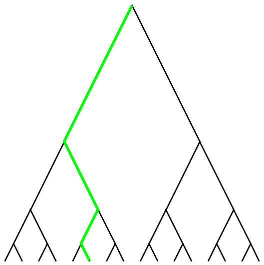
SKIP

- Secure tunnelling of multicast groups between VPN components

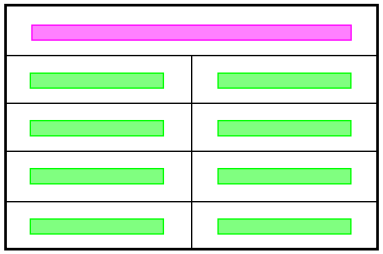
- Manual configuration of ACL and group key to encrypt/ authenticate traffic to/ from group

New Approaches

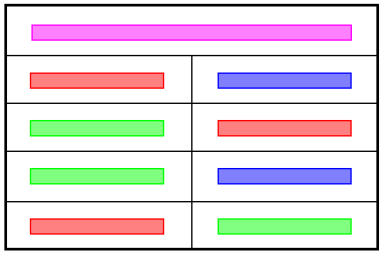
The Tree



The Table



The Distributed Table



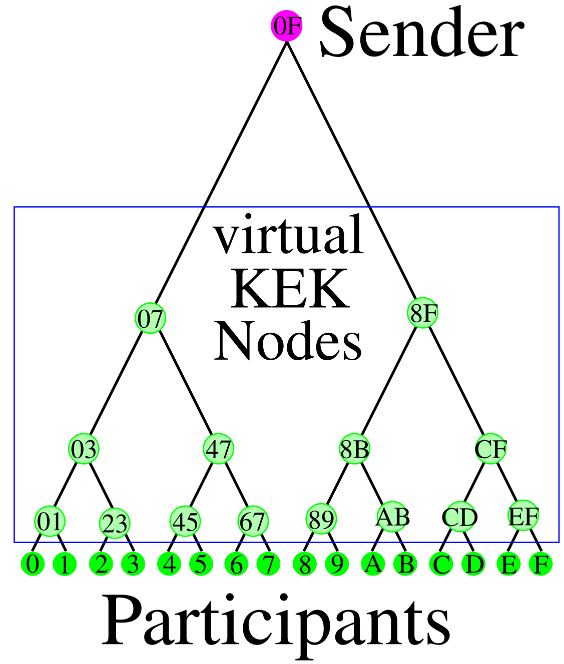
Tree – Properties

+ Privacy of messages
+ No trust in third parties needed
+ Full exploitation of multicasting mechanism
+ Handles dynamic join / leave
+ Minimized amount of computation and traffic per join or leave operation
+ Unability to read traffic out of admission scope

– Sender needs to know all group members
– Setup implosion may occur
– Multiple senders may require synchronisation

Tree – Data Structures

Sender side: (Tree structure)



Participants have ID represen-ting path through tree.
Each node holds one key encryption key (KEK).
Traffic encryption key (TEK) is the tree root.

Receiver side:

TEK	K07	K47	K45	K5

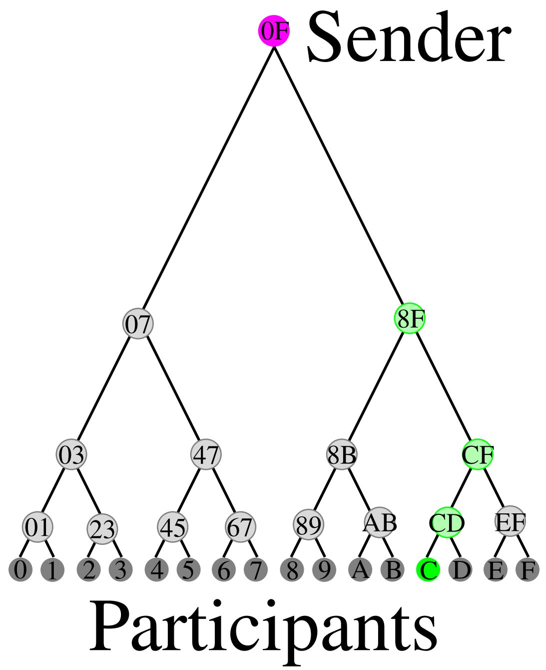
Every participant (e.g. ‘5’) holds keys in his path.

Tree – Join

Example: Participant ‘C’ joins

All key revisions in the path are increased, K <– H(K)

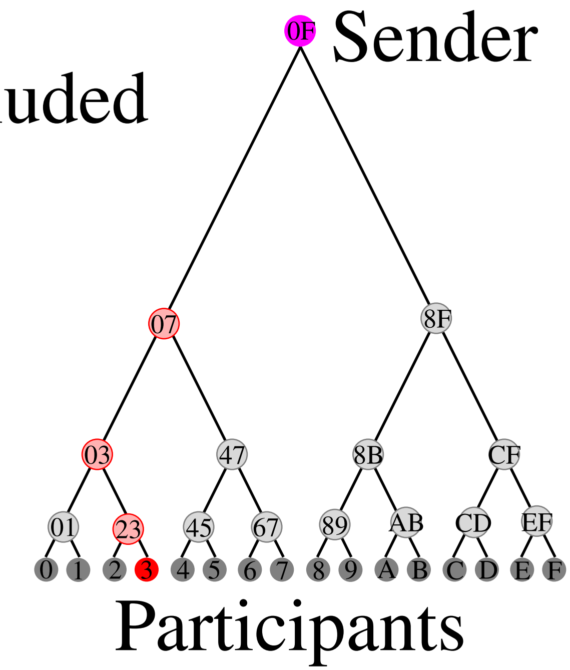
Individual KEK C is generated, C, CD, CF ,8F and the TEK 0F are passed to new participant via unicast channel.



Sender, new receiver: $\log_2(N)$ operations.

Other receivers: 1 to $\log_2(N)$-1 op, eventually.

Tree – Leave



Example: Participant ‘3’ is excluded

The key 3 can be junked, keys 23, 03, 07 and TEK 0F need to be replaced with new versions.

This can be done with one multicast message:

$(23^n)_2$	$(03^n)_{01}$	$(03^n)_{23}$	$(07^n)_{03}$	$(07^n)_{47}$	$(0F^n)_{07}$	$(0F^n)_{8F}$

Operations: $\log_2(N+1)$, less for most receivers.

Tree – Implementation Architecture

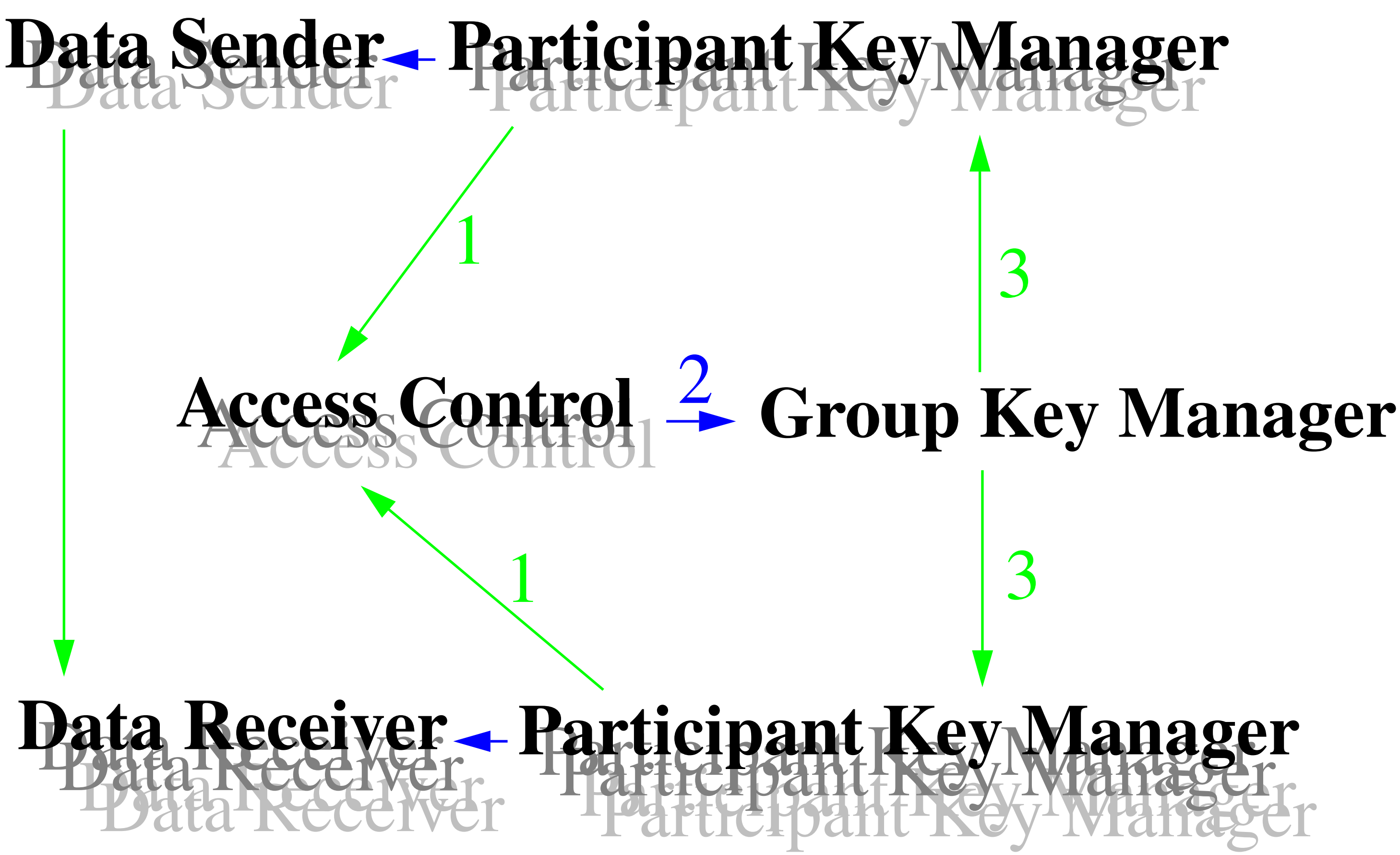


Table – Properties

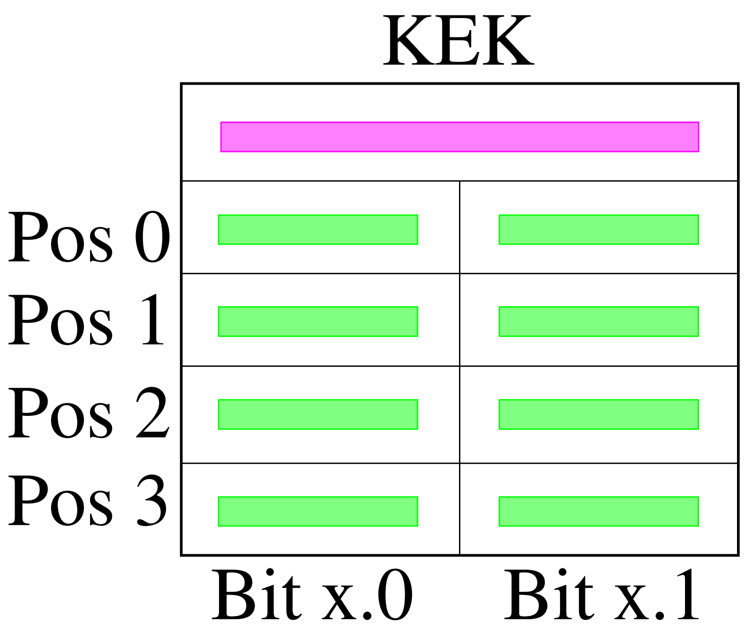
+ Much smaller (fixed) database size
+ Manager does not have to remember all participants
+ In case of manager failure, transfer to a new manager can be done easily.

– Knowledge about current participants may not be available
– Colluding participants are hard to exclude

Table – Data Structures

Sender side: (Table structure)

Participant's ID bit pattern indicates which KEK keys he holds. TEK is shared by all participants.



Receiver side:
Example: Participant '5' holds TEK and his KEK keys.

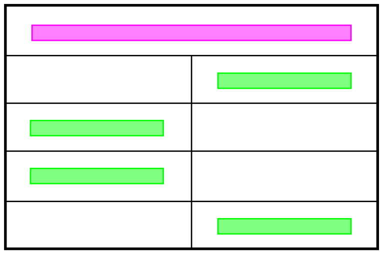


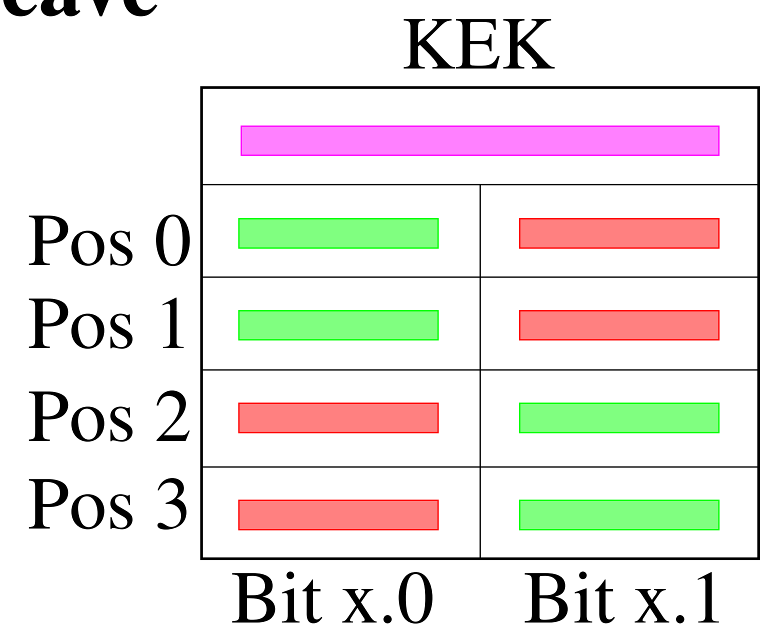
Table – Join

All key revisions of the keys the necomer will get are increased, K <– H(K)

TEK and relevant KEKs are are passed to new participant via unicast channel.

Sender, new receiver: W+1 operations.
Other receivers: 1 to W op, eventually.

Table – Leave



Example: Participant ‘3’ is excluded. 0.1, 1.1, 2.0, 3.0 and TEK need to be replaced.

Done in one message:

$T^n_{(0.0)}$	$(0.1)^n_{Tn(0.1)}$
$T^n_{(1.0)}$	$(1.1)^n_{Tn(1.1)}$
$(2.0)^n_{Tn(2.0)}$	$T^n_{(2.1)}$
$(3.0)^n_{Tn(3.0)}$	$T^n_{(3.1)}$

Operations: 2*W
(W for participants)

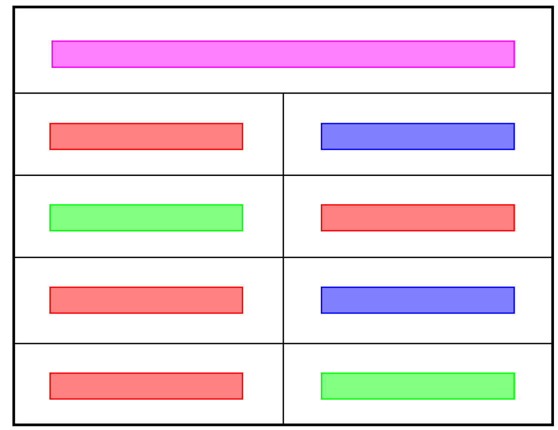
Distributed Table – Properties

+ No global knowledge required by anybody
+ No setup implosion danger

– Multiple participants involved in a join or leave operation
– Colluding participants are hard to exclude
– More complex communication protocol

Distributed Table – Data Structures

- TEK
- known KEK
- ‘owned’ KEK
- unknown KEK



Heartbeat by ‘owners’ indicates responsible addresses for version updates.

Distributed Table – Join

Distributed Table – Leave

Complicated ;-)

Overview of Properties

Tree: (sender has full participant database)

- participant database required for sender
- centralized control
- sender/receiver asymmetrically handled

Table: (sender has global knowledge)

- participant database not needed
- centralized control
- sender/receiver asymmetrically handled

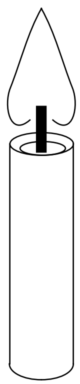
Distributed table: (participants have partial knowledg.)

- no setup implosion
- fully distributed, all participants ‘equal’

Secure IP Multicasting in the Future

- Integration of presented mechanisms as user-level key management components, using GIK concept.
- Encapsulation of multicast payload in SKIP
- Automatic detection and use of key management for accessed multicast groups.

Thank you!



Many thanks to

Radia Perlman,
Rich Skrenta & Tom Markson, Sun Microsystems

Questions?

Email contact address: GEC@ACM.ORG