A Prototype Implementation for Dynamically Configuring Node-Node Security Associations using a Keying Server and the Internet Key Exchange

Suresh Krishnaswamy

Thesis Defense for the Degree of
***Master of Science* in**
Computer Science
University of Kansas

Committee:
Dr. Joseph B. Evans (Chair)
Dr. Gary J. Minden
Dr. W. Perry Alexander

I T T C

Motivation

- Security in Active Networks is complex
  - Many participating entities
  - Complex Threat Model
- Need for an acceptable short term solution
  - End-to-End Security
  - Hop-by-Hop Security
- Our Prototype

  "Design a framework for Hop-by-Hop security, maintaining enough flexibility to allow its use by a larger community"

Organization

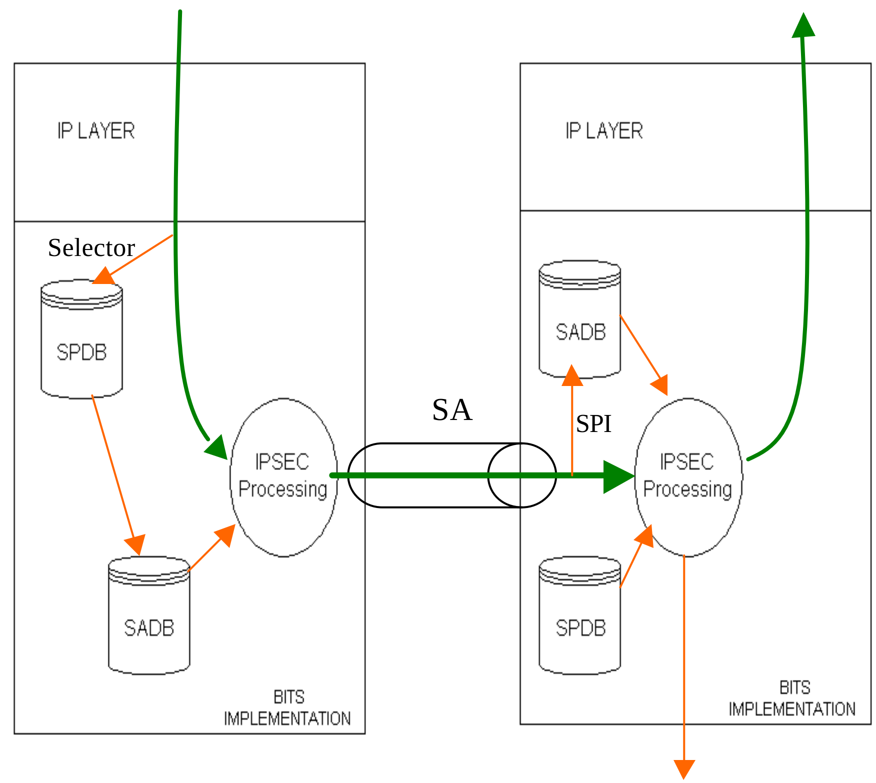
- Components
- Design
- A Sample Topology using our prototype
- Discussion of Results
- Conclusions and Future Work

Prototype Components

- Component choices are motivated by
  - Layer where we should place security services
    - Network Layer - IPSec
  - Authentication framework possible in this layer
    - DNSSEC
  - Keying mechanism
    - IKE

The IPSec Framework

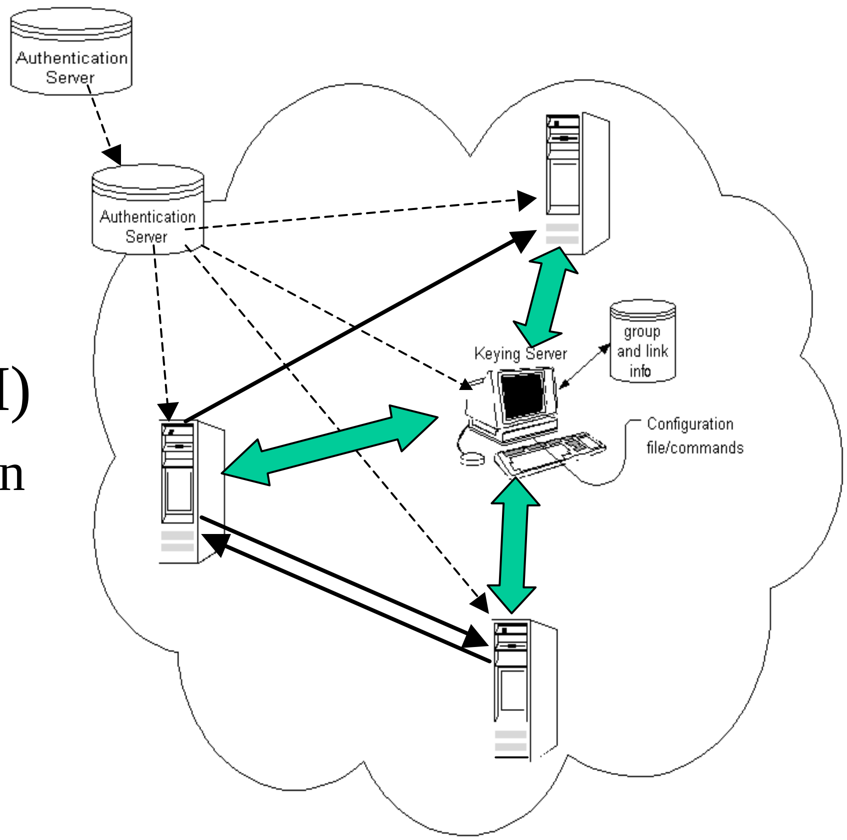
- AH/ESP Protocols
- Components
- Outbound Processing
- Inbound Processing
- IPSEC Modes
  - Transport
  - Tunnel



IP Hdr	IPSec	IP Hdr	Upper	Data	ESP Trailer

Framework Overview

- Three basic components
  - Auth Server (ASV)
    - DNSSEC
  - Keying Server (KSV)
  - Key Mgmt. Module (KMM)
    - Extension of the IKE daemon
- Protocol
  - Node Registration
    - IKE set-up + Authentication
  - IPSec SA Installation



Setting-up DNSSEC

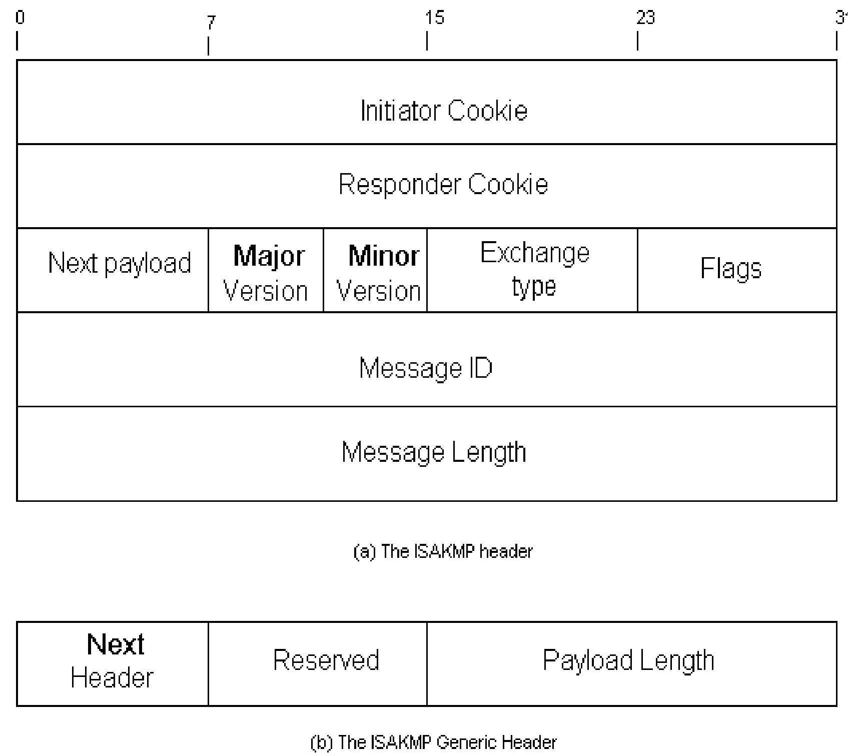
- BIND 9.2
- DNSSEC server
  - Sign the Zone File
  - Send SIG RRs along with the Query Response
- Security Aware Resolver
  - Check the Signatures
    - Configure the trusted-keys
- Applications just have to check the RRSET_VALIDATE flag

Integrating IKE

- FreeSWAN Implementation
  - Enhanced “pluto” to use it as the Keying Server
  - Enhanced the “whack” command-set
    - Add/Delete Link
    - Add/Delete Group
    - Node Register

Information Packaging

- ISAKMP messages
- Payloads
  - 13 distinct types
- Payload Chaining
  - Using the “Next Payload” field
  - Last payload is 0

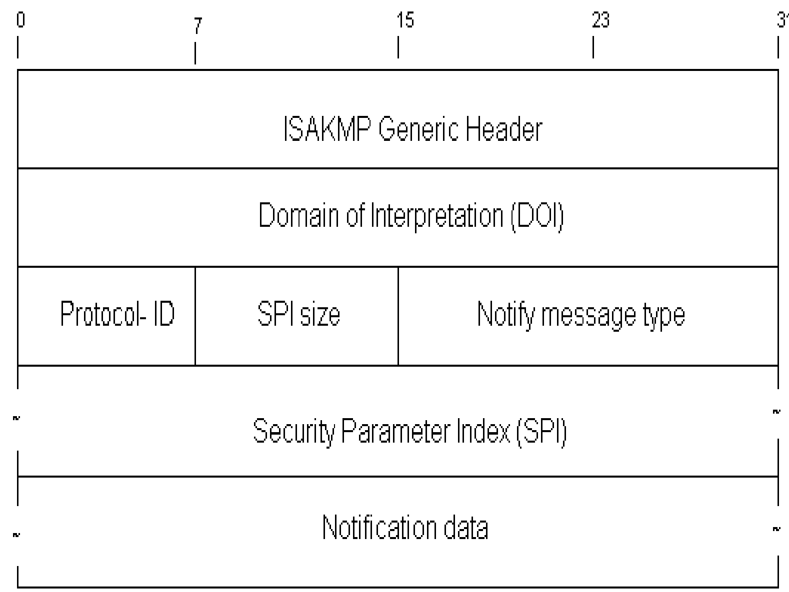


(a) The ISAKMP header

(b) The ISAKMP Generic Header

Notification Message

- Defined during an "Informational Exchange"
- Notification data depends on the Notify message type
- We define 4 new Notify Message types
  - KEYEXCHANGE_REGISTER = 32769
  - KEYEXCHANGE_DELETE = 32770
  - KEYEXCHANGE_ACK = 32768
  - KEYEXCHANGE_ALARM = 32771



KLIPS Processing

- Outbound processing using “eroutes”
  - Every physical interface has its own virtual counterpart
    - e.g. eth0 = ipsec0
- AH and ESP registered as new protocols for inbound processing

Installing Security Associations

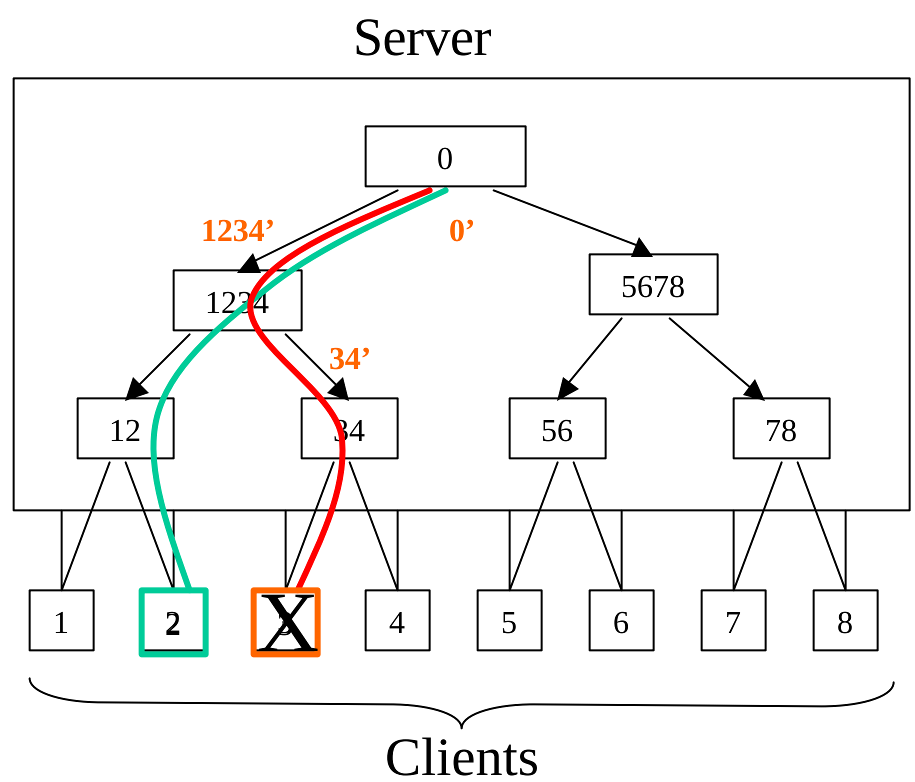
- Multicast considerations
  - SPI cannot be Receiver-Specific anymore
    - Let the Key-Server distribute the SPI values
  - We cannot synchronize the Replay Counters
    - Keep Replay-Protection OFF

Multicast Key Distribution

- Problem areas in Secure Multicast
  - Group Key Management
  - Source Authentication
  - Member Revocation
- We focus on the member revocation problem

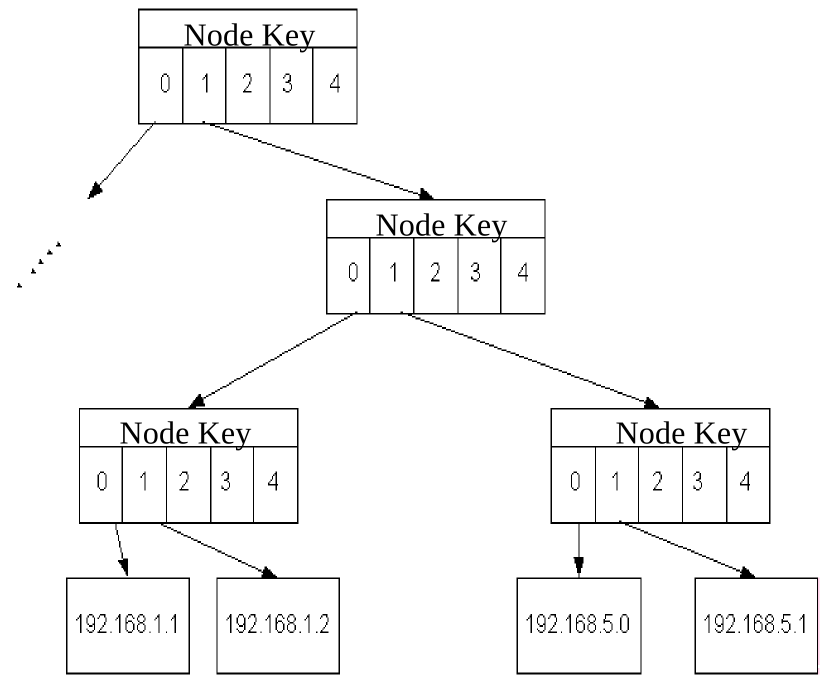
Logical Key Hierarchy (LKH)

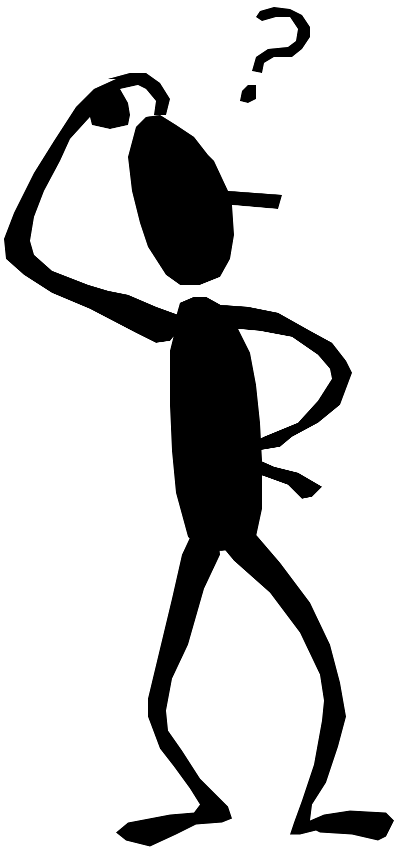
- Secured Removal with Transmission and Storage Efficiency
- No more than one server required
- Benefits
  - Cost of storage and transmissions scale well
  - Subgroups possible
  - Resistant to collusion



Integrating LKH

- LKH Tree Design
  - All members at the leaf
  - We use a B+ tree
- Define a new ID for every multicast group member
- Update-messages are signed using the KSV private key





So… is our Security-Association
“Secure” now ?

NO!

Integrating the Packet Filter

- The Inbound SPD problem
  - Is this IP packet missing any AH/ESP headers?
  - Packets containing no IPSec headers (maybe spoofed??) are still accepted as valid
- IPChains

```
ipchains -A input -d $ME -s $PEER -p 50 -j ACCEPT
ipchains -A input -d $ME -s $PEER -p 51 -j ACCEPT
ipchains -A input -d $ME -s $PEER -i $IF -j DENY
```

- Integrate this with IKE

Sample Topology

POLICY auth

TRANS-TYPE=AH

LIFE-SEC=100

LIFE-KBYTE=10000

END_POLICY

POLICY enc

TRANS-TYPE=ESP

LIFE-SEC=100

LIFE-KBYTE=10000

END_POLICY

GRP 224.0.1.5, POLICY=auth

MEMBERS

testnode2

testnode3

testnode5

testnode6

END_GRP

LINK

testnode1 <=> testnode3, KEY=0:0, POLICY=auth

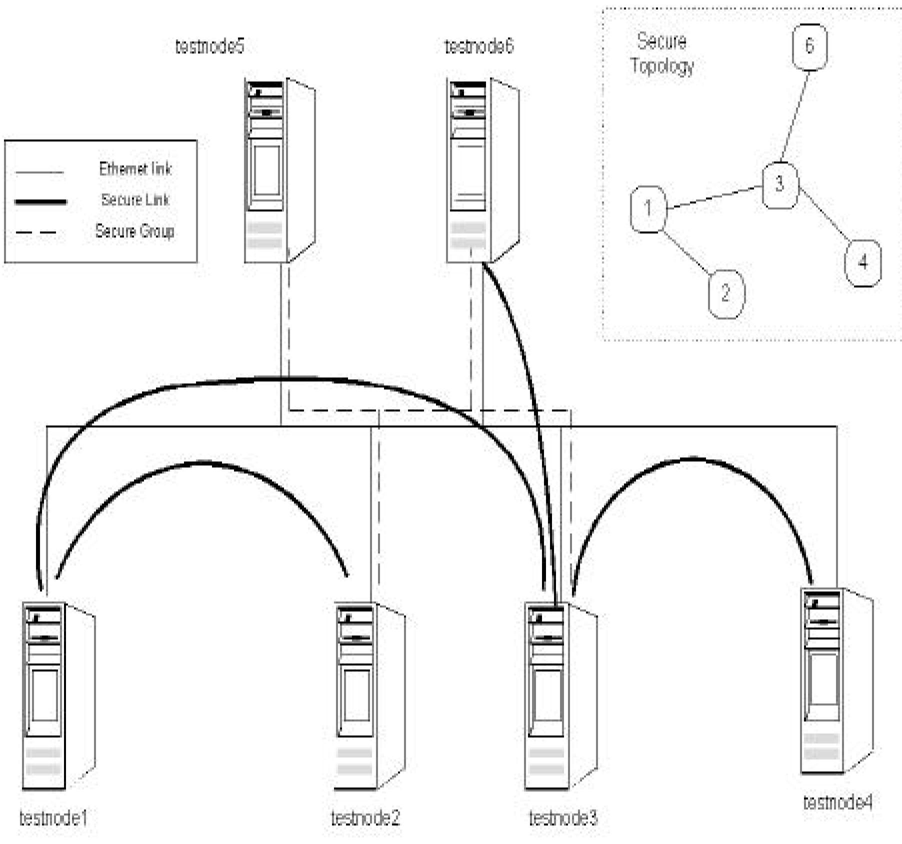
testnode3 <=> testnode4, KEY=0:0, POLICY=auth

testnode1 => testnode2, KEY=0:0, POLICY=auth

testnode2 => testnode1, KEY=0:0, POLICY=enc

testnode3 => testnode6, KEY=0:0, POLICY=auth

END_LINK



Testing

- Testing DNSSEC
- We trust FreeSWAN to provide us reliable implementation of the Security Association
- We check if the receiving application received the packet properly after it was afforded IPSec protection
  - Simple client-server to test reachability
  - Log the packet filter output to check encapsulation
- Testing Secure Multicasts
  - Update Messages sent during Revocation

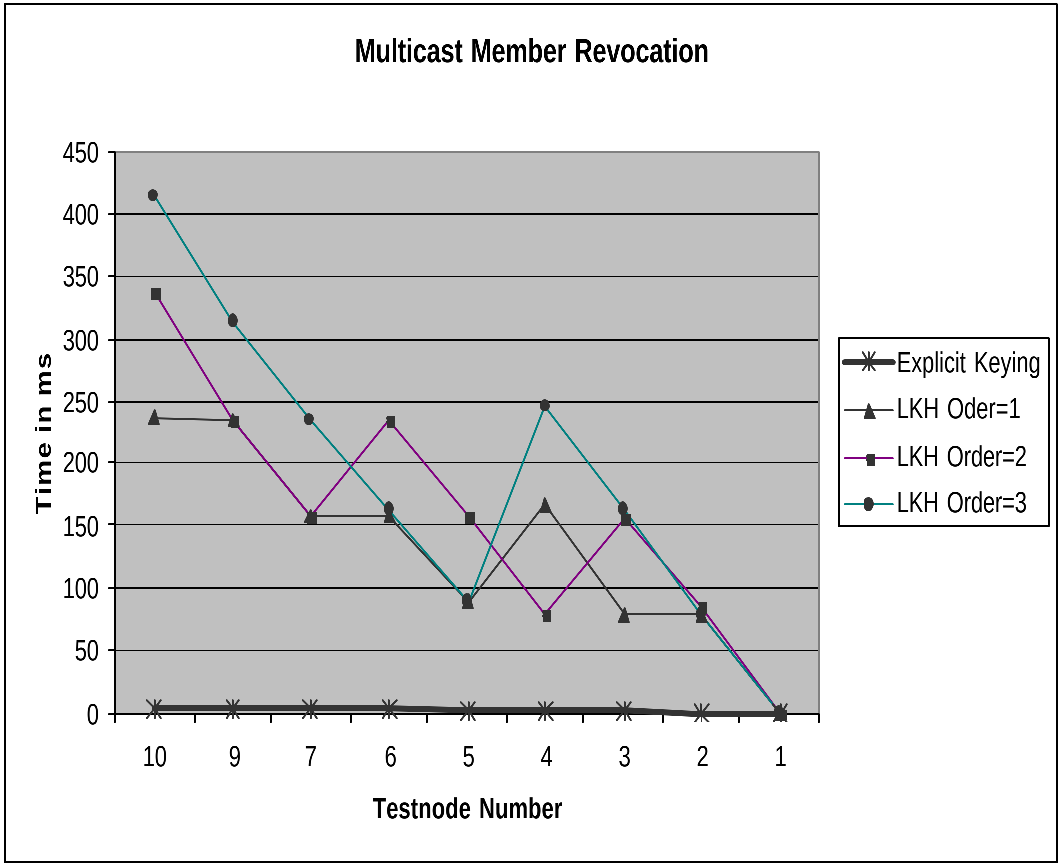
Timing Evaluation

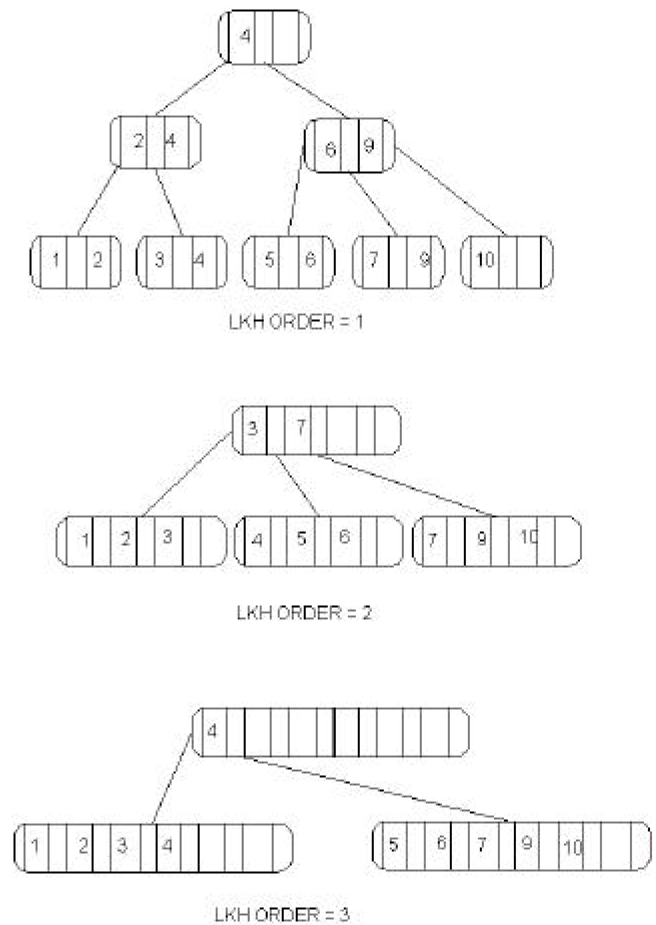
- Some observations are expected
  - DNSSEC Timing ? 2.5 ms
  - Round-Trip Timing
    - Without IPSec ? 0.58 ms
    - With IPSec ? 1.2 ms
- More interesting evaluation is that of the key update channel

Key Update Channel Evaluation

- What do we want to compare ?
- How we performed the test
  - Configure a multicast group containing all members
  - Register all these members to maximize the potential re-key messages
  - Revoke every member one-by-one to trigger any updates
  - Perform the same operation - this time using Explicit Keying
  - Repeat the tests for different orders of the B+ tree

Key Update Channel Evaluation





Summary

- In this thesis we have successfully built in and/or integrated
  - Support for Source Authentication (DNSSEC)
  - Authentication, Integrity, Confidentiality (IPSec)
  - A keying framework (IKE + LKH)

Future Work

- Configuration Mechanism lacks a GUI
- Optimize the multicast key distribution mechanism (LKH+, OFT)
- Extending the framework to work between KSV domains
  - Key management and SA arbitration