Practical Considerations for DNSSEC Automation

Joe Gersch

OARC Presentation
September 24, 2008

SECURE64

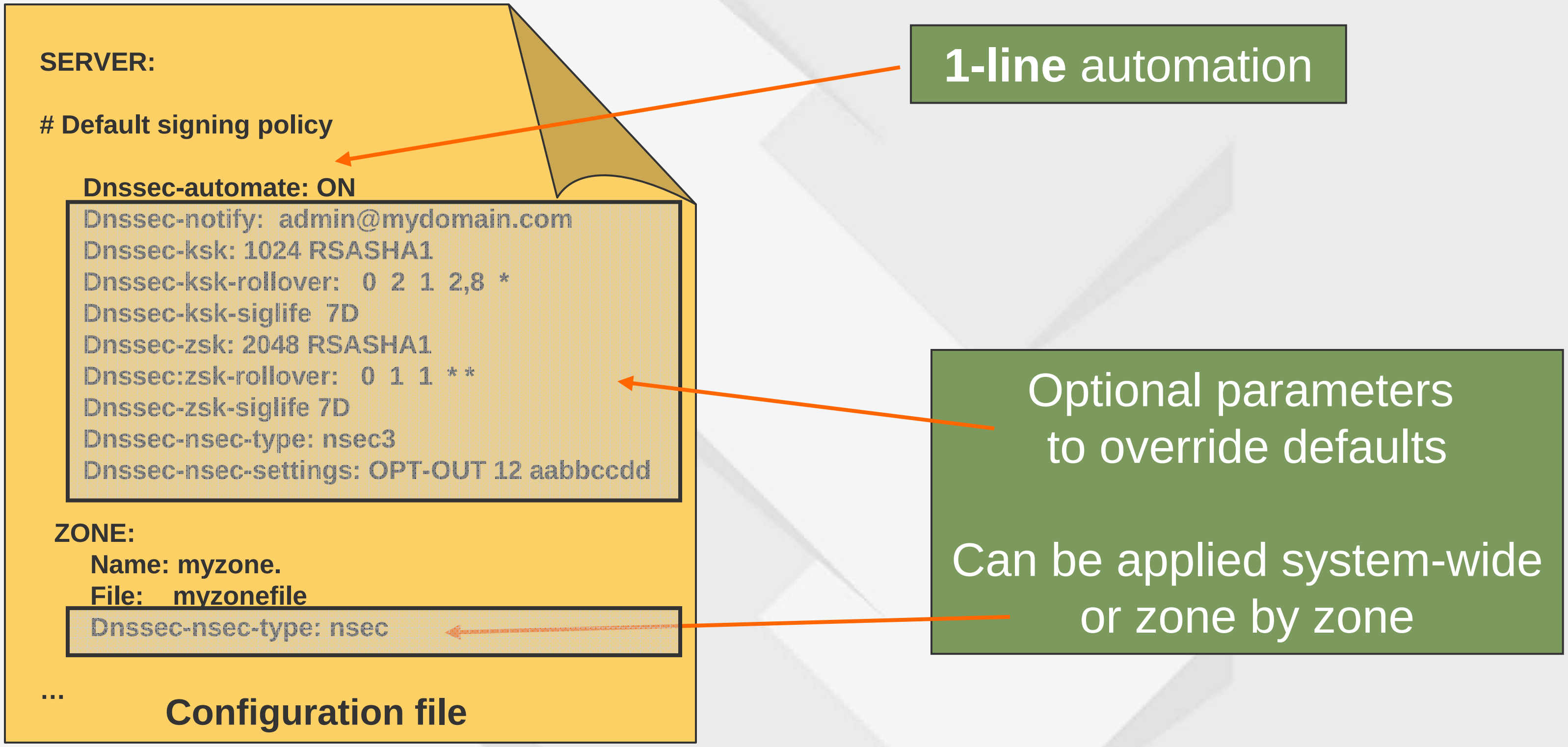
SOFTWARE CORPORATION

The Design Goal:

Secure DNSSEC Automation on a Trusted Computing Platform

- a turnkey *DNSSEC Signer* appliance
  - **Plug-n-Play “DNSSEC-in-a-box”**
  - **Just “set it and forget it”.**
- Built on a secure *Trusted Computing Platform*
  - **Private Signing Keys must be kept safe – they are NEVER in the clear**
  - **The DNSSEC SIGNER runs on an platform designed from the ground up to prevent malware, rootkits and other attacks from compromising the machine.**
  - **FIPS 140-2 certification pending (in testing lab)**

Simple to Configure



```
SERVER:

# Default signing policy

    Dnssec-automate: ON
    Dnssec-notify:  admin@mydomain.com
    Dnssec-ksk: 1024 RSASHA1
    Dnssec-ksk-rollover:   0  2  1  2,8  *
    Dnssec-ksk-siglife  7D
    Dnssec-zsk: 2048 RSASHA1
    Dnssec:zsk-rollover:   0  1  1  * *
    Dnssec-zsk-siglife 7D
    Dnssec-nsec-type: nsec3
    Dnssec-nsec-settings: OPT-OUT 12 aabbccdd

  ZONE:
     Name: myzone.
     File:    myzonefile
     Dnssec-nsec-type: nsec

...
```

Configuration file

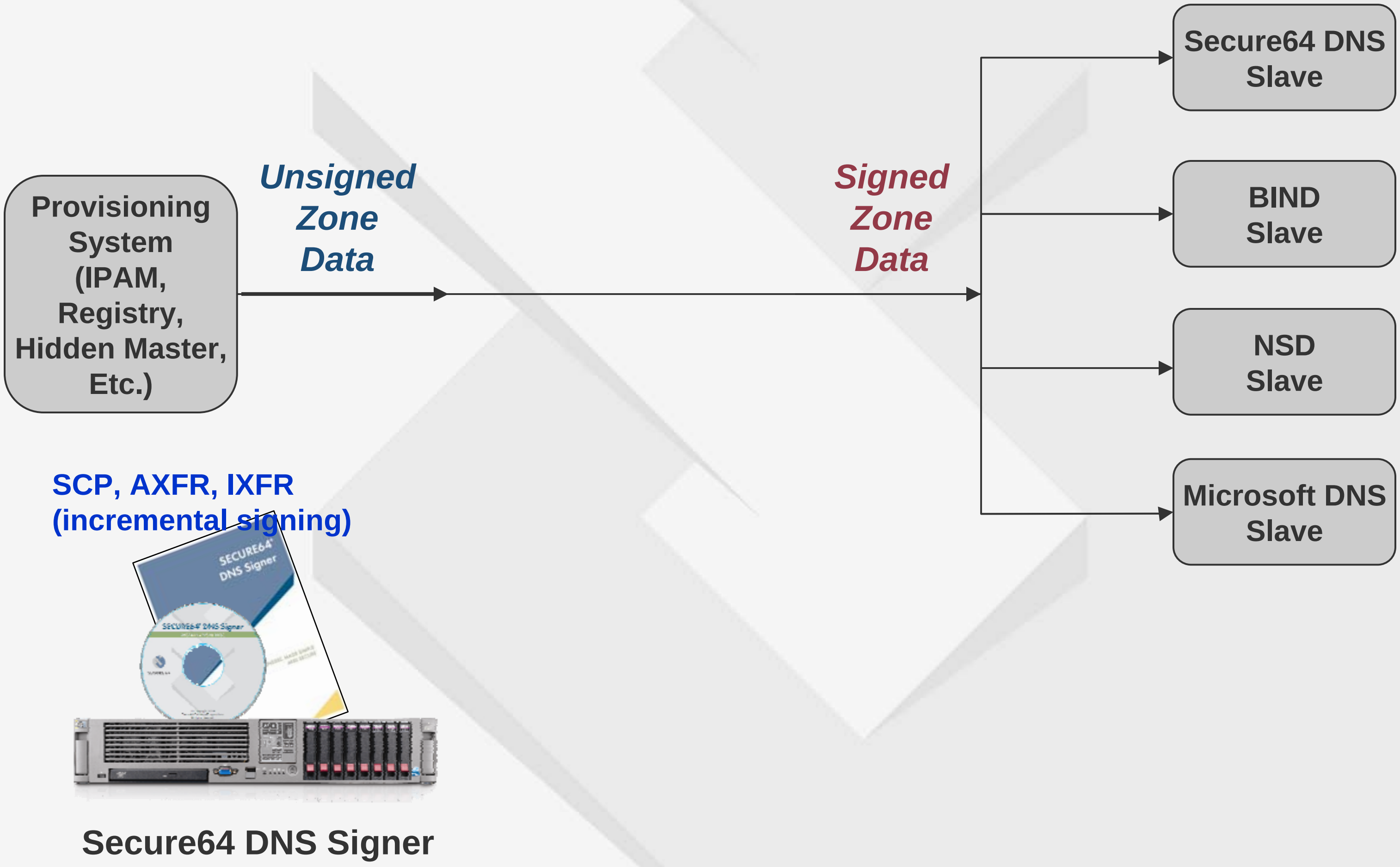
1-line automation

Optional parameters
to override defaults

Can be applied system-wide
or zone by zone

DNSSEC can be deployed in days, not months

Compatible With Current Infrastructure



Provisioning System (IPAM, Registry, Hidden Master, Etc.)

Unsigned Zone Data

SCP, AXFR, IXFR (incremental signing)

Secure64 DNS Signer

Signed Zone Data

Secure64 DNS Slave

BIND Slave

NSD Slave

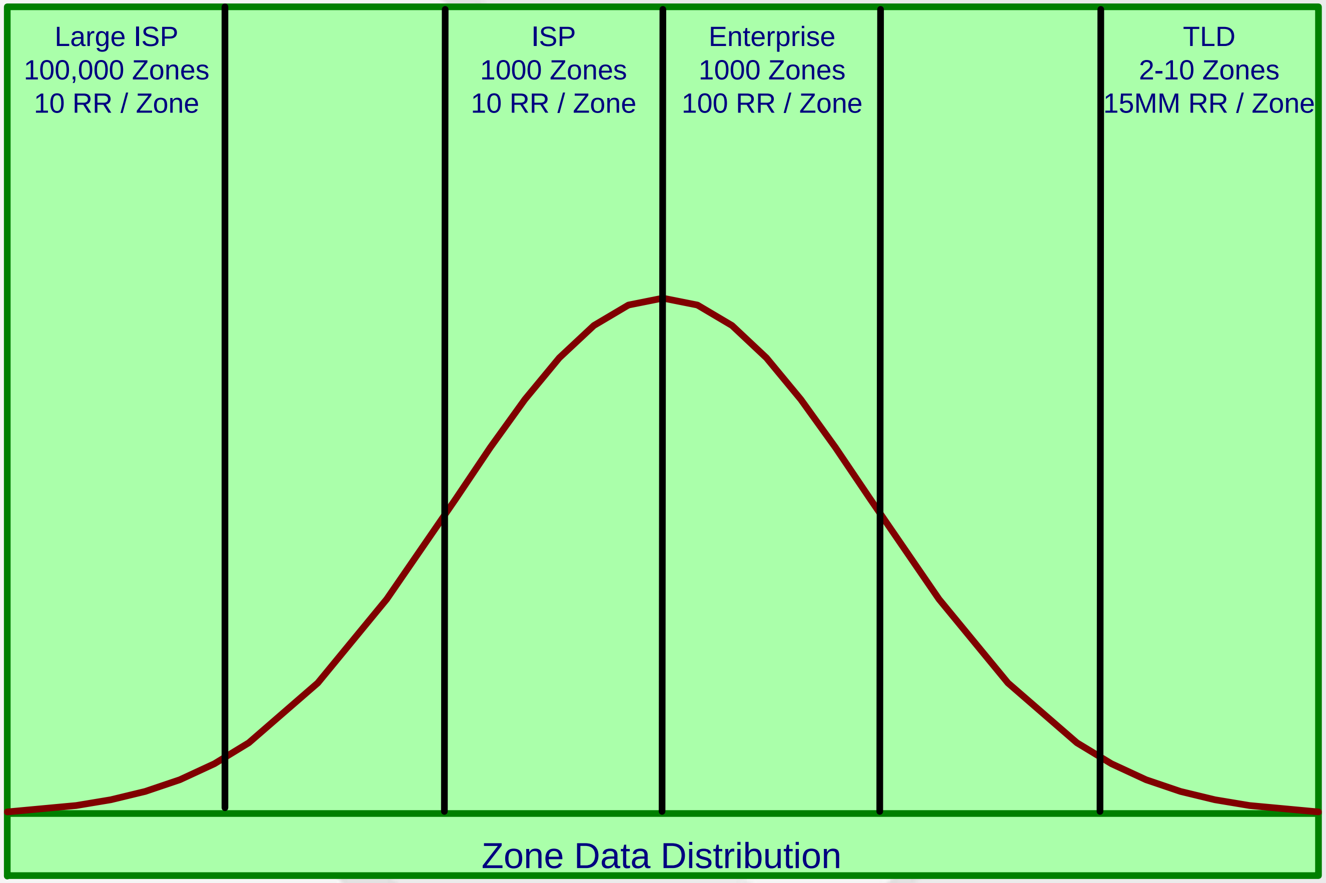
Microsoft DNS Slave

A Few Initial Design Principles

- The provisioning system owns the DNS zone data
  - ➢ Don’t touch the original zone data
- Never permit private keys to be in plaintext
  - ➢ Avoid insider attacks
- Assume the DNS Administrator knows little about DNSSEC, wants to do less, and will make errors
  - ➢ Use *Best Practice* defaults for all parameters
- Manage Errors & Failures:
  - ➢ Backup, fail-over, error detection

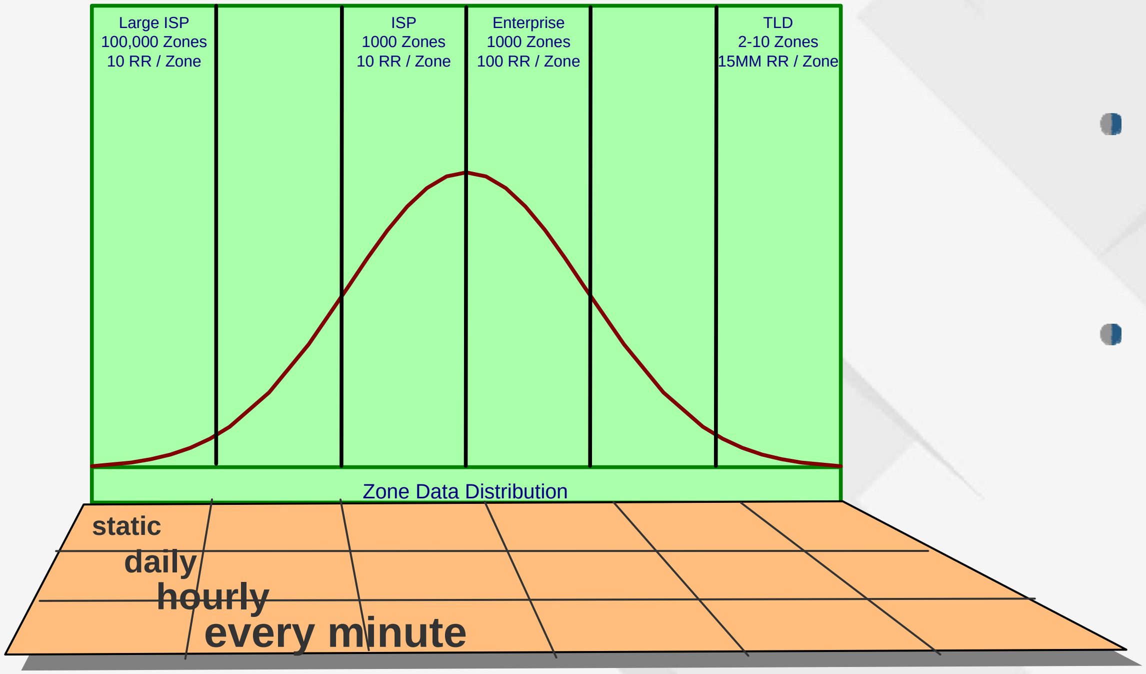
The Reality Check

Designing for the mainstream… Or for 3-sigma out?



Design for the extremes and the normal cases will take care of themselves

Designing for Dynamic Data



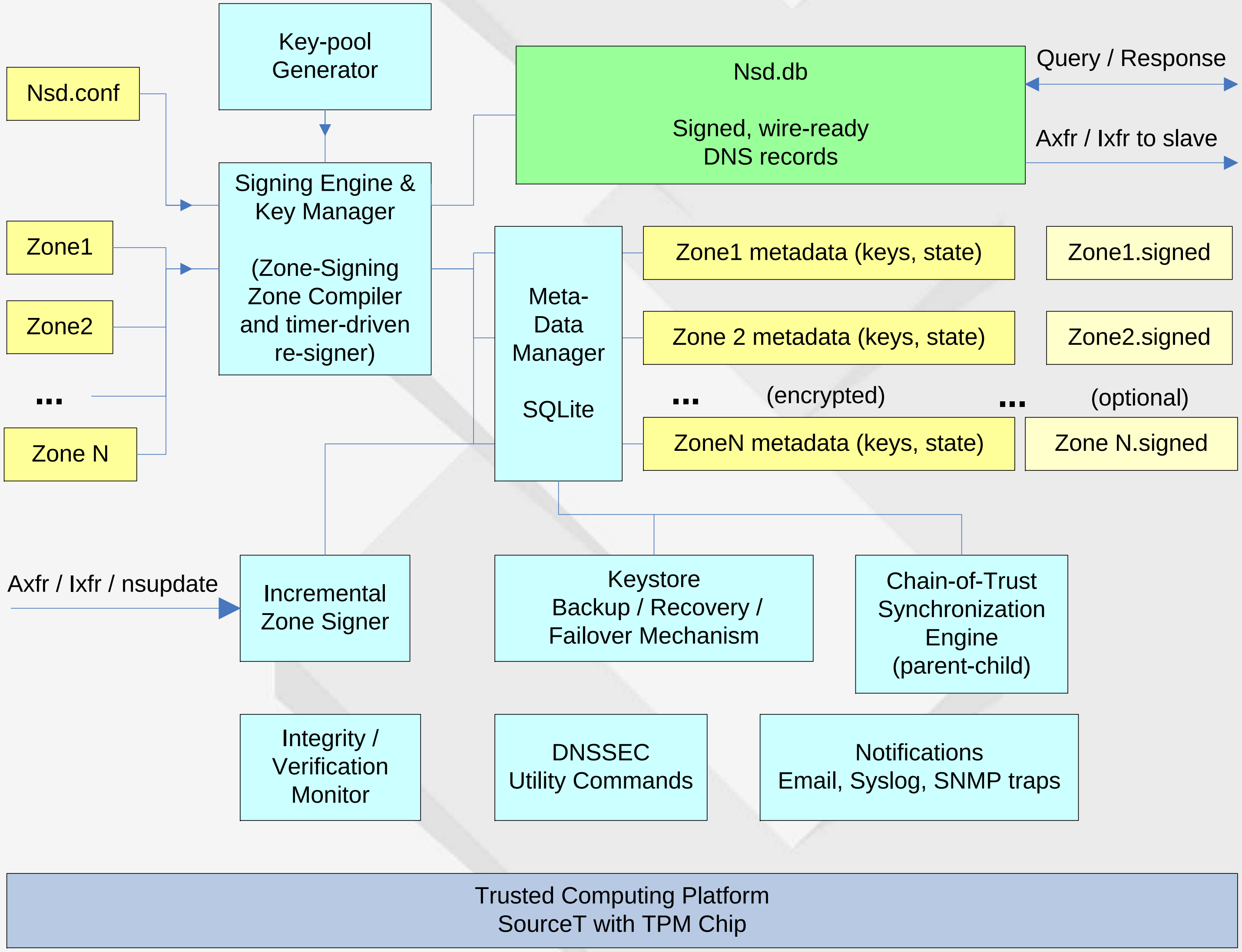
- ISP's & TLD's:
  - new customers result in new delegations
- Enterprise:
  - Active Directory & DHCP
- Example:
  - TLD with millions of RR's
  - Updates every minute
  - How to deal with NSEC3 pre-hash calculation (hint: you don't)

"The need for speed"

Where does this have an Impact?

- Key Generation
  - Potentially an enormous number of keys
  - Real-time or pre-generated in a keypool
- Bulk Signing and Scheduled Re-signing can take lots of time
  - And the duty cycle may be too short
- NSEC3 pre-hash may take too long to calculate
- Metadata Management (including backup & recovery)
  - Private keys
  - Rollover state
  - Serial number management
- Synchronization of Parent-Child DS records and coordination with KSK rollover

System block diagram


Key-pool Generator
Nsd.conf
Nsd.db
Signed, wire-ready DNS records
Query / Response
Axfr / Ixfr to slave
Signing Engine & Key Manager
(Zone-Signing Zone Compiler and timer-driven re-signer)
Zone1
Zone2
...
Zone N
Meta-Data Manager
SQLite
Zone1 metadata (keys, state)
Zone 2 metadata (keys, state)
...
(encrypted)
ZoneN metadata (keys, state)
Zone1.signed
Zone2.signed
...
(optional)
Zone N.signed
Axfr / Ixfr / nsupdate
Incremental Zone Signer
Keystore Backup / Recovery / Failover Mechanism
Chain-of-Trust Synchronization Engine (parent-child)
Integrity / Verification Monitor
DNSSEC Utility Commands
Notifications Email, Syslog, SNMP traps
Trusted Computing Platform
SourceT with TPM Chip

What’s in the MetaData?

- Per Zone Information not contained in nsd.conf
  - Signing Keys – private & public
    - Active, Standby, Revoked
  - Serial #
  - ZSK & KSK state data for rollover
  - parent DS info
  - etc

- nsd.conf specifies attributes such as key length, algorithm, signature life, etc.

Dealing with Serial Numbers

- Remember the prime directive: Don’t mess with the original zone data
  - The provisioning system owns the data and its serial number
- But….
  - An automatic re-signing must increment the serial number in order to issue a NOTIFY to slaves
  - Need to leave room for N automatic increments
  - We will NOT increment if new serial # higher than the serial number saved in our metadata
- Incremental transfers (IXFR from provisioner to the signer) already increment the serial number.

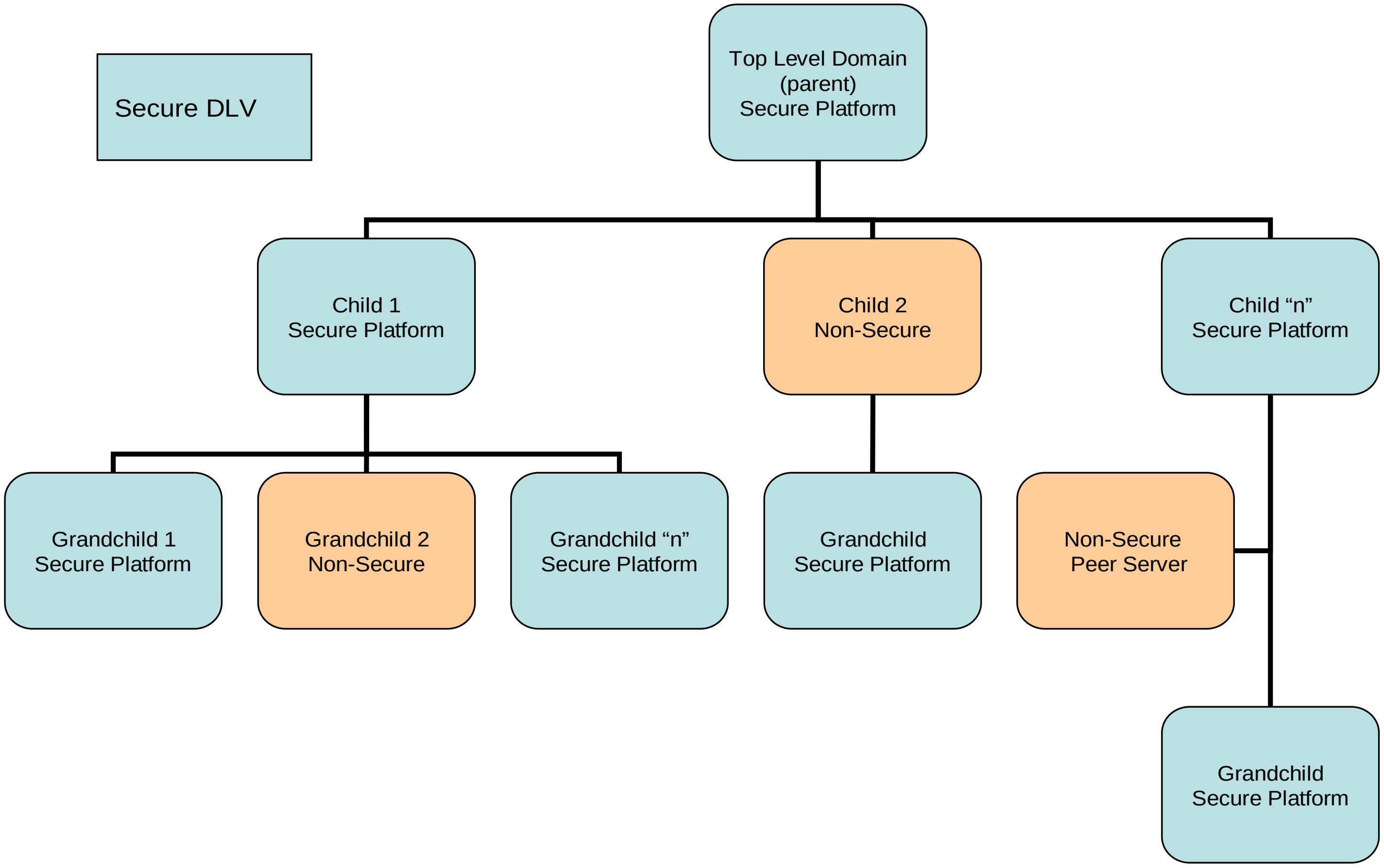
Dealing with Delegation

- Phased Product Rollouts to Improve Parent-Child Synchronization
  - Child polls parent (needed to finish KSK rollover)
  - Parent polls child
  - Integration with
    - EPP
    - IPAM systems
  - Automatic provisioning of DS record when administrator allows

- Publish DS and DNSKEY records
  - send to parent if parent is signed
  - send Trust-Anchors to TAR if parent isn't signed

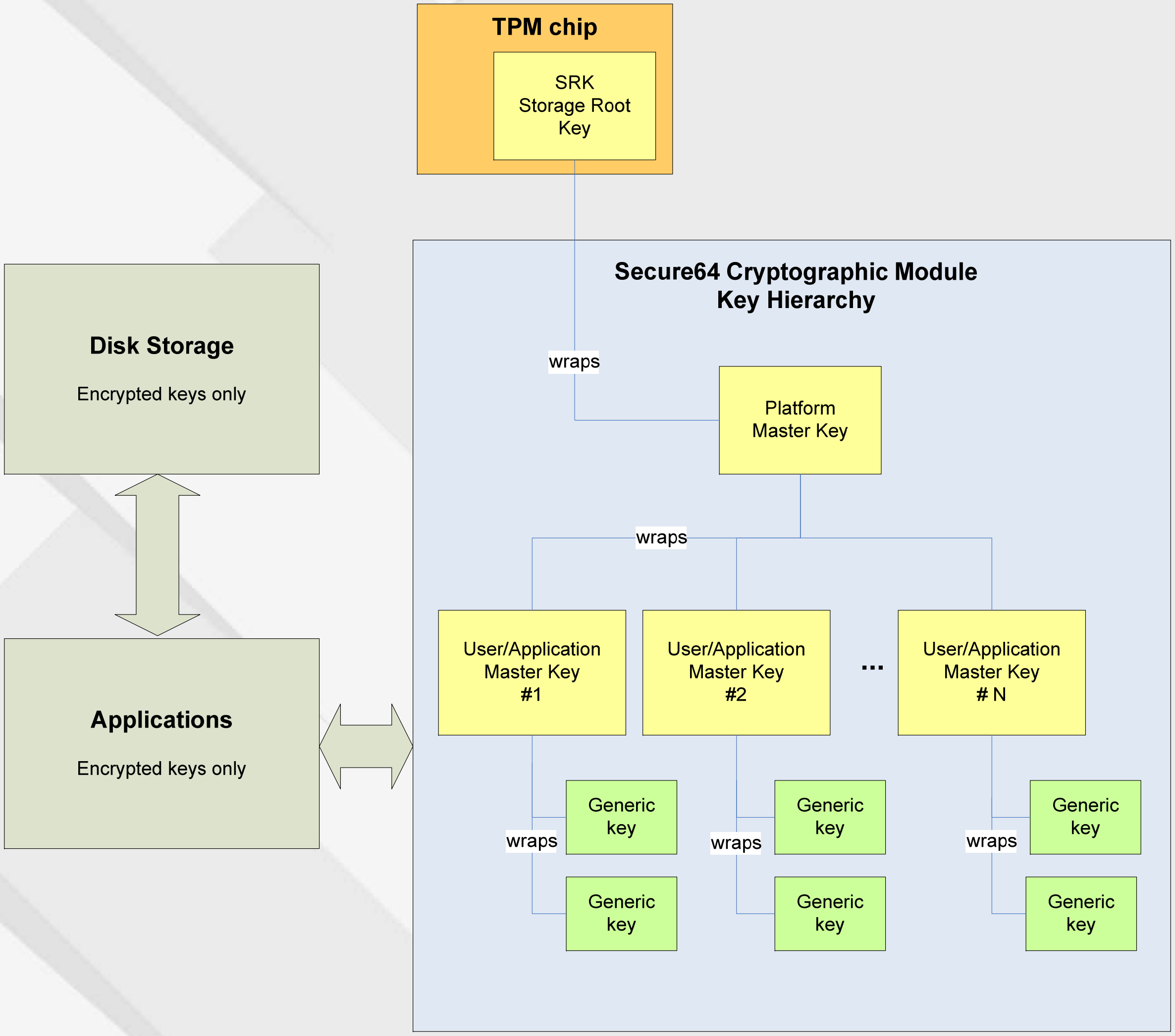
- May help with issues raised on current discussion regarding .MUSEUM
  - Polling parent may find rogue DS record or lame delegation – "Danger, Will Robinson"

Parent-Child Automated DNSSEC Network


Secure DLV
Top Level Domain (parent) Secure Platform
Child 1 Secure Platform
Child 2 Non-Secure
Child “n” Secure Platform
Grandchild 1 Secure Platform
Grandchild 2 Non-Secure
Grandchild “n” Secure Platform
Grandchild Secure Platform
Non-Secure Peer Server
Grandchild Secure Platform

Secure Key Storage & Backup

- TPM migration of master keys
  - Wraps keys to alternate TPM so master keys are never in clear
- Copy encrypted MetaData to backup storage server
  - *Automatic after each re-signing with timestamp*



Secure MetaData Backup & Recovery

Primary Signer

1. One-time TPM migration of master keys
2. synchronized backup of encrypted metadata

Backup Signer

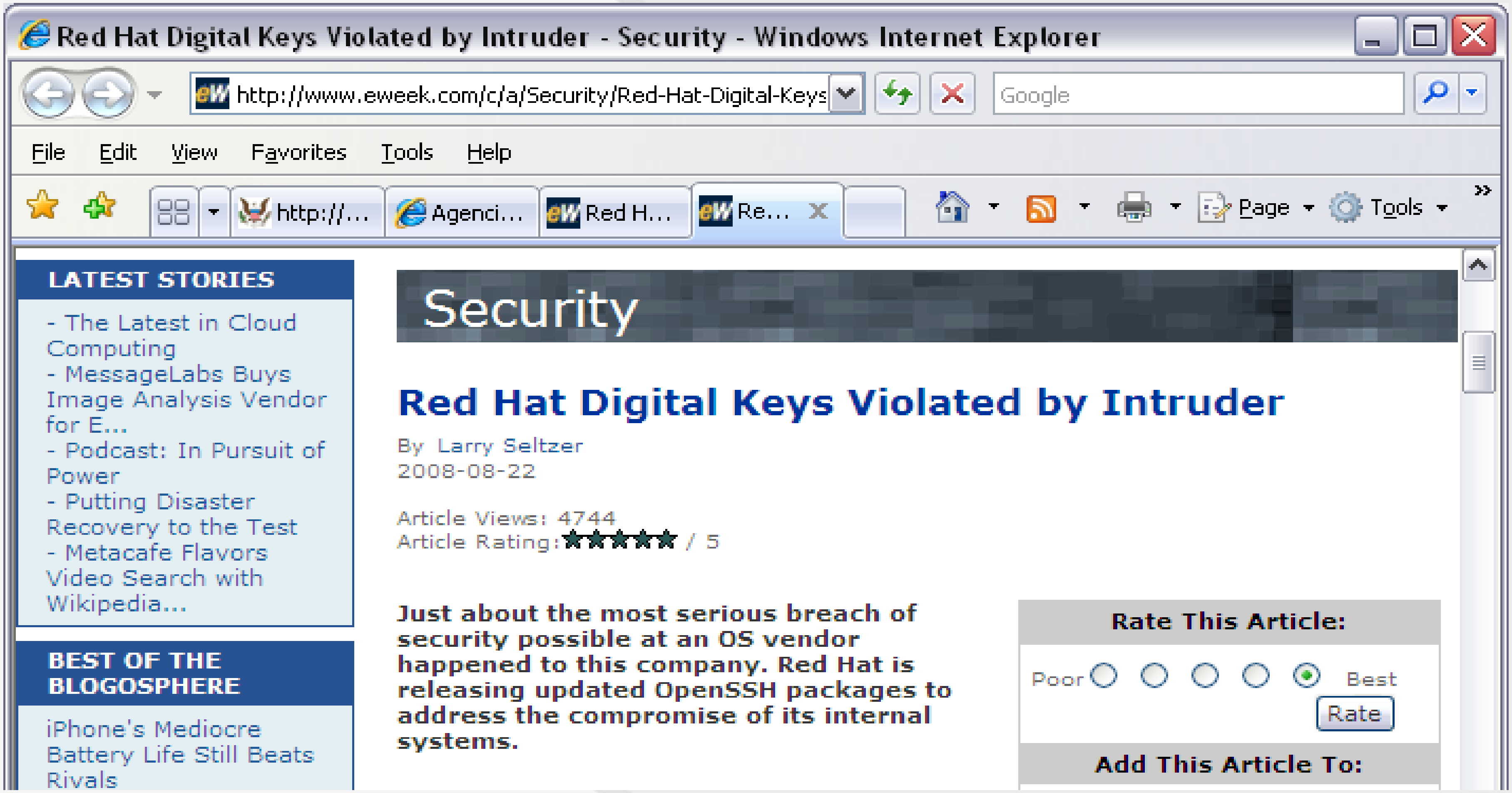
3. Or synchronized backup of encrypted metadata to a storage server

Migration mechanism allows multiple signers owned by different organizations to backup to a community resource

Storage Server

Side Note: Do keys get stolen?

Tuesday, Aug 26 e-week article



Red Hat Digital Keys Violated by Intruder - Security - Windows Internet Explorer

http://www.eweek.com/c/a/Security/Red-Hat-Digital-Keys

LATEST STORIES

- The Latest in Cloud Computing
- MessageLabs Buys Image Analysis Vendor for E...
- Podcast: In Pursuit of Power
- Putting Disaster Recovery to the Test
- Metacafe Flavors Video Search with Wikipedia...

BEST OF THE BLOGOSPHERE

iPhone's Mediocre Battery Life Still Beats Rivals

Security

Red Hat Digital Keys Violated by Intruder

By Larry Seltzer
2008-08-22

Article Views: 4744
Article Rating: ★★★★★ / 5

Just about the most serious breach of security possible at an OS vendor happened to this company. Red Hat is releasing updated OpenSSH packages to address the compromise of its internal systems.

Rate This Article:

Poor ○ ○ ○ ○ ● Best

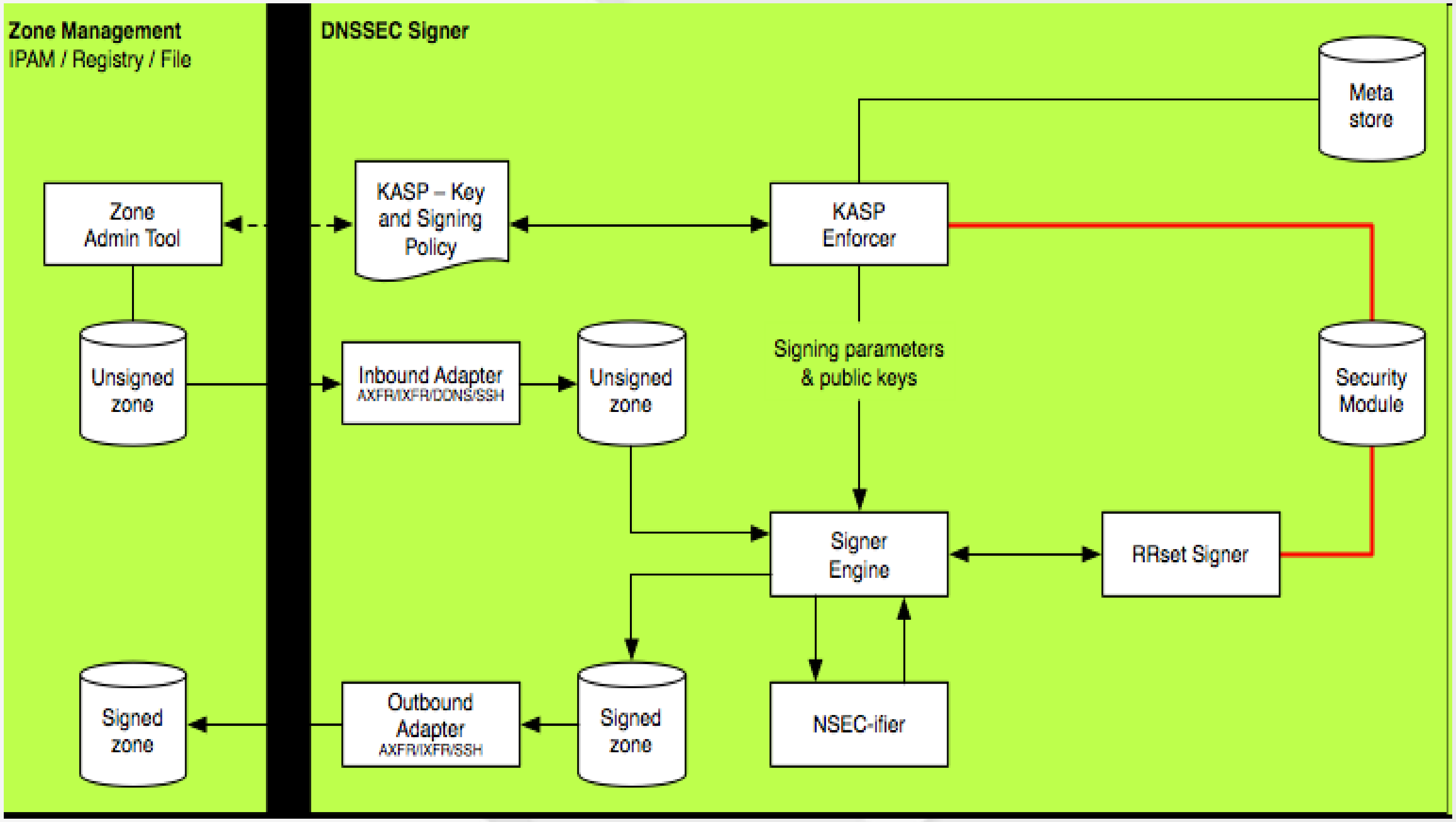
Add This Article To:

SECURE64® DNS Signer
DNSSEC MADE SIMPLE AND SECURE
SECURE64® DNS Signer
INSTALLATION DISC
SECURE64
© Copyright 2008 Secure64 Software Corporation.
DVD

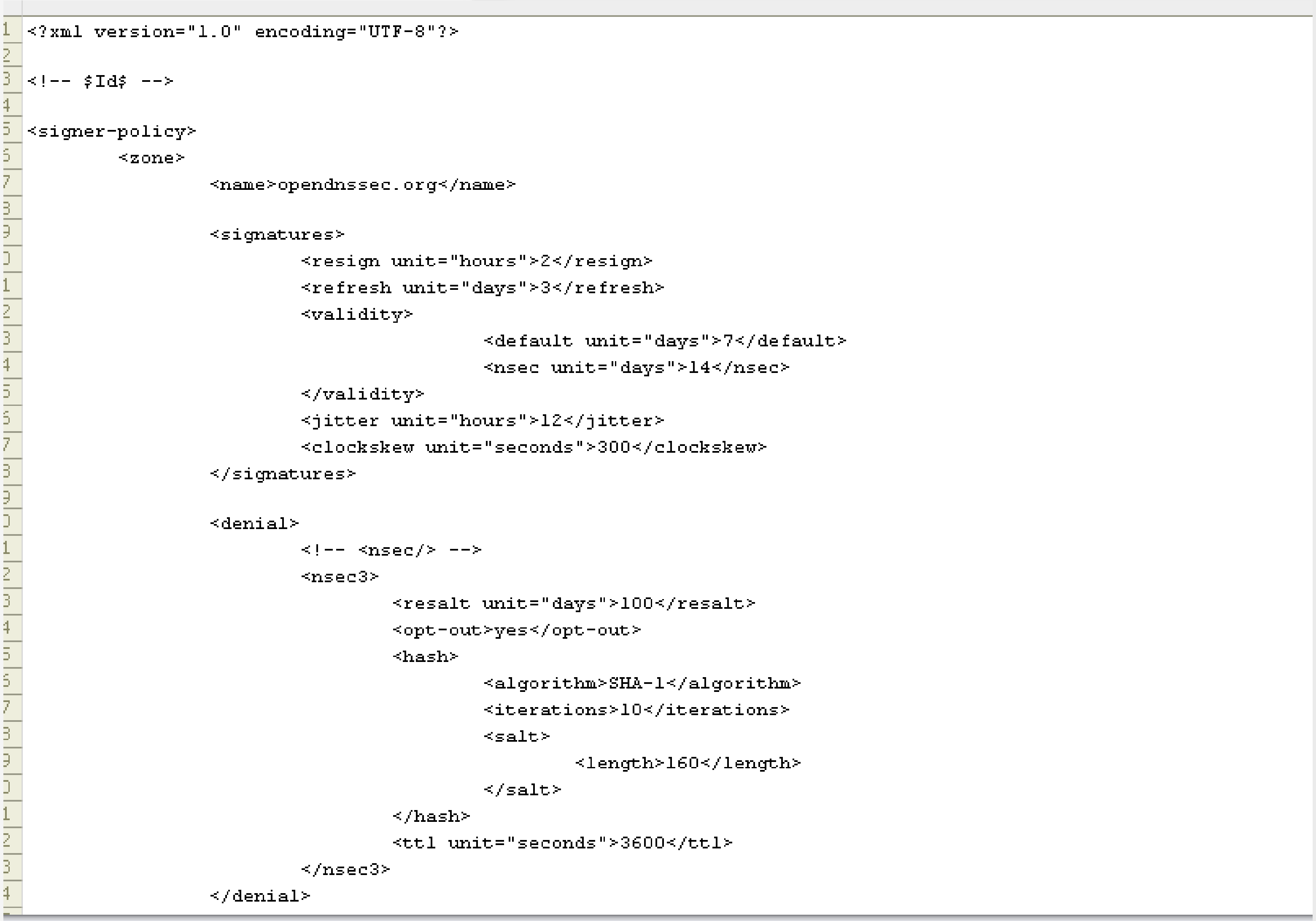
Secure64 DNS Signer

Backup Slides Follow….

OpenDNSSEC architecture with KASP



Eventual integration with XML KASP



```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id$ -->

<signer-policy>
        <zone>
                <name>opendnssec.org</name>

                <signatures>
                        <resign unit="hours">2</resign>
                        <refresh unit="days">3</refresh>
                        <validity>
                                        <default unit="days">7</default>
                                        <nsec unit="days">14</nsec>
                        </validity>
                        <jitter unit="hours">12</jitter>
                        <clockskew unit="seconds">300</clockskew>
                </signatures>

                <denial>
                        <!-- <nsec/> -->
                        <nsec3>
                                <resalt unit="days">100</resalt>
                                <opt-out>yes</opt-out>
                                <hash>
                                        <algorithm>SHA-1</algorithm>
                                        <iterations>10</iterations>
                                        <salt>
                                                <length>160</length>
                                        </salt>
                                </hash>
                                <ttl unit="seconds">3600</ttl>
                        </nsec3>
                </denial>
```