Build Your Own SIE

Oct 22, 2012

Baltimore, MD

Eric Ziegast
<info@sie.isc.org>

Agenda

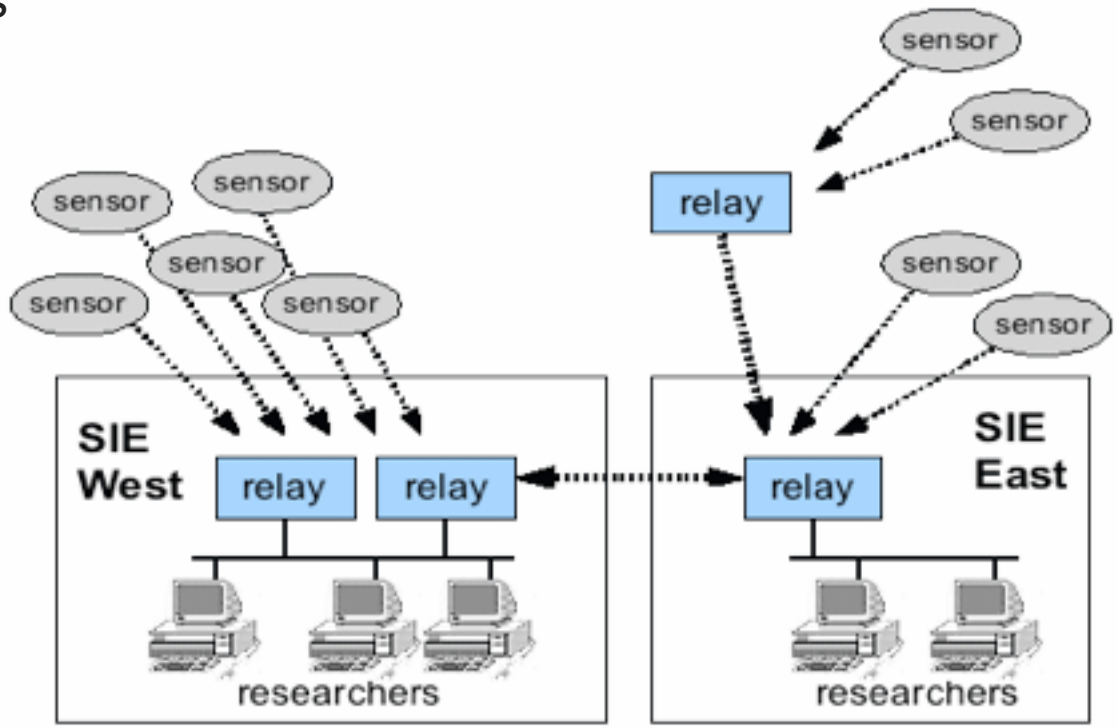
- Limited Scope
  - (internal only - no policy stuff)
- Hardware
- Infrastructure concepts
- nmsgtool
- Q&A

Hardware

- SIE Switch
  - Many will fail over to broadcast traffic
  - Some that work
    - D-Link DGS-3026 (early, had issues)
    - Extreme Networks x450a-24t (production
    - Cisco 3750X (in beta)
- Servers
  - 2 or more 2.0GHz amd64 cores
  - lots of ram
  - Intel server-grade GigE NICs

ISC SIE Infrastructure

- Sensors
- Sensor upload relays
- Inter-node relays
- Participants



Functionality / roles

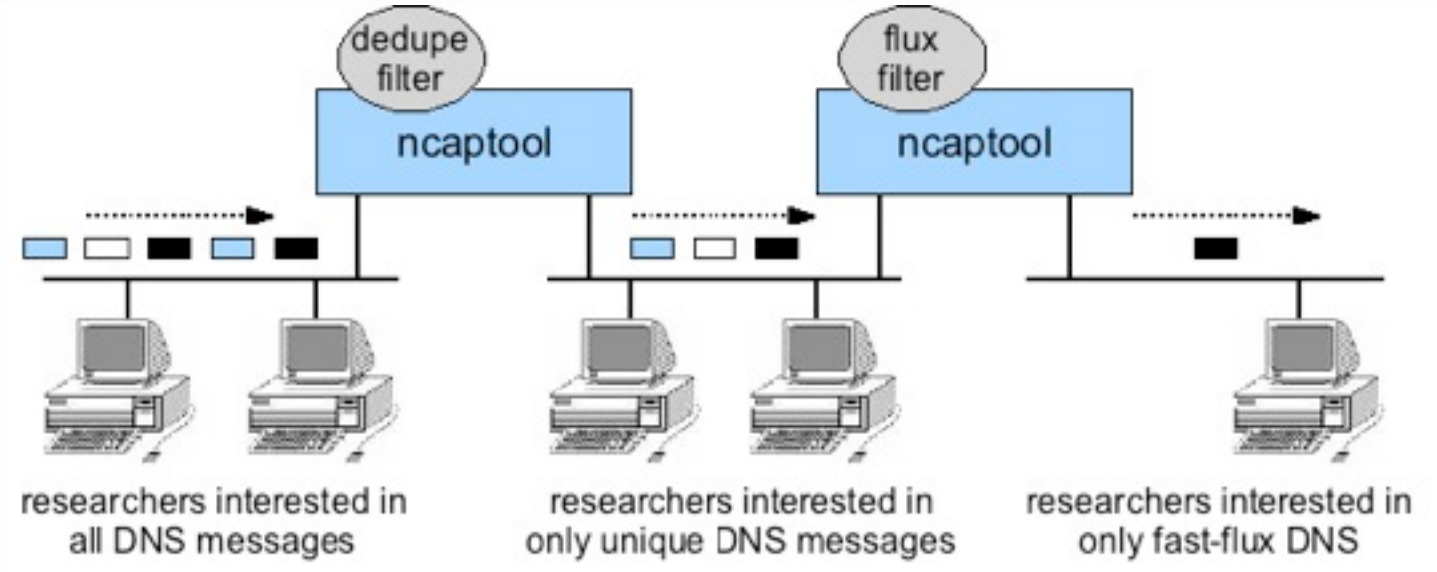
- SIE switch
  - partition VLANs
  - broadcast data
  - trunk to other switches
- Sensor
  - collect data (ethernet tap, stream input)
  - upload:
    - dump to file regularly, rsync, ssh
    - broadcast directly to channel
    - relay over network to another nmsgtool

Functionality / roles

- Inter-node relay
  - Listen to VLAN and rebroadcast to another nmsgtool on remote side (lossy)
  - Dump to file, rsync to other side, replay (no loss, but might delay or clog)
- Participants
  - Listen to data off the wire or broadcast
  - Dump to file for batch processing, or multi-threaded programmng to process in real time
  - Relay into server or off server to remote processing

Functionality / roles

- Participants (properties)
  - Loosely coupled multi-processor



VLANs (Cisco)

```
interface GigabitEthernet1/0/48
 description SPLIT2
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 7,14,25,26,80,81,201-204,206-209
 switchport mode trunk

interface GigabitEthernet1/0/7
 desciption mc7.sie
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 7,14,25,80
 switchport mode trunk

interface GigabitEthernet1/0/8
 description mc8.sie
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 7,14,25,80,202-204,206-208
 switchport mode trunk
```

VLANs (Extreme)

```
create vlan "sie-ch7"
configure vlan sie-ch7 tag 7
...etc...
create vlan "sie-ch209"
configure vlan sie-ch209 tag 209

configure ports 7 display-string mc7.sie
configure ports 8 display-string mc8.sie
configure ports 48 display-string SPLIT2

configure vlan sie-ch7 add ports 7-8, 48 tagged
configure vlan sie-ch14 add ports 7-8, 48 tagged
configure vlan sie-ch25 add ports 7-8, 48 tagged
configure vlan sie-ch26 add ports 48 tagged
configure vlan sie-ch80 add ports 7-8, 48 tagged
configure vlan sie-ch81 add ports 48 tagged
configure vlan sie-ch201 add ports 48 tagged
configure vlan sie-ch202 add ports 7, 48 tagged
...etc...
configure vlan sie-ch208 add ports 7, 48 tagged
configure vlan sie-ch209 add ports 48 tagged
```

VLANs (servers)

```
Linux: ip link add link eth1 name eth1.209 type vlan id 209

       ip link set up eth1 mtu 9000
       vconfig add eth1 209
       ip addr add 10.0.209.18/24 dev eth1.209


  eth1.7    inet addr:10.255.1.18  Bcast:0.0.0.0  Mask:255.255.255.0
  eth1.14   inet addr:10.0.14.18  Bcast:0.0.0.0  Mask:255.255.255.0
  ...etc...
  eth1.209  inet addr:10.0.209.18  Bcast:0.0.0.0  Mask:255.255.255.0


BSD: ifconfig create vlan 209 vlandev em1

  vlan7:   inet 10.255.1.18 netmask 0xffffff00 broadcast 10.255.1.255
           vlan: 7 parent interface: em1
  vlan14:  inet 10.0.14.18 netmask 0xffffff00 broadcast 10.0.14.255
           vlan: 14 parent interface: em1
  vlan209: inet 10.0.209.18 netmask 0xffffff00 broadcast 10.0.209.255
           vlan: 209 parent interface: em1
```

Study our auto-config script:
http://rsfcode.isc.org/git/sie-update/tree/sie-update

nmsgtool

- NMSG
  - read/write from UDP
    - unicast (remote copy)
    - broadcast (SIE switch)
  - read/write from 0mq (Unix or TCP sockets)
  - read/write from files in NMSG binary
  - print presentation output
  - read presentation output

nmsgtool

Receiving messages off the switch:

[-C channel] or --readchan read nmsg data from socket(s)
[-l so] or --readsock read nmsg data from socket (addr/port)

See: (/usr/local)/etc/nmsgtool.chalias
-C ch25 == -l 10.0.25.255/8430 -l 10.0.25.255/9430

[-c count] or --count stop or reopen after count payloads output

Example:

```
nmsgtool -C ch202 -o - -c 5
```

nmsgtool

Capturing data:
[-i if[+][,snap]] or --readif read pcap data from interface ('+' = promisc)
[-p file] or --readpcap read pcap data from file
[-b filter] or --bpf filter pcap inputs with this bpf

[-V vendor] or --vendor vendor
[-T msgtype] or --msgtype message type

Darknet relay example:

```
nmsgtool -V ISC -T pkt -i sie.14+ -m 1280 \
  -s ${REMOTE_SERVER_IP}/50140 -z
```

nmsgtool

```
SIE DNS sensor:

     NMSG_KICKER=/usr/local/lib/sie/sie-kicker ch202
     DNSQR_CAPTURE_RD=0
     DNSQR_RES_ADDRS=149.20.XX.YY, 2001:4f8:ZZ:XX::YY
     ARGV_NMSGTOOL= -i rl0 -V ISC -T dnsqr -z
                -t 60 -w /var/spool/sie/new/ch202/XX.isc.org

     /usr/local/bin/nmsgtool -D -P /var/run/sie_dns_sensor.pid
```

Take a look at scripts included with sie-dns-sensor on rsfcode.isc.org or these:
ftp://ftp.isc.org/isc/nmsg/misc/sie-scripts/

nmsgtool

Cheating to create messages from text input:
[-f file] or --readpres read pres format data from file

[-V vendor] or --vendor vendor
[-T msgtype] or --msgtype message type

Event injection example (old ch21):

```
#!/bin/sh
REASON=$1
NMSGTOOL="nmsgtool -V sie -T reputation -f - \
         --setsource $NMSG_ID -s 10.0.21.255/8430"
while read ip
do
  $NMSGTOOL <<EOF
type: ADDRESS
address: $ip
tag: aa419_ddos_add
value: $REASON

EOF
done
```

nmsgtool

Writing them to files:

[-w file] or --writenmsg write nmsg data to file
[-z] or --zlibout compress nmsg output

[-c count] or --count stop or reopen after count payloads output
[-t secs] or --interval stop or reopen after secs have elapsed
[-k cmd] or --kicker make -c, -t continuous; run cmd on new files

```
Writes files every 15 minutes and processes:

nmsgtool -C ch113 -w /data/ch204 -z -t 900 -k convert2csv.sh

The convert2csv.sh script gets $1 set as file argument
```

nmsgtool

Rebroadcasting them:
[-s so[,r[,f]]] or --writesock write nmsg data to socket (addr/port)
[-m mtu] or --mtu MTU for datagram socket outputs
[--mirror] mirror payloads across data outputs
[--unbuffered] don't buffer writes to outputs

```
Take a file and spit it out to a channel:

 nmsgtool -r FILE -s 10.0.113.255/8430,10000,1000 --unbuffered

Stripe it across multiple USP ports:

 nmsgtool -r FILE --unbuffered \
          -s 10.0.113.255/8430,10000,1000 \
          -s 10.0.113.255/8431,10000,1000

Mirror it to another port:

 nmsgtool -r FILE --unbuffered --mirror \
          -s 10.0.113.255/8430,10000,1000 \
          -s 127.0.0.1/8430
```

nmsgtool

Misc:

[--getsource sonum]	only process payloads with this source value
[--getoperator opname]	only process payloads with this operator value
[--getgroup grname]	only process payloads with this group value
[--setsource sonum]	set payload source to this value
[--setoperator opname]	set payload operator to this value
[--setgroup grname]	set payload group to this value

Relay uploader

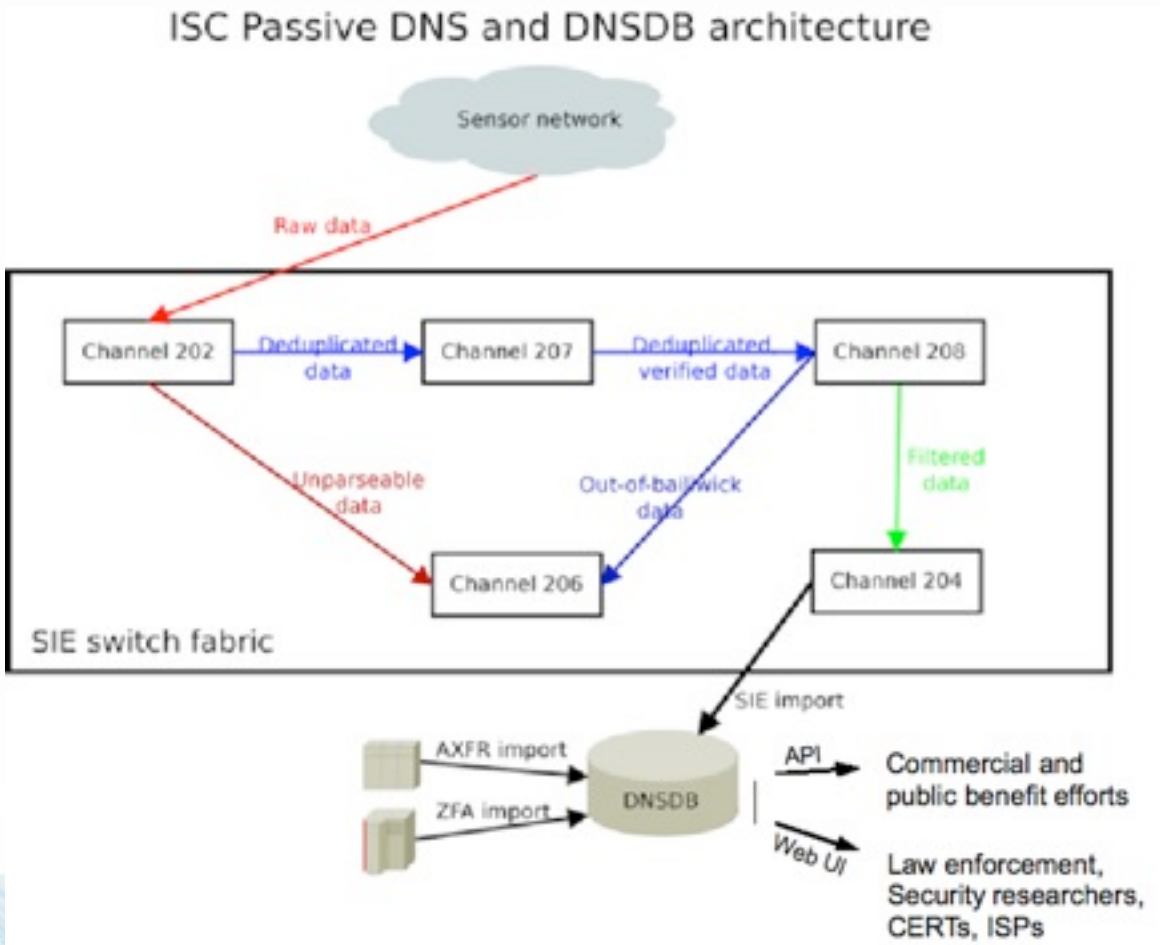
Used to run chroot sshd+rsync or sftp server on FreeBSD.
Maintained custom script to take an upload and queue uploaded FILEs for “nmsgtool -r FILE -s IP/PORT” playback.

Now:
http://rsfcode.isc.org/git/isc-sleigh/

```
isc-sleigh: Debian-based minimal privilege rsync/ssh file service
----------------------------------------------------------------

sleigh is a utility for automatically setting up a multi-user minimal
privilege rsync-over-ssh file submission service on Debian-based systems.

The sleigh Debian package ships a dedicated sshd_config file and runit
service directory. Authentication is public key based, with public keys
stored outside of chroot user home directories in the directory
/etc/sleigh/authorized_keys.d. Individual users and "queues" (per-user
writeable upload directories) are configured with the "sleigh" command line
utility.

sleigh makes use of the Debian libnss-extrausers package in order to avoid
modifying the main /etc/passwd and /etc/group databases, and also depends on
the isc-rsync-static and isc-rsync-server-wrapper packages, which are
available from http://rsfcode.isc.org/.
```

ISC SIE Properties (cont.)

- Loosely Coupled Multi Processor
- Open sourced data - making intermediate products available



Build Your Own Internally

- Get a switch
  - We use Extreme Networks x450a-24t
  - Testing Cisco 3750X
  - Configure VLANs

- Find data
  - nmsgtool sensors
    JSON, XML, sie-dns-sensor
  - darknet
    arp vrf onto switch
  - sinkhole
    httpk null web server
  - netflow
    nfdump + nfreplay to roadcast address

Build Your Own Internally

- Relays
  - batched replay (sie-scripts)
    1. dump data to file
    2. kicker script runs every N seconds
    3. rsync/ssh uploader to relay servers
    4. nmsgtool on relay servers replays data onto VLAN
    5. scripts manage backlogs to prevent overflow

    ftp://ftp.isc.org/isc/nmsg/misc/sie-scripts/sie-scripts-0.21.tar.gz

  - nmsgtool can forward using unicast udp
    - use when data loss is ok

  - nmsgtool can use 0mq for TCP

Build Your Own Internally

- Automation
  - sie-update
    auto-configuring interfaces on participant servers (per-VLAN and per-customer address)
  - channel data replication
  - server build template

- Robustness
  - wrapsrv

Questions?

- Email: info@sie.isc.org
- Web: https://sie.isc.org/
- Eric Ziegast – SIE Programme Manager
  - +1.650.423.1363 (Pacific Time)