OpenLI: What Network Operators Need to Know

Shane Alcock
NZNOG 2019

Agenda

- What OpenLI is and what it is not
- Architectural Overview
- Example integration of OpenLI into a real network (Dave)
- Installing OpenLI packages for Debian / Ubuntu
- Configuring OpenLI
- Running the OpenLI services
- Checking that OpenLI is working
- Gotchas to be wary of when using OpenLI
- Installing and running OpenLI from source
- Installing and configuring DPDK
- Using DPDK with OpenLI
- Future development of OpenLI
- Q & A, discussions, etc.

What is OpenLI?

- Software that can help operators meet LI obligations
  - Written in NZ, specifically for TICSA
  - ETSI format is widely used elsewhere, though

openLI

What is OpenLI?

- Open source, licensed under GPLv3
  - Free, even for commercial use
  - Source code is available, modify as needed
  - Written in C
  - On GitHub

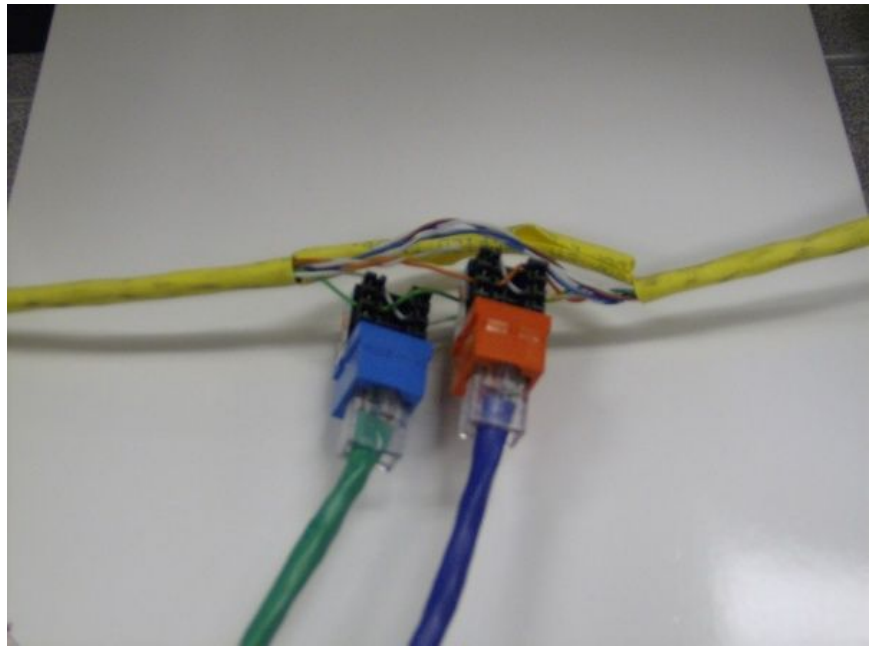
https://github.com/wanduow/openli

What is OpenLI?

- Designed to run on x86 Linux boxes
  - Install OpenLI boxes alongside your regular kit
  - Choose your own hardware
  - Affordable!

What is OpenLI?

- Entirely passive
  - Mirror interceptable traffic into OpenLI boxes
    - This is *your* responsibility!



What OpenLI is NOT

- An automatic solution to all your TICSA problems!
  - Some integration effort is required
  - Testing with NZP will still be necessary

What OpenLI is NOT

- Capable of configuring other devices in your network
  - You are in charge of mirroring traffic to OpenLI
  - Some mirrors may need to be dynamic
    - Only mirror traffic for intercept targets

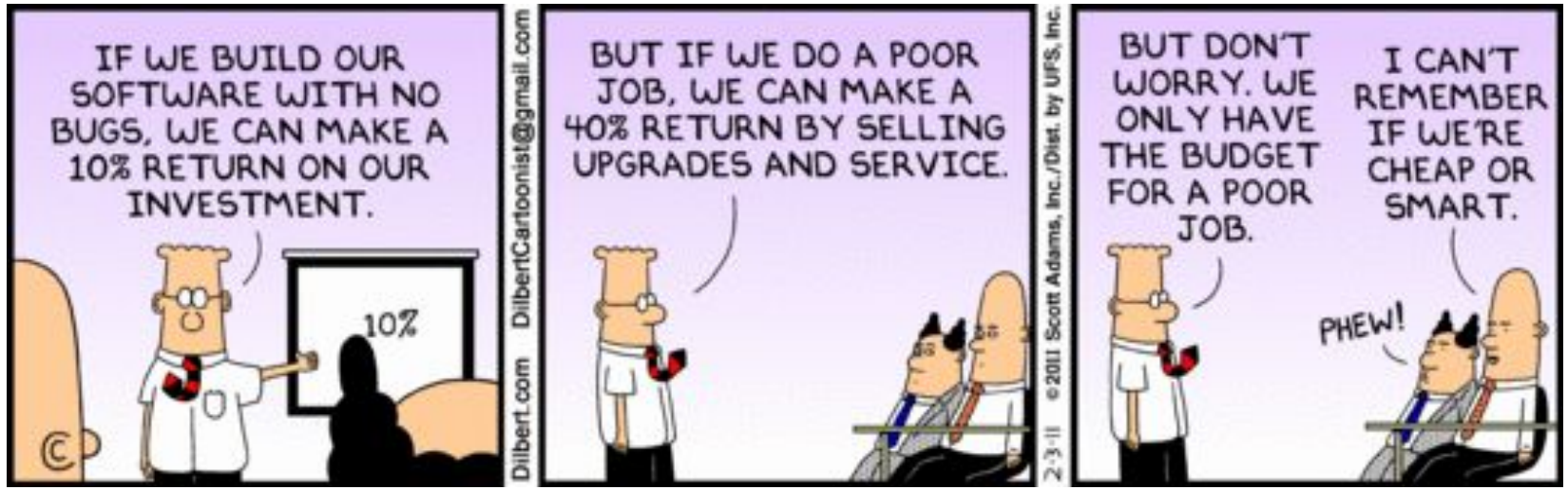
- Secure communication with the LEAs
  - An IPSec tunnel to NZ Police == your job

What OpenLI is NOT

- An opportunity to turn your LI into my problem
  - There's no 24/7 helpdesk for OpenLI
  - You need to be able to troubleshoot OpenLI
  - I'm happy to help out, but it will be "best-effort"
  - You get what you pay for

What OpenLI is NOT

- Perfect (yet?)
  - It's still early days
  - There will be bugs, incompatibilities, etc.
  - Be patient
  - Be nice to me and I'll be good to you ;)



Overview of the OpenLI Architecture

Overview of the OpenLI Architecture

Warrant

Config

OpenLI Provisioner

Forwarding Instructions

Intercept Instructions

OpenLI Mediator

OpenLI Collector

Packets

Interception Point

Overview of the OpenLI Architecture

Warrant

Config

OpenLI Provisioner

Intercept Instructions

Forwarding Instructions

Encoded Packets

OpenLI Mediator

Encoded Packets

OpenLI Collector

Packets

Interception Point

Overview of the OpenLI Architecture

Warrant

Config

OpenLI Provisioner

Encoded Packets

Forwarding Instructions

Intercept Instructions

OpenLI Mediator

OpenLI Collector

OpenLI Collector

Encoded Packets

Provisioner

- Centralised management of OpenLI components
  - Sole interface for interacting with OpenLI

- Low workload
  - Could run on a VM or alongside other services

- Currently configured by hand
  - Use signals to reload modified configuration

Mediator

- Forward encoded IRIs and CCs to LEAs
  - Only component that needs external access
  - All outgoing comms over an encrypted tunnel

- Connections to LEAs are always up and maintained
  - Keep alives (both TCP and application level)

- Multiple mediators are supported
  - Most deployments only need one

Collector

- Workhorse component
  - One per interception location (e.g. BNG)

- Mirror your interceptable traffic into collectors
  - Can read from multiple interfaces

- Internals are quite complex
  - A lot of parallelism going on
  - Session tracking for IP customers, VOIP calls
  - Spread workload to avoid bottlenecks

Collector

- Collector hardware
  - Each collector should be a dedicated x86 box
  - Linux (Debian recommended)
  - CPU: 8 virtual cores (more is definitely better)
  - RAM: at least 16 GB
  - NICs:
    - 10G DPDK-compatible card for IP intercept
    - 1 interface reserved for management
    - Additional 1G ports for RADIUS, SIP, RTP, etc.

Collector

- Dave's collector hardware: Lenovo x3550 M5

x3550 M5 Base Chassis 8x2.5(Standard)
Intel Xeon Processor E5-2609 v4 **8C** 1.7GHz 20MB Cache 1866MHz 85W (Standard)
16GB TruDDR4 Memory (2Rx4, 1.2V) PC4-19200 CL17 2400MHz LP RDIMM (Standard)
System x3550 M5 4x 2.5" HS HDD Kit (Standard)
ServeRAID M1215 SAS/SATA Controller(Standard)
System x3550 M5 PCIe Riser 1 (1x ML2 x16 CPU0)
Intel X710-DA2 ML2 2x10GbE SFP+ Adapter
System x 900W High Efficiency Platinum AC Power Supply
4 onboard GE NICs using Broadcom NetXtreme GE controllers.

Collector

- Traffic to mirror into a collector
  - IP customer traffic
    - Preferably just traffic for intercept targets!
  - RADIUS traffic for all customers at the location
    - Static IP assignment is a special case
  - SIP traffic for all VOIP customers
  - RTP traffic for all VOIP customers

Mirroring Options

- Buy a flow tap license from your BNG vendor
  - Easiest (?) but also most expensive

- Modify target's traffic flow using a RADIUS CoA (or similar)
  - Only do this for target IP traffic

- Customise your BNG in some other way

Hands On (optional)

- Virtual machines
  - openli-vm-blank-mediator
  - openli-vm-blank-collector
  - Login username: `openli` password: `tutorial`

- Dockerfiles
  - Download from https://bit.ly/2LS5WZD

Hands On (optional)

- VMs will require some network configuration
  - Two interfaces: one NAT, one “internal” network
  - Edit `/etc/network/interfaces`
    - First interface should be “auto <name> inet dhcp”
    - Second interface should be set to a static IP
    - Static IPs for both VMs should be on the same subnet
  - Restart networking
    - `sudo service networking restart`

Hands On (optional)

- Docker users
  - README should contain all the instructions you need...

OpenLI Packages

- Packaged for Debian / Ubuntu
  - Packages are “unofficial”, hosted by WAND and Bintray
  - Support stable / oldstable releases

- Highly recommended
  - Installing and updating will be much easier

OpenLI Packages

- Adding the WAND and OpenLI bintray repos to your apt sources

```
sudo apt-get install apt-transport-https curl

echo "deb https://dl.bintray.com/wand/general $(lsb_release -sc) \
  main" | sudo tee -a /etc/apt/sources.list.d/wand.list

echo "deb https://dl.bintray.com/wand/libtrace $(lsb_release -sc) \
  main" | sudo tee -a /etc/apt/sources.list.d/wand.list

echo "deb https://dl.bintray.com/wand/OpenLI $(lsb_release -sc) \
  main" | sudo tee -a /etc/apt/sources.list.d/wand.list

sudo curl https://packages.wand.net.nz/keyring.gpg -o \
  /etc/apt/trusted.gpg.d/wand.gpg

sudo apt-get update
```

OpenLI Packages

- Adding the WAND and OpenLI bintray repos to your apt sources

https://bit.ly/2RT1M97

OpenLI Packages

- Installing the provisioner

```
sudo apt-get install openli-provisioner
```

OpenLI Packages

- Installing the collector

```
sudo apt-get install openli-collector
```

OpenLI Packages

- Installing the mediator

```
sudo apt-get install openli-mediator
```

Configuring OpenLI

- Configuration files for OpenLI (packaged version)

```
/etc/openli/provisioner-config.yaml

/etc/openli/collector-config.yaml

/etc/openli/mediator-config.yaml
```

Configuring OpenLI

- Configuration format is YAML

```
# comments look like this
key: value

list_of_items:
 - listitem1
 - listitem2
 - listitem3

list_of_complex_items:
 - key_a: value_a_for_item1
   key_b: value_b_for_item1
   key_c: value_c_for_item1

 - key_a: value_a_for_item2
   key_b: value_b_for_item2
   key_c: value_c_for_item2
```

Configuring OpenLI

- Starting with the provisioner
  - `/etc/openli/provisioner-example.config`

https://bit.ly/2Pj4mA5

Configuring OpenLI

```
# Listen for incoming collectors on 10.0.0.1:9001
clientaddr: 10.0.0.1
clientport: 9001

# Listen for incoming mediators on 10.0.0.1:12001
mediationaddr: 10.0.0.1
mediationport: 12001
```

Configuring OpenLI

```
# List of SIP servers on our network
sipservers:
  - ip: 192.168.110.100
    port: 5060

# List of RADIUS servers on our network
# NOTE that we have an entry for both Auth and Accounting
radiusservers:
  - ip: 10.199.0.253
    port: 1812
  - ip: 10.199.0.253
    port: 1813
```

Configuring OpenLI

```
# List the set of law enforcement agencies
agencies:
 - agencyid: "Police"
   hi2address: 192.168.200.1
   hi2port: 35530
   hi3address: 192.168.200.1
   hi3port: 35531
   keepalivefreq: 300
   keepalivewait: 30
```

Configuring OpenLI

```
# List of active VOIP intercepts
voipintercepts:

  - liid: 1A23RD112
    authcountrycode: NZ
    deliverycountrycode: NZ
    mediator: 6001
    agencyid: "Police"
    siptargets:
      - username: 078384466
        realm: myisp.co.nz

      - username: xyzzy
```

Configuring OpenLI

```
# List of active IP intercepts
ipintercepts:

  # Target identified by username
  - liid: LP45NNST3
    authcountrycode: NZ
    deliverycountrycode: NZ
    mediator: 6001
    agencyid: "Police"

    user: "shanealcock"
    accesstype: "fiber"
```

Configuring OpenLI

```
# Target identified by static IP ranges(s)
- liid: XP45NN710T
  authcountrycode: NZ
  deliverycountrycode: NZ
  mediator: 6001
  agencyid: "Police"
  accesstype: "LAN"

  staticips:
    - iprange: "192.168.10.0/26"
      sessionid: 7
    - iprange: "2001:db8::/56"
      sessionid: 7
```

Configuring OpenLI

- Supported accesstype values (case insensitive)
  - Core types are defined by the ETSI spec

```
"dialup"
"adsl", "vdsl"
"cable"
"lan", "ethernet"
"wireless"
"fiber", "ufb"
"wimax"
"satellite"
"wireless-other"
```

Configuring OpenLI

- Let’s configure a “real” provisioner...

Configuring OpenLI

- Now let's move on to the mediator
  - `/etc/openli/mediator-example.config`

https://bit.ly/2ExtaTU

Configuring OpenLI

```
# Unique ID string for my network (16 chars max)
operatorid: WAND

# Set a unique ID for this mediator
mediatorid: 6001
```

Configuring OpenLI

```
# How to connect to the provisioner
provisioneraddr: 10.0.0.1
provisionerport: 12001

# Listen for collectors on 10.200.5.100:8888
listenaddr: 10.200.5.100
listenport: 8888
```

Configuring OpenLI

```
# Options for writing intercepts to pcap
# Hopefully you won't need to use this feature...

# Directory to store pcap files in
pcapdirectory: /traces/openlipcaps/

# Maximum duration of pcap file, in minutes
pcaprotatefreq: 30
```

Configuring OpenLI

- And we’re done with the mediator
  - Configure our “real” one...

Configuring OpenLI

- Finally, let's look at the collector config
  - `/etc/openli/collector-example.config`

https://bit.ly/2EtAv6U

Configuring OpenLI

```
# Connect to our provisioner
provisioneraddr: 10.0.0.1
provisionerport: 9001
```

Configuring OpenLI

```
# Unique ID string for my network (16 chars max)
operatorid: WAND

# Unique ID string for the location of my collector
# (16 chars max)
networkelementid: HLZ-BNG

# Unique ID string for this collector (optional)
# (8 chars max)
interceptpointid: col01
```

Configuring OpenLI

```
# Number of threads to use for sequence number tracking.
# You probably don't need to change this.
seqtrackerthreads: 1

# Number of threads to use to encode intercepted packets.
encoderthreads: 2

# Number of threads for forwarding records to mediators
# You probably don't need to change this.
forwardingthreads: 1
```

Configuring OpenLI

```
# List of interfaces to capture packets on
inputs:
 - uri: eth1
   threads: 2
 - uri: eth2
   threads: 4
```

Configuring OpenLI

- Configure our “real” collector and we’re ready to go!

Starting OpenLI services

- Assuming a packaged install, run one of:

```
sudo service openli-provisioner start                OR

sudo systemctl start openli-provisioner.service      OR

/etc/init.d/openli-provisioner start
```

- Repeat for the collector and mediator services

OpenLI services

- At this point, OpenLI is just like any other service

- Commands to interact with an OpenLI service
  - `start`
  - `stop`
  - `restart`
  - `reload`
  - `status`

OpenLI Logging

- Each component logs to `/var/log/openli/`
  - Docker users may want to restart rsyslogd

Starting OpenLI services

- Let’s try and turn it all on...

So Now What?

- How do we know if it is all working?
  - Libtrace is able to act as an ETSI LI client
  - Use `tracepktdump` as a pretend LEA

Testing with tracepktdump

- First, add your “test” agency to provisioner config

```
# List the set of law enforcement agencies
agencies:
 - agencyid: "Test"
   hi2address: 127.0.0.1
   hi2port: 44444
   hi3address: 127.0.0.1
   hi3port: 44445
   keepalivefreq: 300
   keepalivewait: 30
```

Testing with tracepktdump

- Next, create an intercept for your agency

```
# List of active IP intercepts
ipintercepts:

  # Target identified by static IP
  - liid: TEST01
    authcountrycode: NZ
    deliverycountrycode: NZ
    mediator: 6001
    agencyid: "Test"
    accesstype: "fiber"
    staticips:
       - iprange: X.X.X.X/32
         sessionid: 10
```

Testing with tracepktdump

- Tell the provisioner to reload its config
  - Mediator logs should mention the new agency
  - Collector logs should mention the new intercept

```
sudo service openli-provisioner reload
```

Testing with tracepktdump

- On the mediator, start `tracepktdump`
  - Use the IP + port from our agency configuration
  - Two instances, one per handover interface

```
sudo apt-get install libtrace4-tools

tracepktdump etsilive:127.0.0.1:44444

tracepktdump etsilive:127.0.0.1:44445
```

Testing with tracepktdump

- Generate some traffic on your capture interface
  - Make sure it belongs to the intercept target!

Testing with tracepktdump

```
Mon Dec 10 14:56:32 2018
 Capture: Packet Length: 170/170 Direction Value: -1
 ETSILI: pS-PDU:
 ETSILI:   PSHeader:
 ETSILI:     li-psDomainId: 0.4.0.2.2.5.1.17.0
 ETSILI:     lawfulInterceptionIdentifier: TEST01
 ETSILI:     authorizationCountryCode: NZ
 ETSILI:     communicationIdentifier:
 ETSILI:       networkIdentifier:
 ETSILI:         operatorIdentifier: Shane
 ETSILI:         networkElementIdentifier: intercept01
 ETSILI:       communicationIdentifier: 1322238349
 ETSILI:       deliveryCountryCode: NZ
 ETSILI:     sequenceNumber: 133518
 ETSILI:     microSecondTimeStamp:
 ETSILI:       seconds: 1544406992
 ETSILI:       microSeconds: 528115
 ETSILI:     timeStampQualifier: timeOfInterception
```

Testing with tracepktdump

```
Mon Dec 10 14:56:32 2018
 Capture: Packet Length: 170/170 Direction Value: -1
 ETSILI: pS-PDU:
 ETSILI:   PSHeader:
 ETSILI:     li-psDomainId: 0.4.0.2.2.5.1.17.0
 ETSILI:     lawfulInterceptionIdentifier: TEST01
 ETSILI:     authorizationCountryCode: NZ
 ETSILI:     communicationIdentifier:
 ETSILI:       networkIdentifier:
 ETSILI:         operatorIdentifier: Shane
 ETSILI:         networkElementIdentifier: intercept01
 ETSILI:       communicationIdentifier: 1322238349
 ETSILI:       deliveryCountryCode: NZ
 ETSILI:     sequenceNumber: 133518
 ETSILI:     microSecondTimeStamp:
 ETSILI:       seconds: 1544406992
 ETSILI:       microSeconds: 528115
 ETSILI:     timeStampQualifier: timeOfInterception
```

Testing with tracepktdump

```
Mon Dec 10 14:56:32 2018
 Capture: Packet Length: 170/170 Direction Value: -1
 ETSILI: pS-PDU:
 ETSILI:   PSHeader:
 ETSILI:     li-psDomainId: 0.4.0.2.2.5.1.17.0
 ETSILI:     lawfulInterceptionIdentifier: TEST01
 ETSILI:     authorizationCountryCode: NZ
 ETSILI:     communicationIdentifier:
 ETSILI:       networkIdentifier:
 ETSILI:         operatorIdentifier: Shane
 ETSILI:         networkElementIdentifier: intercept01
 ETSILI:       communicationIdentifier: 1322238349
 ETSILI:       deliveryCountryCode: NZ
 ETSILI:     sequenceNumber: 133518
 ETSILI:     microSecondTimeStamp:
 ETSILI:       seconds: 1544406992
 ETSILI:       microSeconds: 528115
 ETSILI:     timeStampQualifier: timeOfInterception
```

Testing with tracepktdump

```
Mon Dec 10 14:56:32 2018
 Capture: Packet Length: 170/170 Direction Value: -1
 ETSILI: pS-PDU:
 ETSILI:   PSHeader:
 ETSILI:     li-psDomainId: 0.4.0.2.2.5.1.17.0
 ETSILI:     lawfulInterceptionIdentifier: TEST01
 ETSILI:     authorizationCountryCode: NZ
 ETSILI:     communicationIdentifier:
 ETSILI:       networkIdentifier:
 ETSILI:         operatorIdentifier: Shane
 ETSILI:         networkElementIdentifier: intercept01
 ETSILI:       communicationIdentifier: 1322238349
 ETSILI:       deliveryCountryCode: NZ
 ETSILI:     sequenceNumber: 133518
 ETSILI:     microSecondTimeStamp:
 ETSILI:       seconds: 1544406992
 ETSILI:       microSeconds: 528115
 ETSILI:     timeStampQualifier: timeOfInterception
```

Testing with tracepktdump

```
Mon Dec 10 14:56:32 2018
 Capture: Packet Length: 170/170 Direction Value: -1
 ETSILI: pS-PDU:
 ETSILI:   PSHeader:
 ETSILI:     li-psDomainId: 0.4.0.2.2.5.1.17.0
 ETSILI:     lawfulInterceptionIdentifier: TEST01
 ETSILI:     authorizationCountryCode: NZ
 ETSILI:     communicationIdentifier:
 ETSILI:       networkIdentifier:
 ETSILI:         operatorIdentifier: Shane
 ETSILI:         networkElementIdentifier: intercept01
 ETSILI:       communicationIdentifier: 1322238349
 ETSILI:       deliveryCountryCode: NZ
 ETSILI:     sequenceNumber: 133518
 ETSILI:     microSecondTimeStamp:
 ETSILI:       seconds: 1544406992
 ETSILI:       microSeconds: 528115
 ETSILI:     timeStampQualifier: timeOfInterception
```

Testing with tracepktdump

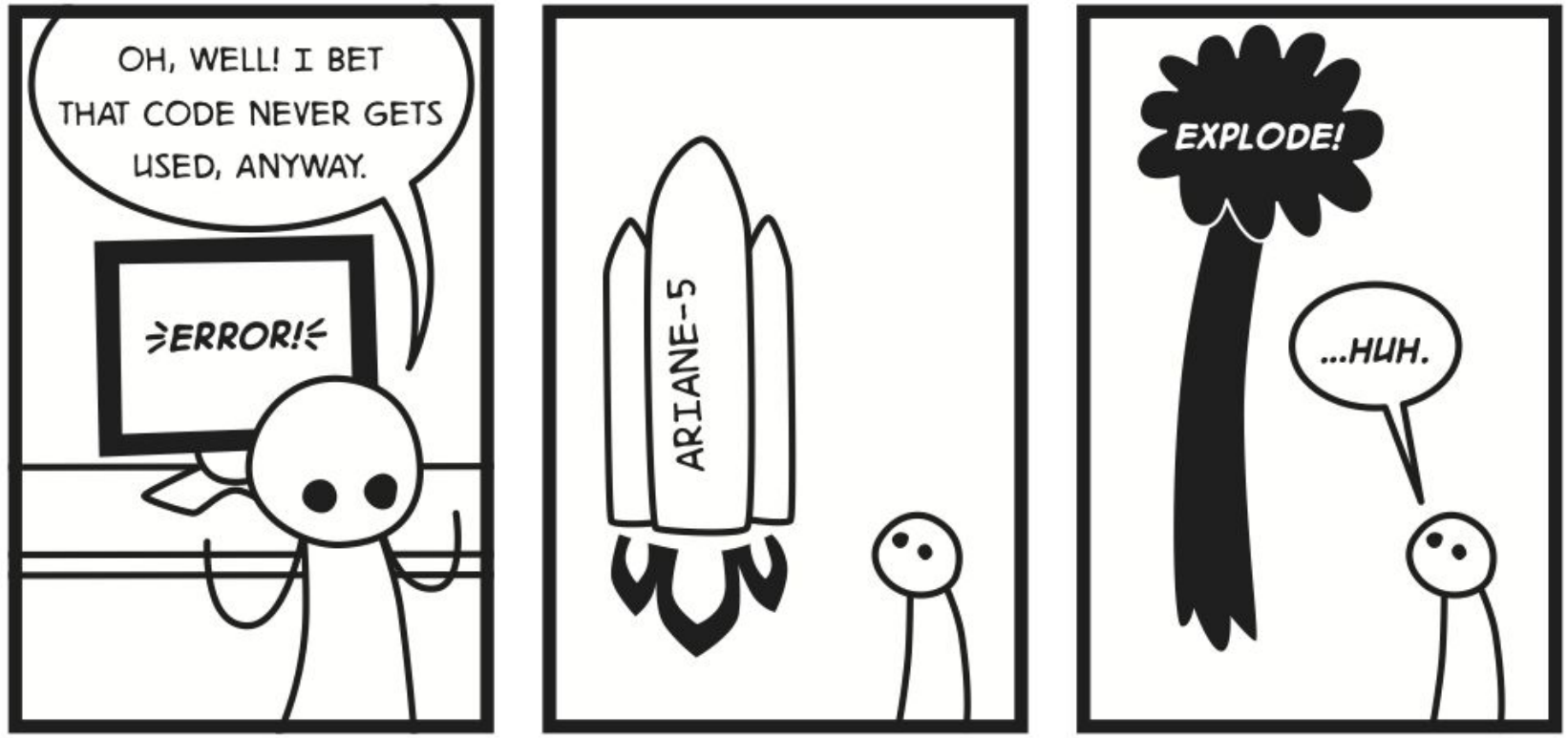
```
ETSILI:   Payload:
ETSILI:     cCPayloadSequence:
ETSILI:       CCPayload:
ETSILI:         payloadDirection: toTarget
ETSILI:         cCContents:
ETSILI:           iPCC:
ETSILI:                 iPCCObjId: .5.3.10.2
ETSILI:                 iPCCContents:
ETSILI:               iPPackets: ...
IP: Header Len 20 Ver 4 DSCP 00 ECN 0 Total Length 52
IP: Id 46742 Fragoff 0 DONT_FRAG
IP: TTL 62 Proto 6 (tcp) Checksum 7463
IP: Source 17.253.67.204 Destination 10.0.0.1
TCP: Source 80 (http) Dest 51380
TCP: Seq 446314071
TCP: Ack 228132013
TCP: DOFF 8 Flags: FIN ACK Window 130
TCP: Checksum 19262 Urgent 0
...
```

Testing with tracepktdump

```
ETSILI:   Payload:
ETSILI:     cCPayloadSequence:
ETSILI:       CCPayload:
ETSILI:         payloadDirection: toTarget
ETSILI:         cCContents:
ETSILI:           iPCC:
ETSILI:                 iPCCObjId: .5.3.10.2
ETSILI:                 iPCCContents:
ETSILI:               iPPackets: ...
IP: Header Len 20 Ver 4 DSCP 00 ECN 0 Total Length 52
IP: Id 46742 Fragoff 0 DONT_FRAG
IP: TTL 62 Proto 6 (tcp) Checksum 7463
IP: Source 17.253.67.204 Destination 10.0.0.1
TCP: Source 80 (http) Dest 51380
TCP: Seq 446314071
TCP: Ack 228132013
TCP: DOFF 8 Flags: FIN ACK Window 130
TCP: Checksum 19262 Urgent 0
...
```

Testing with tracepktdump

- Try testing a “real” OpenLI deployment



Interfacing with Neil

- We think we're doing OK…

- Now try testing with someone else
  - Arrange some testing with Neil

Interfacing with Neil

- LIVE DEMO (maybe)!

Gotchas

- VOIP vendor eccentricities
  - Static “random” fields
  - Timestamps of 0
  - Non-incrementing sequence numbers

- Expect some teething problems, especially while OpenLI is new

Gotchas

- RADIUS
  - Don’t forget to include the port for the accounting packets!
  - Make sure all relevant RADIUS is mirrored to the collector

Gotchas

- IPSec
  - /24 vs /32
  - Firewalls

Building from Source

- Not recommended, but possible

- Install “standard” dependencies
  - libyaml
  - libosip2 (collector only)
  - uthash
  - libzmq
  - libJudy (collector & mediator)
  - libtcmalloc / libgoogle-perftools (collector only)
  - libpcap

- Install development versions of libraries

Building from Source

- Install developer tools
  - gcc
  - autoconf
  - automake
  - libtool
  - flex
  - bison

Building from Source

- Install “WAND-specific” dependencies
  - libwandio
    - https://research.wand.net.nz/software/libwandio.php
  - libwandder
    - https://github.com/wanduow/libwandder/releases
  - libtrace
    - https://research.wand.net.nz/software/libtrace.php

Acquiring OpenLI source code

- Clone the git repository
  - https://github.com/wanduow/openli.git

- Run `./bootstrap.sh` script inside openli directory

Acquiring OpenLI source code

- Download a source tarball
  - https://github.com/wanduow/openli/releases

- Extract the tarball

```
tar -xvzf openli-1.0.0.tar.gz
```

Building OpenLI source code

```
./configure   <options>
make
sudo make install
```

- configure options can include:
  - --disable-provisioner
  - --disable-collector
  - --disable-mediator

Running OpenLI source code

- Components are installed in `/usr/local/bin/`

- No init scripts are installed and no services registered
  - You can steal init scripts from `debian/`

Running OpenLI source code

```
[sudo] /usr/local/bin/openli-<component>
          -c <configfile>
        [ -d ]
        [ -p <pidfile> ]

-c <configfile> : use <configfile> as the
                  configuration file  (required!)
-d              : run component in the background
-p <pidfile>    : write PID into <pidfile>


NOTE: openli-collector will almost certainly require
root privileges, other components should be fine without.
```

DPDK

- Data Plane Development Kit
  - Originally developed by Intel for their NICs
  - Now more generalised

- Libraries + drivers for fast packet processing
  - Packets bypass kernel stack

- Enable 10Gb packet capture using commodity NICs

Why DPDK for OpenLI?

- UFB means customers have plenty of bandwidth
  - 1Gb plans are not uncommon

- Kernel packet capture is not reliable at high speed
  - Packet drops == undercollection

- Don’t forget you may have simultaneous intercepts!

Recommendations for DPDK

- Use DPDK for your customer IP traffic interface
  - RADIUS, SIP, RTP, etc. can all use kernel capture

- One 10Gb DPDK-capable card per collector
  - https://core.dpdk.org/supported/
  - Intel X520 or 82599 are good options

Configuring DPDK

- OpenLI source installs
  - https://github.com/wanduow/openli/blob/develop/doc/DPDKNotes.md
  - You'll need to rebuild libtrace
    - Add `--with-dpdk=yes` to `./configure`

Installing DPDK

- Packaged OpenLI installs
  - In theory, this should be easier
  - In practice, not so much :(

Installing DPDK

- Install the package for the igb_uio kernel module

```
sudo apt-get install dpdk dpdk-igb-uio-dkms
```

Running DPDK

- Run script to setup required environment variables

```
source /usr/share/dpdk/dpdk-sdk-env.sh
```

- Consider copying the contents of the script into `~/.profile`

Running DPDK

- Manually load the igb_uio module (!)

```
sudo modprobe uio

sudo insmod /var/lib/dkms/dpdk-igb-uio/ \
[dpdk version]/[kernel version]/x86_64/modules/igb_uio.ko
```

Running DPDK

- Run the DPDK configuration script

```
sudo -E /usr/share/dpdk/usertools/dpdk-setup.sh
```

OR

```
sudo -E /usr/share/dpdk/tools/dpdk-setup.sh
```

Running DPDK

- Setup hugepage mappings
  - Choose “Setup hugepage mappings for NUMA systems”
  - Enter ‘2048’ for each NUMA node that you are asked about

Running DPDK

- Bind an interface to DPDK
  - Choose “Bind Ethernet/Crypto device to IGB UIO module”
  - Enter PCI address of interface to bind
    - Use “`if=`” field to identify interface by name
    - PCI address will look like `0000:XX:00.X`

- `IMPORTANT:` once bound, the interface is no longer usable by the kernel
  - No more regular traffic
  - Don’t bind the interface you are using to reach the machine!

Using DPDK with OpenLI

- In your collector config...

```
# List of interfaces to capture packets on
inputs:
 - uri: eth1
   threads: 2

 - uri: dpdk:0000:42:00.1
   threads: 4
```

Future Development

- Provisioner configuration
  - Socket for pushing live intercept config
  - Expressed as a JSON object
  - Can integrate directly into existing web platforms

Future Development

- Internal communication security
  - Inter-component comms are unencrypted
  - You should keep these boxes isolated!
  - SSL, keys + certs, cert management
    - Leverage existing AMP experience

Future Development

- Disk backed buffering
  - Memory-backed for now, but limited capacity
  - Fall back to disk before memory gets full
  - Clear backlog when situation is resolved

Future Development

- Deal with bug reports and user feedback
  - Listen to the users :)

- Community building
  - Encourage others to become OpenLI experts
  - Invite new developers to work on the code base
  - Shared documentation covering common issues, use cases

Winding Up

- Questions
- Answers?
- Discussions
- Feature Requests / Ideas
- Problems
- Beers

Winding Up

Website: https://openli.nz

Code: https://github.com/wanduow/openli

Packages: https://www.bintray.com/wand/

Email me: shane.alcock@waikato.ac.nz

Workshop Resources:
https://wand.net.nz/~salcock/nznog19/openli-tutorial