Writing Bro Analyzers

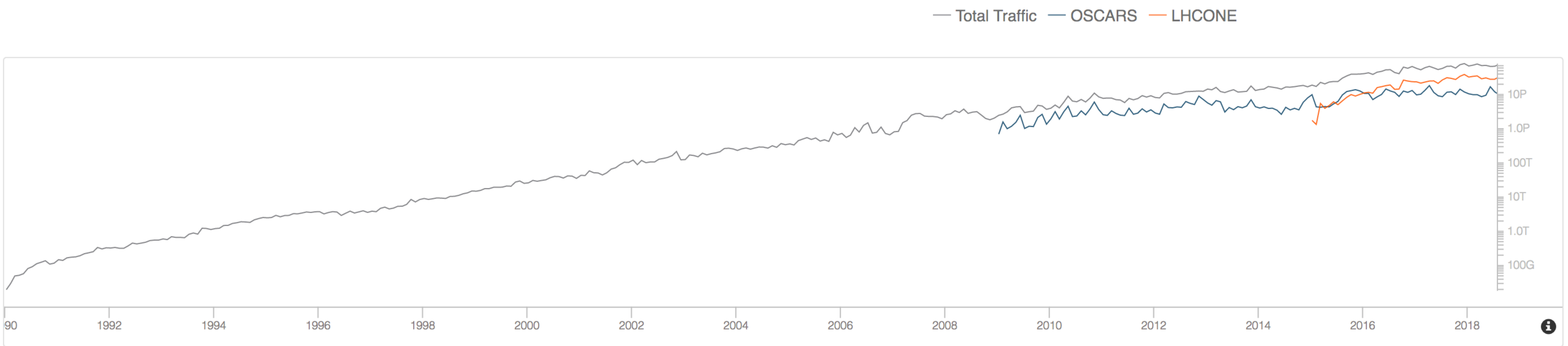
Vlad Grigorescu
Bro Workshop Germany 2018

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

HOME »

Traffic Volume



August 2018

	Bytes	Percent of Total	One Month Change	One Year Change
OSCARS	9.51PB	12.8%	-18.4%	-15.6%
LHCONE	31.83PB	42.9%	+17.7%	+6.07%
Normal traffic	32.80PB	44.2%	+26.8%	+41.0%
Total	74.13PB		+14.9%	+14.9%

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Goal

- You're not going to become experts in 30 minutes.
- Present an overview of the steps needed to write a new binpac protocol or file analyzer in Bro,
- Demonstrate a couple of tools that will help you along the way,
- Provide you with some resources for when you get stuck.

Bro Data Flow

Core Layer

1. Parse data "off the wire"
2. Generate Bro events

Script Layer

3. Handle the events
4. Generate logs

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

1: Parsing Data Off the Wire

U.S. DEPARTMENT OF ENERGY

Office of Science

ESnet

binpac: Binary Protocol Analyzer Compiler

- Parsing binary protocols is hard (and dangerous!)
- binpac provides a higher-level abstraction which compiles into C++
- Common primitives make it much easier to parse binary protocols
- binpac restricts what it allows you to do
  - ...but can always be extended with C++ code.

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Syslog Refresher

```
<  Priority  >  Message
<     30     >  Mar 30 13:10:01 ...
```

```
<30>Mar 30 13:10:01 bro-net-lon systemd: Created slice User
Slice of root.

<78>Mar 30 13:10:01 bro-net-lon CROND[32309]: (root) CMD (/usr/
lib64/sa/sa1 1 1)

<189>Mar 30 13:15:01 net-lon-sw1 NET-LON-SW1: xntpd: kernel time
sync enabled 6001
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

binpac Sample (syslog-protocol.pac)

```
type Syslog_Message = record {
        PRI: Syslog_Priority;
        msg: bytestring &restofdata;
} &byteorder = littleendian;
```

U.S. DEPARTMENT OF
ENERGY
Office of Science

ESnet

binpac Sample (syslog-protocol.pac)

```
type Syslog_Message = record {
        PRI: Syslog_Priority;
        msg: bytestring &restofdata;
} &byteorder = littleendian;

type Syslog_Priority = record {
        lt: uint8; # '<'
        val: RE/[[:digit:]]+/;
        gt: uint8; # '>'
} &let {
        val_length: int = sizeof(val) - 1;
        int_val: int = bytestring_to_int(val, 10);
        severity: int = (int_val & 0x07);
        facility: int = (int_val & 0x03f8) >> 3;
};
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Parsing Data

- binpac makes it much easier to parse the data
- Use records as building blocks
- Can execute custom (C++) code
- Can execute code after a record is parsed
- Data is often parsed by *-protocol.pac file

2: Generating Events

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

binpac Sample (radius-analyzer.pac)

```
refine typeattr RADIUS_Attribute += &let {
  proc: bool = $context.flow.proc_radius_attribute(this);
};
```

binpac Sample (radius-analyzer.pac)

```
function proc_radius_attribute(attr: RADIUS_Attribute): bool
    %{
    BifEvent::generate_radius_attribute(
        connection()->bro_analyzer(),
        connection()->bro_analyzer()->Conn(),
        ${attr.code},
        bytestring_to_val(${attr.value}));

    return true;
    %}
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

binpac Sample (radius-analyzer.pac)

```
event radius_attribute(c: connection, attr_type: count, value: string)
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

binpac Sample (radius-analyzer.pac)

```
event radius_attribute(c: connection, attr_type: count, value: string)
```

```
function proc_radius_attribute(attr: RADIUS_Attribute): bool
    %{
    BifEvent::generate_radius_attribute(
        connection()->bro_analyzer(),
        connection()->bro_analyzer()->Conn(),
        ${attr.code},
        bytestring_to_val(${attr.value}));

    return true;
    %}
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Generating Events

- Need to define your events (events.bif)
- Need to convert binpac data types and structures to native Bro types
- This part is often handled by *-analyzer.pac file
- Complex types (records, vectors, etc.) are harder to convert

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

3: Handling Events

U.S. DEPARTMENT OF ENERGY

Office of Science

ESnet

base/protocols/syslog.bro

```
event syslog_message(c: connection, facility: count, severity:
count, msg: string) &priority=5 {
   local info: Info;
   info$ts         = network_time();
   info$uid        = c$uid;
   info$id         = c$id;
   info$proto      = get_port_transport_proto(c$id$resp_p);
   info$facility = facility_codes[facility];
   info$severity = severity_codes[severity];
   info$message  = msg;

   c$syslog = info;
   }
```

U.S. DEPARTMENT OF ENERGY Office of Science

ESnet

4: Generate Logs

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

base/protocols/syslog.bro

```
event syslog_message(c: connection, facility: count,
severity: count, msg: string) &priority=-5
    {
    Log::write(Syslog::LOG, c$syslog);
     }
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Getting Started: The Good

- You can use the existing analyzers as examples!
  - Parsing
  - Converting datatypes and generating events
  - Handling events and creating logs
- Some code can be abstracted and re-used by multiple analyzers
  - For example, ASN1 binpac code in src/analyzer/protocol/asn1

Getting Started: The Bad

- Code is needed to interface binpac analyzers to the Bro source code
    - Differences for TCP and UDP
- Existing analyzers use several different files across a couple of directories
- A lot of moving parts (dynamic protocol detection, plugins?)

binpac Quickstart

Templates to create the boilerplate code for you!

https://github.com/esnet/binpac_quickstart

binpac Quickstart

```
Usage: start.py NAME DESCRIPTION PATH_TO_BRO_SRC (--tcp|--udp) [--buffered] [--plugin]


Arguments:

NAME - Short name of protocol to be used in filenames (e.g. HTTP)
DESCRIPTION - Long name of protocol (e.g. Hypertext Transfer Protocol)
PATH_TO_BRO_SRC - Full path to the Bro source directory, where the files will be written. e.g. ~/src/bro/

Options:

--tcp - Include the TCP analyzer class. You probably want this if this protocol uses TCP.
--udp - Include the UDP analyzer class. You probably want this if this protocol uses UDP.
--buffered - Enable the flow buffer, enabling use of &oneline and &length
             in record types. Without this option, it will be a datagram analyzer,
             which is faster but has no incremental input or buffering support.
--plugin - Create the BinPac files as a plugin. The path to the plugin is substituted for
           the Bro source directory (PATH_TO_BRO_SRC).
```

binpac Quickstart

```
$ ./start.py WHOIS "WHOIS Protocol" ~/projects/bro/bro --tcp --buffered

$ git status
Changes to be committed:
   modified:   scripts/base/init-default.bro
   new file:   scripts/base/protocols/whois/__load__.bro
   new file:   scripts/base/protocols/whois/dpd.sig
   new file:   scripts/base/protocols/whois/main.bro
   modified:   src/analyzer/protocol/CMakeLists.txt
   new file:   src/analyzer/protocol/whois/CMakeLists.txt
   new file:   src/analyzer/protocol/whois/Plugin.cc
   new file:   src/analyzer/protocol/whois/WHOIS.cc
   new file:   src/analyzer/protocol/whois/WHOIS.h
   new file:   src/analyzer/protocol/whois/events.bif
   new file:   src/analyzer/protocol/whois/whois-analyzer.pac
   new file:   src/analyzer/protocol/whois/whois-protocol.pac
   new file:   src/analyzer/protocol/whois/whois.pac
```

binpac Quickstart

```
$ grep -r TODO src/analyzer/protocol/whois scripts/base/
protocols/whois

scripts.../main.bro: TODO: Add other fields here that you'd like to log.
scripts.../main.bro: TODO: The recommended method to do DPD
scripts.../main.bro: TODO: If you're using port-based DPD, uncomment this.
scripts.../dpd.sig:  TODO: Define the payload. When Bro sees this regex
src.../events.bif:   TODO: Edit the sample event, and add more events.
src.../whois.pac:    TODO: Determine if you want flowunit or datagram
src.../whois-protocol.pac:  TODO: Add your protocol structures in here.
```

binpac Quickstart

```
$ ./configure
$ make
$ source build/bro-path-dev.sh
$ bro -N | grep WHOIS
Bro::WHOIS - WHOIS Protocol analyzer
(built-in)
```

File Analyzers

- Some file analyzers are written entirely (or partially) in binpac.
  - PE (Windows Portable Executable), Unified2 (Snort), x509
- binpac is very well suited for this type of binary protocol as well.
  - ...with some limitations
- Hope to extend binpac_quickstart to support file analyzers too.

pynpac

File analyzer prototyping in Python!

https://github.com/grigorescu/pynpac

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

pynpac (binpac version)

```
type DOS_Header = record {
    signature                 : bytestring &length=2;
    UsedBytesInTheLastPage : uint16;
    FileSizeInPages           : uint16;
    ...
    OverlayNumber             : uint16;
    Reserved                  : uint16[4];
    OEMid                     : uint16;
    OEMinfo                   : uint16;
    Reserved2                 : uint16[10];
    AddressOfNewExeHeader   : uint32;
} &length=64;
```

U.S. DEPARTMENT OF ENERGY Office of Science

ESnet

pynpac (Python version)

```
f = open("samples/payload.exe")
p = Parser(f)
dos_header = Record()

logging.debug("Parsing dos_header")
dos_header["signature"]             = p.parse("2s")
dos_header["UsedBytesInTheLastPage"] = p.parse(uint16)
dos_header["FileSizeInPages"]       = p.parse(uint16)
dos_header["OverlayNumber"]         = p.parse(uint16)
dos_header["Reserved"]              = p.parse(uint16*4)
dos_header["OEMid"]                 = p.parse(uint16)
dos_header["OEMinfo"]               = p.parse(uint16)
dos_header["Reserved2"]             = p.parse(uint16*10)
dos_header["AddressOfNewExeHeader"] = p.parse(uint32)

p.data["dos_header"] = dos_header
```

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet

Resources

- *binpac: A yacc for Writing Application Protocol Parsers*
- https://www.bro.org/development/howtos/binpac-sample-analyzer.html
- https://www.bro.org/sphinx/components/binpac/README.html
- GitHub:
    - esnet/binpac_quickstart
    - grigorescu/pynpac

U.S. DEPARTMENT OF ENERGY Office of Science

ESnet

Tips

- E-mail bro-dev before starting
- Read the RFC
    - ...even if Microsoft doesn't
- Compare your analyzer with Wireshark and Microsoft Message Analyzer
    - ...but neither should be considered "more correct"
- Try refactoring BinPAC records
- Often hard to find PCAPs (generate your own?)

U.S. DEPARTMENT OF ENERGY
Office of Science

ESnet