YAF

A Case Study in Flow Meter Design

presented at
FloCon 2008 - Savannah, Georgia

Brian Trammell
Technical Lead, Engineering
CERT Network Situational Awareness

YAF

Open-source, IPFIX-compliant bidirectional flow meter

- Available from http://tools.netsa.cert.org

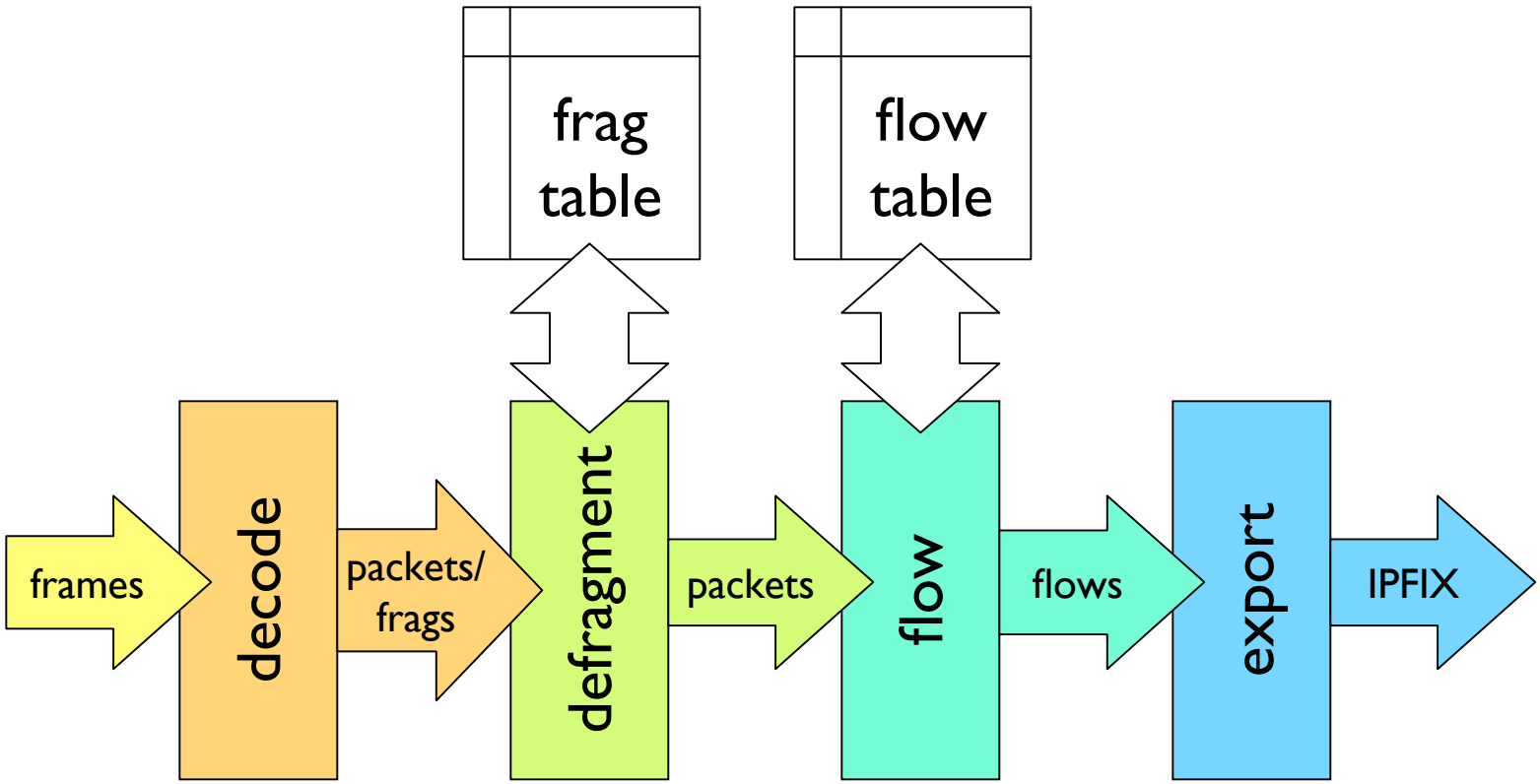
Processes packets from multiple inputs

- libpcap dumpfiles (ad-hoc packet analysis)
- libpcap live capture (including proprietary pcap interfaces, e.g. Bivio)
- Endace DAG live capture

Performance is network hardware and I/O bound…

- …easily handles OC3, OC12, GigE at line speed, but
- 10GigE requires proprietary hardware at saturation.

Flow Meter Design



Flow Meter Effects on Flow Data

Fragmentation

End Conditions

Timeouts

Delta Counters

Biflows

The Packet Clock

Fragmentation

Three approaches for flowing fragmented traffic:

- pretend there's no such thing as fragmentation,
- drop all fragmented packets, or
- full or partial fragment reassembly

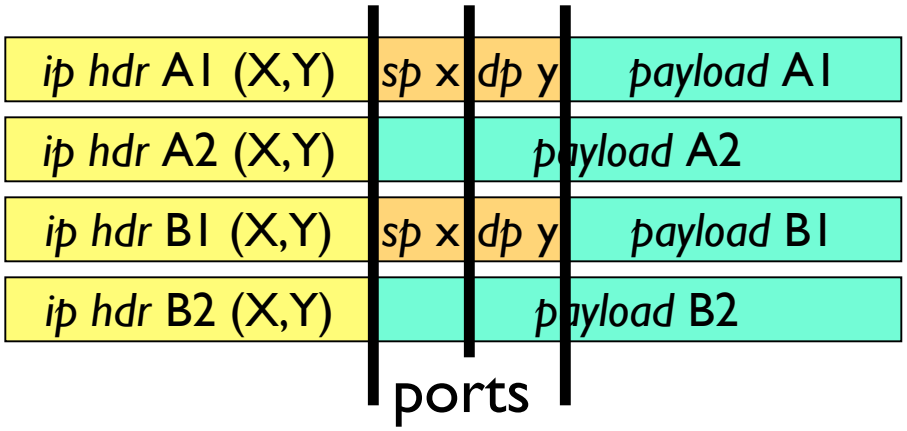
Each approach has tradeoffs, and is applicable in certain situations.

YAF supports partial reassembly.

Fragmentation?

Easiest way to handle fragmentation: don't.

Leads to inaccurate flow data as subsequent fragment port numbers are incorrectly decoded:



sip	*dip*	*proto*	*sp*	*dp*	*pkts*
X.X.X.X	Y.Y.Y.Y	6	x	y	2
X.X.X.X	Y.Y.Y.Y	6	$A2_0$	$A2_2$	1
X.X.X.X	Y.Y.Y.Y	6	$B2_0$	$B2_2$	1

Fragmentation? (2)

Often used in resource-restricted environments (e.g., routers).

- Much faster: no requirement even to recognize fragmented packets.
- Much less memory consumption: no fragment table.
- Less susceptible to resource exhaustion attacks.

Trivially easy to implement.

Difficult or impossible to recover actual flows from random fragment offset port data.

Dropping fragmented packets

Requires minimal resources at flow meter:

- need to recognize fragments, but not store them.

Leads to meter blindness:

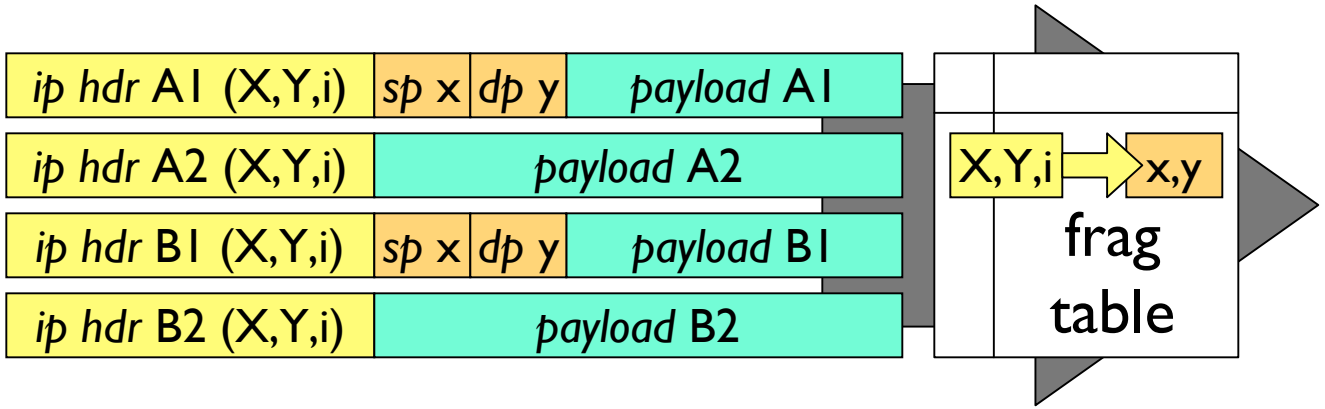
- all an attacker must do to hide from the measurement infrastructure is fragment all packets.

Only applicable behind perimeter devices which also drop all fragmented packets.

```
sip             dip             proto sp    dp    pkts
                      [no flows]
```

Partial fragment reassembly

Associate each fragmented packet with its actual transport ports:



sip	*dip*	*proto*	*sp*	*dp*	*pkts*
X.X.X.X	Y.Y.Y.Y	6	x	y	4

Partial fragment reassembly (2)

Accurately assigns fragments to respective flows.

Requires additional resources at flow meter:

- need to recognize, look up, and store every fragment.

More difficult to implement and maintain.

Requires care to avoid vulnerability to resource exhaustion attacks.

Flow End Conditions

Flow meter must recognize actual connection shutdown…

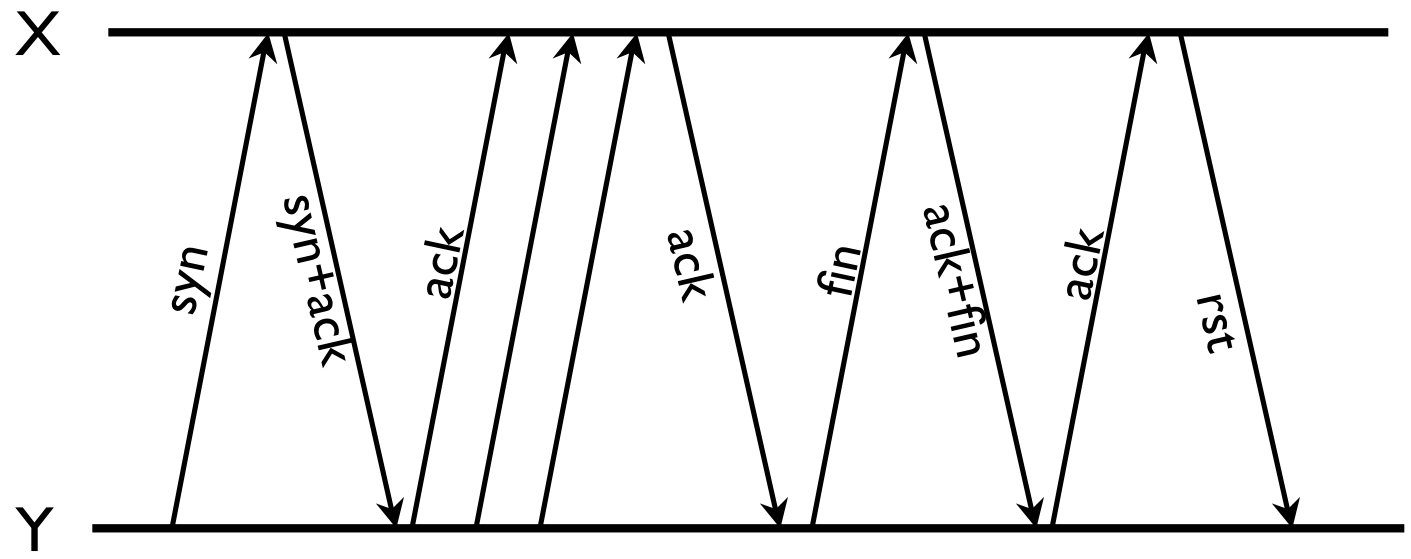
- …through varying degrees of modeling the host TCP state machine.

Flows on the wire are not always so well-behaved.

Example: multiple-RST teardown.

Multiple RST teardown

How many flows here?



sip	*dip*	*flags*	*sp*	*dp*	*pkts*
Y.Y.Y.Y	X.X.X.X	SAF	x	y	6
Y.Y.Y.Y	X.X.X.X	SAF	y	x	3
Y.Y.Y.Y	X.X.X.X	R	y	x	1

Multiple RST teardown (2)

Tempting to group RSTs on teardown into original flow…

- …how long to keep closed flow state?
- …how far to take this RST grouping?
- …how to communicate new configuration parameters to analysts?

YAF stays predictable, at the expense of generating multiple flow records for this behavior.

Passive Timeouts

Flows which have no packets over $TO_{passive}$ seconds are closed.

Necessary to terminate flows for all non-connection-oriented transports,

- i.e., anything but TCP.

Longer passive timeouts consolidate low-frequency periodic activity into fewer flows.

Shorter passive timeouts reduce flow table resource consumption for such activity.

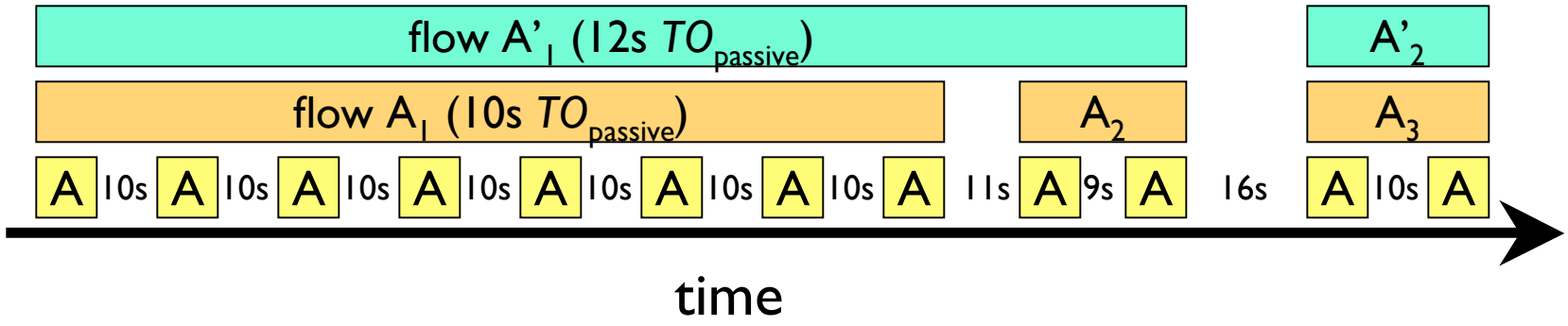
Passive timeouts (2)

Generally chosen to match common protocol timeouts…

- … which are generally round numbers, e.g., 10, 30, 60 sec.

May be chosen to avoid flow closure ambiguity due to minor variations:

- e.g., 12, 33, 64 sec.



Active Timeouts

Flows which have been open for TO_{active} seconds are closed.

- Maximum flow duration is TO_{active} seconds.

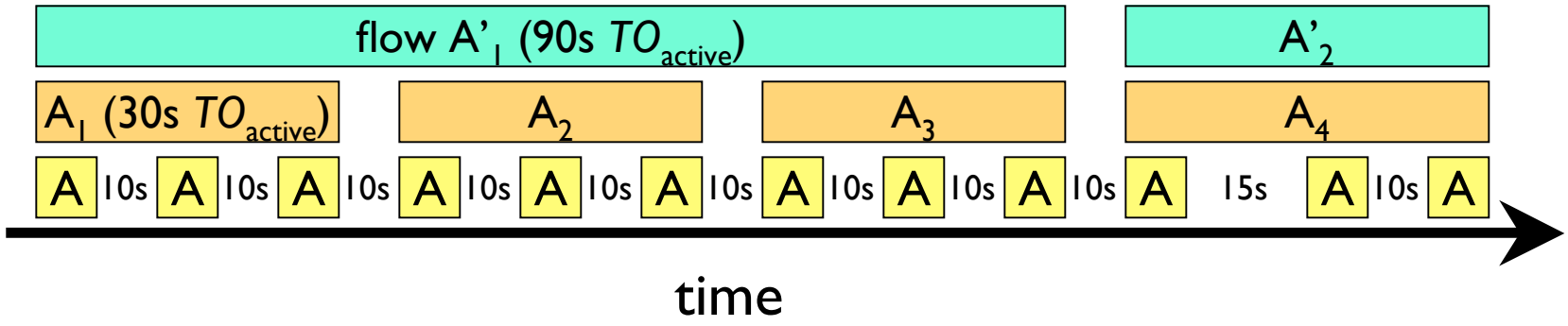
Necessary to ensure long-lived flows are eventually flushed from the flow table.

Active timeout determines reporting delay.

Active Timeouts (2)

Shorter active timeouts used for more rapid reporting.

Longer active timeouts used for better data reduction.



Delta Counters

Flow meters which periodically emit multiple flow records per flow (for rapid reporting) may use total or delta counters.

Total counters replace values in previous flow records.

Delta counters add to values in previous flow records…

- …thereby reducing state requirements on meter and increasing them on collector.

YAF uses total counters, but doesn't emit multiple records per flow…

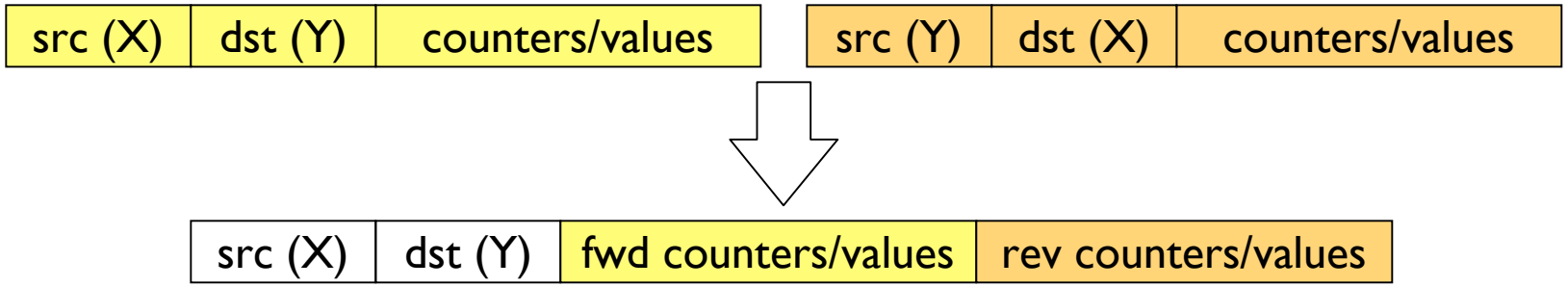
- …uses active timeout instead.

Biflows

Representation of two sides of a connection with a single flow record:

- Allows additional data reduction
- Enables easier connection analysis
- Improves flow state modeling at flow meter

YAF is a biflow meter, but SiLK stores uniflows.



The Packet Clock

Important to drive all processes within a flow meter with a single clock

- fragment timeouts, flow timeouts, time stamping, etc.

When building a flow meter, gettimeofday(2) is not your friend.

- often a problem with porting host-based software into a network-based monitoring environment

Use the timestamp from the packet instead!

- ensures that the resulting flow stream identical whether captured live or generated from dumpfile.

Getting YAF

http://tools.netsa.cert.org

Builds on Mac OS X, Linux, BSD, Solaris

- Bug reports from these or other Unices welcome!

Some prerequisites

- glib-2.0 (C modernization layer)
- libairframe (application utility library from NetSA)
- libfixbuf (IPFIX protocol implementation from NetSA)
- libpcap (generally available on most modern Unices)
- libdag (only required for Endace DAG capture)

Questions?

Ask now…

…or later:

- Brian Trammell <bht@cert.org>
- Chris Inacio <inacio@cert.org>