WireFugue

Scalable Intrusion Detection with Akka, Kafka, and Play

Ian Robertson <iroberts@uw.edu>
University of Washington Professional & Continuing Education Program
Compute-intensive Applications with Scala
Instructors: John Nestor & Jerry Kuch
Spring 2017

Inspiration

- The Bro Network Security Monitor - http://bro.org
- Stoffer, Sharma, Krous, 2015. 100G Intrusion Detection.

100G Intrusion Detection

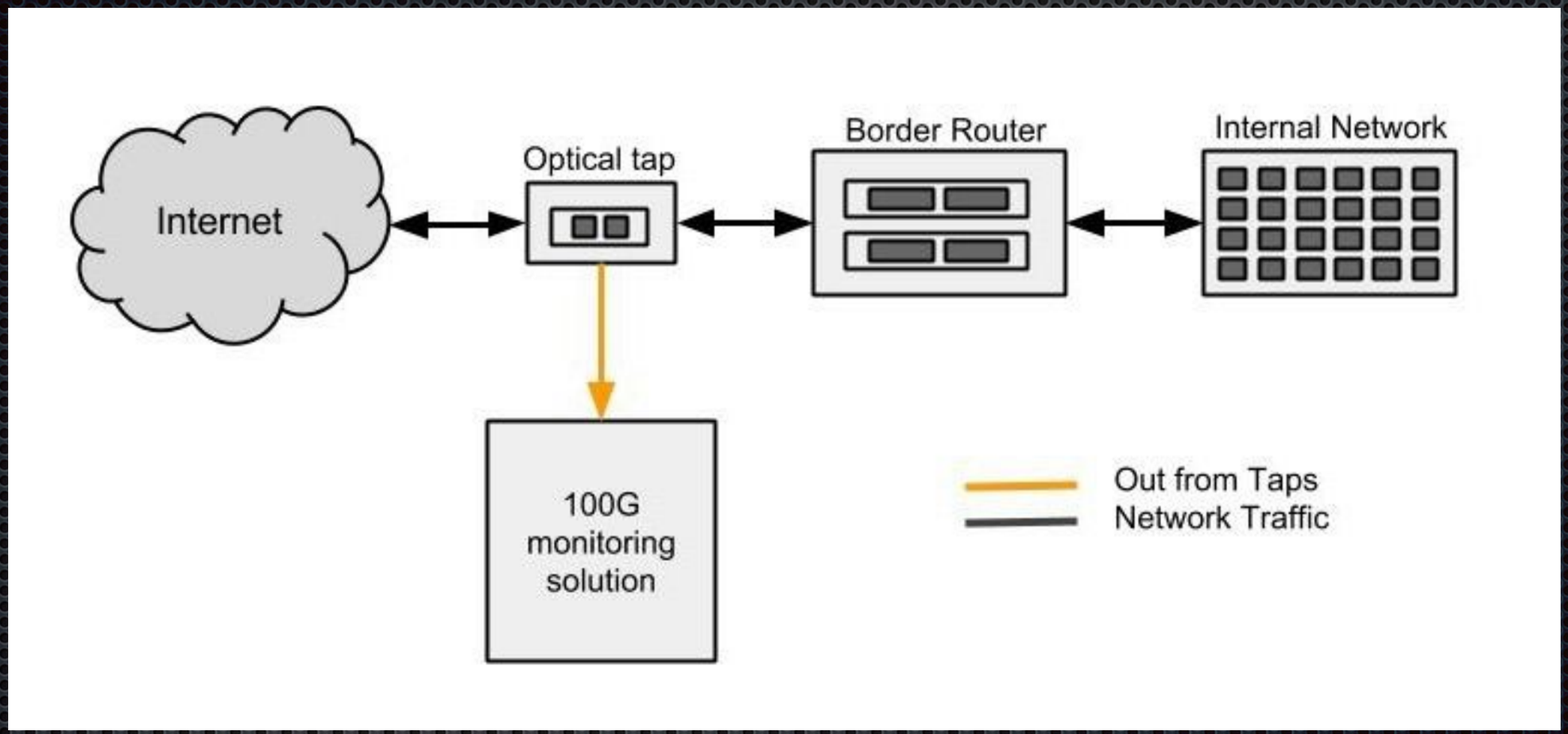
August 2015

v1.0

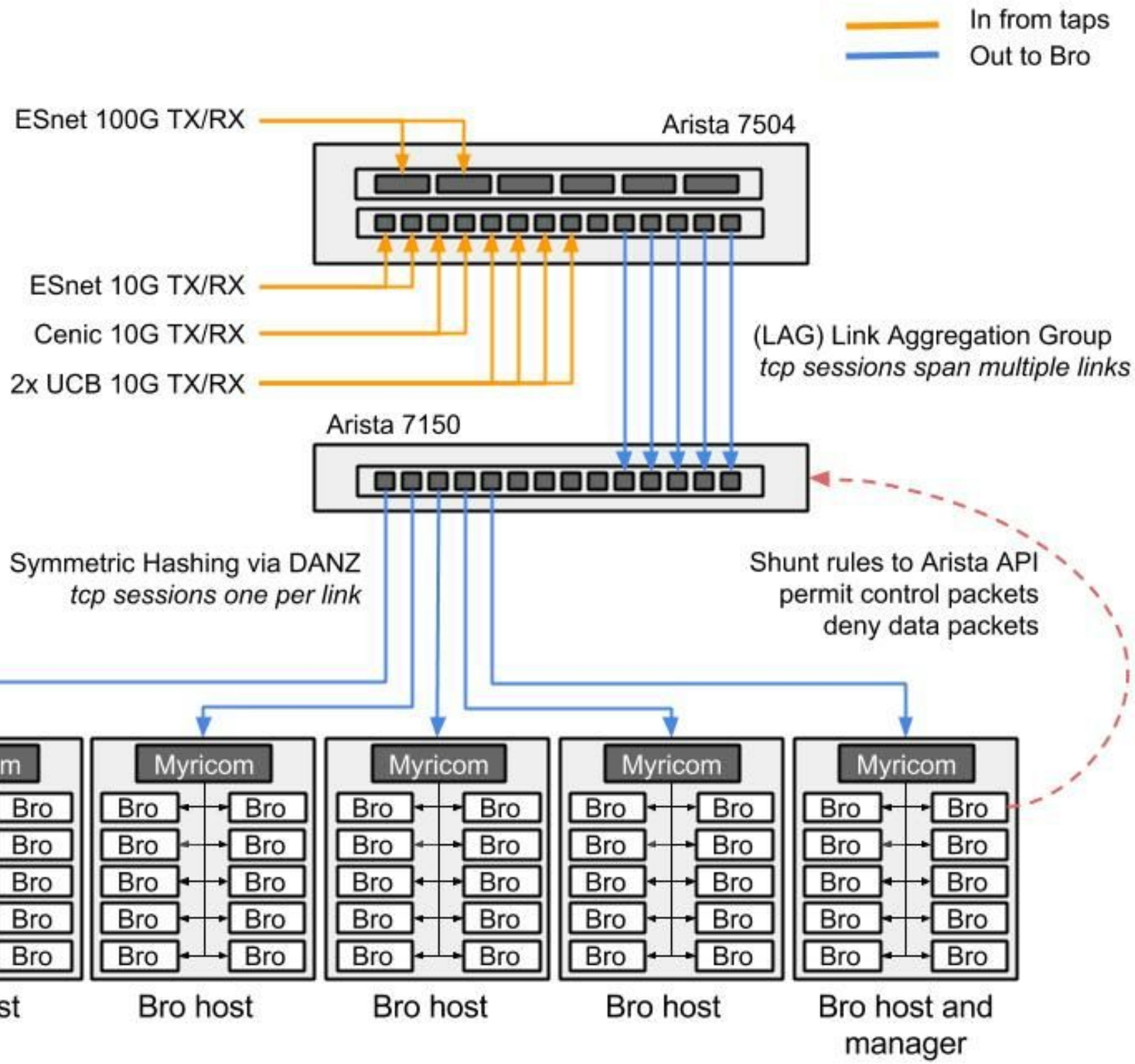
Vincent Stoffer

Aashish Sharma

Jay Krous



Source: Stoffer, Sharma, Krous, 2015. “100G Intrusion Detection”



Source: Stoffer, Sharma, Krous, 2015. “100G Intrusion Detection”

Sensor Group

Sensor

Sensor

Sensor Group

Sensor

Sensor

Sensor

...

To firewall

Firewall Control
(Kafka Consumer)

Kafka Cluster

Analysis Cluster
(Akka clustering + Kafka Streams)

Firewall control

Filter control

Control
backend

REST API

Web App
(Play)

Sensors

- Sensor node sniffs a portion of traffic after hardware load-balancing.
- OS-based capture -> Akka streams filter -> Kafka producer
- Basic filtering available - no heavy parsing
- Filter control via Kafka consumer
- Sensor groups separate logically different network flows

Analysis Node

- Consumes packets from a subset of partitions
- Writes results (events, aggregates, reduced streams, firewall control messages) back to Kafka
- Some nodes may process secondary streams
- Needs to see whole bidirectional TCP sessions

Load balancing

- Analysis nodes need to see complete TCP streams from a single partition
- Hash of Kafka key determines partition:
  (group, proto, Set((sip, sport), (dip, dport))
- Kafka value: [whole IP datagram with IP headers]
- Timestamp: as recorded by sensor

Control Backend

- Consumes event streams generated by analysis cluster
- Makes decisions to set and remove firewall blocks
- Provides event data to web client, other clients
- Forwards sensor filter control messages

Limitations
(project scope reduction)

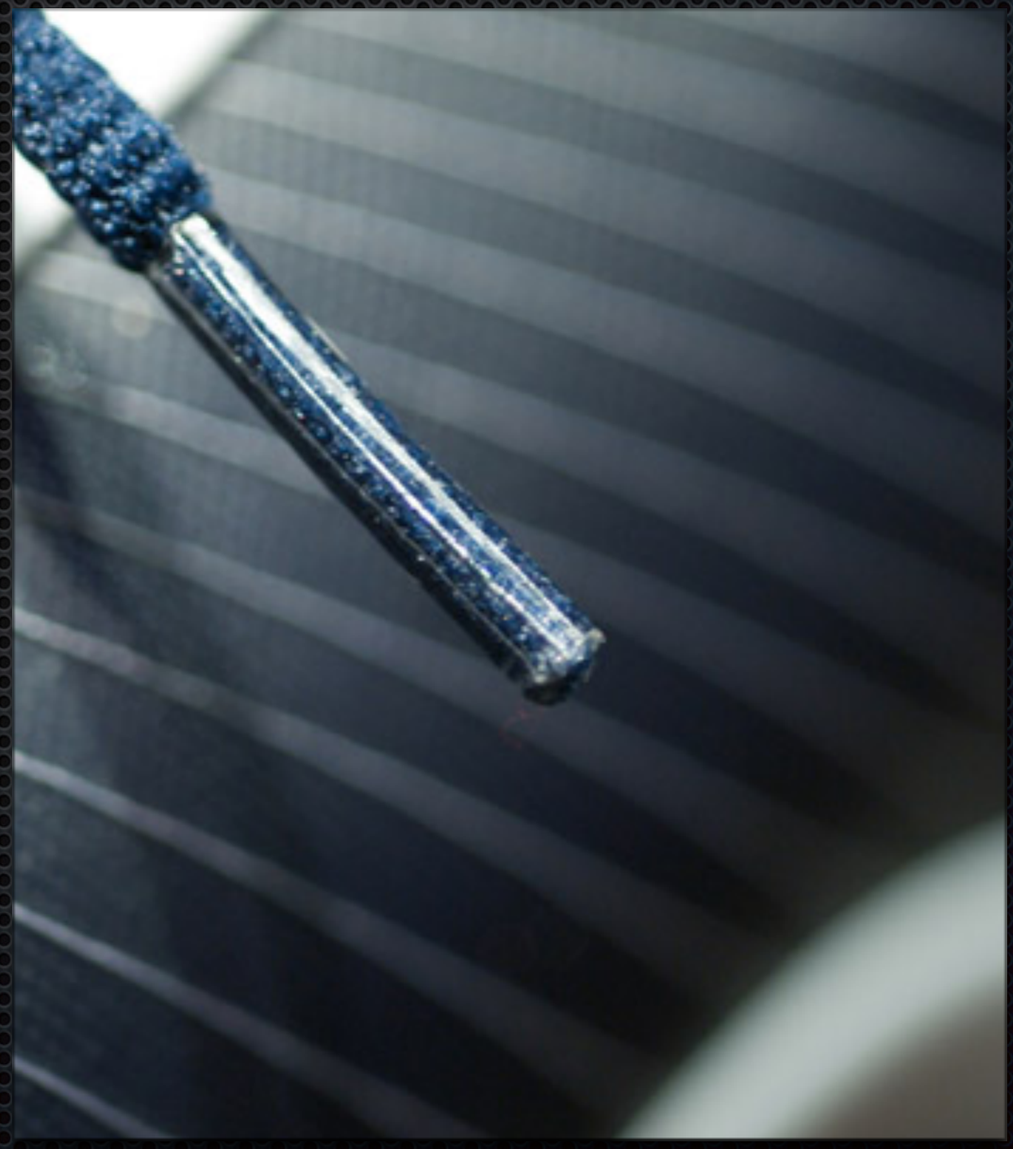
- Only support ICMP, UDP, TCP over Ethernet
- Drop link-level header at sensor. Deal with IP Packets (Datagrams) only. Only IPv4 for now.
- Only implement a few basic analysis agents, but make it extensible.
- Frontend may be rudimentary, but extensible

Test Strategy

1. Basic data flow integration: traffic generator, one sensor, Kafka, one analysis agent, web app. Display real-time packets/sec, bytes/sec.
2. Automate integration test in Jenkins + Docker (on-demand EB Docker hosts?)
3. Load testing with Amazon VPC

Shoestring test strategy

- Resource budget: $100.00
- Amazon AWS: EC2, EBS, Elastic Beanstalk, VPC
- Github
- Jenkins CI
- Docker



Thus far....

- Read and parse pcap files with Akka Streams
- Install & run Kafka & ZK, command-line tutorial
- Kafka conceptual understanding
- Install Jenkins on EC2, runs `sbt version` in Docker
- GitHub: robertson-tech/wirefugue

Hurdles

- Realistic test traffic generation (+ nefarious patterns)
- TCP reassembly
- Learning Play
- Much to learn about Jenkins, Docker, AWS