Streaming Analysis:
An Alternate Analysis Paradigm

FloCon 2014
John M\textsuperscript{c}Hugh
Overview

- The Landscape
- A Streaming Workflow Prototype
- Results
- The Fathom Framework
- Discussion & Future Work
The Landscape

• Right now, we can only find simple and obvious attacks
• In order to stop the smarter attackers, we need to first build a better detection infrastructure, this needs:
  – **Situational Awareness**: We don’t understand what’s on our networks or what they do
  – **Reconnaissance Detection**: We treat each attack as a completely new event
  – **Automation and Efficiency**: Everything is still done by hand and by heroes
• We are building the next generation detection infrastructure, and by doing so will catch progressively stealthier attacks
Streaming Analytics

• The next generation demands streaming to relieve the volume of stored data and decrease threat reaction time
• We initially implemented using IBM’s InfoSphere Streams
  – More recent work uses our own Fathom framework
• Challenge of streaming
  – Only stateless analytics directly convert
  – Complex analytics require rethinking
  – Understanding the streams improves success
• Benefits of streaming: on-the-fly analyses
  – Near real-time products & actions
  – Selective capture to reduce retained volumes
  – Limited but productive state (context) can be maintained
  – Compile these on-the-fly analyses into long term knowledge
Stream Computations for Analytic Network Security

• We implement real-time streaming analysis using workflows

• Describe several computations in this presentation
  – Scan detection via Threshold Random Walk
  – Situational awareness via Continuous Statistics
  – A reimplementation of AMP
    • With extensions to capture flow
Advancing the State-of-the-art

- Scan detection using Threshold Random Walk
  - Faster oracle based approached
  - Efficiently implemented
  - Extendable to continuous operation via oracle and table maintenance

- Situational awareness using Continuous Statistics
  - Finer granularity than previous efforts
  - Detailed network knowledge
  - Working implementation proves this task is less daunting than previously thought
Benefits of the streaming approach

• Scalable
  – Pipelines: many work steps in a row
  – Divide and conquer: parallel streams
  – Physical distribution: reduced volume at source

• Efficient
  – No bottlenecks

• Replicable
  – Easy to add new analytics
Analytic Capabilities (InfoSphere streams prototypes)

1. Threshold Random Walk (TRW)
   – Detects network scanners
     • Processes 1 hour of data in less than 1 minute
     • Detects all the scans detected by CERT’s *rwscan* and more
     • Graphic display of detections and internal state

2. Continuous Statistics
   – Partial statistics for 260K+ entities in network stream
     • Data into dark /8 at ~1.5Mpkts/minute
     • 1 minute epoch aggregates compared with 60 epoch horizon
     • Alerts for outliers
     • Graphic display of traffic rates and alerts
Source Data

**TRW**
- Synthetic data created for IARPA by DHS PREDICT Project
- Traffic on 100.0.0.0/11 network (OSIS)
- Multiple attacks injected into data, including scans
- 1 to 2 hr. scenarios
- ~ 2GB/Hr Data

**Continuous Statistics**
- Live network traces collected from CAIDA network telescope
- Dark space consisting of a single /8
- 72 hour sample of incoming traffic used to generate statistics
- ~ 6GB/Hr Data
1. Threshold Random Walk

- Connections to nonexistent targets are considered suspicious
- TRW *sequentially* tests suspicious connections and raises an alarm
- TRW only cares about the current state, and the next test
TRW and oracles

• An oracle tracks internal network services
  – Updated dynamically by outgoing traffic
  – Used to evaluate connection attempts
• The TRW table tracks hosts connecting to the network
  – Behavior judged by connection success / failure
    • predicted by oracle
  – Host score is a function of success and failure counts
  – When score crosses a threshold, classify the host as a Scanner or as Benign
• The Oracles and TRW tables are SPL maps
  – This may have scaling problems
The TRW Workflow

PCAP

READ PARSE

SPLIT

EXTRACT CLOCK

Inbound
(To OSIS)

Outbound
(From OSIS)

TRW TABLES

ORACLE TABLES

CLASSIFICATION

DISPLAY DASHBOARD

STATUS MONITORING
Demo (static screen shot)

<table>
<thead>
<tr>
<th>SCANS Dashboard</th>
<th>Statistics Analytic</th>
<th>TRW Analytic</th>
</tr>
</thead>
</table>

**Host**

**17:23 (266.1 X)**

<table>
<thead>
<tr>
<th>Benign</th>
<th>IP</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>92.6.85.110</td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>89.44.100.124</td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>92.6.85.111</td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>192.168.222.4</td>
<td>Host</td>
</tr>
<tr>
<td></td>
<td>192.168.222.4</td>
<td>TCP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scanners</th>
<th>IP</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>47.159.3.217</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>151.105.170.96</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>151.105.170.118</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>51.200.71.104</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>130.85.252.72</td>
<td>TCP</td>
</tr>
</tbody>
</table>

**TCP**

**ICMP**

Benign: 0  Scanners: 4
Discussion

• Implemented a real-time scan detection algorithm using streaming data
  – Multiple oracles effective for TCP / ICMP / UDP
  – Runs at 100x bandwidth capability (slowed for demo)
• Oracle provides dynamically updated information about network composition
  – Provides real-time attack detection and long-term situational awareness
• Integration with existing systems
  – TRW diagnostics can feed firewall or router ACL list to block scanners & inventory benign users
• Long term use requires oracle and table maintenance functionality to be added.
2. Continuous Statistics

- Implement situational awareness using statistics
  - Current statistics show current network behavior
  - Statistical models predict the network behavior
  - Significant departures from prediction raise alerts
- We calculate partial statistics from streaming data
- Partial statistics can be composed to form long term statistical models
- Our proof of concept implementation is simple but effective.
Building a Statistical Model

• Break traffic into one-minute *epochs* and accumulate data over each epoch
• Aggregate over various packet attributes
  – Examples: TCP flags, ports, ICMP Type & Code
  – Currently aggregate over ~260k dimensions
• Measure partial statistics (counts, squares) using *tumbling windows*
  – Developed aggregator which generates longer-term (1 hour horizon) statistical models from partial statistics
  – Calculate mean, \( \sigma \)
• Alert on excessive change in current observed values
PCAP

Punctor Windowed

Functor IP

Aggregate IP

IP Version
IP Protocol
IP Subnet

Functor TCP

Aggregate TCP Ports

TCP Flags

Functor UDP

Aggregate UDP Ports

Functor ICMP

Aggregate ICMP Msg/Code

MySQL/DBMS

Punctor Epoch Agg Union

Horizon Aggregator

Clock

Union

Display Dashboard

Statistics Workflow
Demo (static screen shot)
Statistics results (72 hrs Jan 1-3 2012)
Selected spikes – MySQL results

- TCP at 2012-01-01T17:54:00
  - 8M pkts in peak minute,
  - port 80 SYN from 204.145.0.0/16 anonymized
- UDP at 2012-01-02T14:39:00
  - 1.2M pkts in peak minute
  - port 22 (no comparable TCP activity at this time)
- ICMP at 2012-01-03T14:35:00
  - spike is “port unreachable” (3,3)
    - Back scatter from a SYN flood (spoofed source) ?
  - baseline is mostly “ping”
Overall Results / Conclusions

• Using streaming data…
  – We can implement automated attack detection / response i.e. scan detection / blocking
  – We can acquire situational awareness by collecting partial statistics and combining them into statistical models
• We can generate both real-time alerts and long-term situational awareness from the same data
• Our implementation is efficient, can run at higher rates.
  – unable to use InfoSphere Streams SPL’s distribution as it does not support our multicore, shared memory, architecture.
Rolling our own

- InfoSphere Streams uses a fairly heavyweight IPC based on Corba Middleware for parallelism.
- This is not bad if the computation to communications ratio is high.
  - Our analytics execute a few instructions per packet
  - Communications costs are much more
  - Packet level parallelism or pipelining is not effective
- We want a platform that can use inexpensive IPC on multicore shared memory processors as well as work effectively in a single thread.
- Thus Fathom …
The Fathom platform

- Fathom is RedJack’s platform for implementing streaming analytics.
- It has both sensing and analytic components.
- Initial driving application is a re-implementation of RedJack’s AMP (Analytic MetaData Producer) platform. This implementation is called Ampmill.
- Ampmill produces a variety of aggregated data products
  - TCP stack analysis
  - DNS analysis
  - HTTP banner capture
  - etc.
Comparison with AMP

• On a single threaded platform, Fathom is about 10% faster than the original AMP implementation.
• It also uses less memory giving additional platform headroom.
• AMP data supplements and extends traditional NetFlow capture.
• We are incorporating flow capture into the AMP code
  – Take advantage of existing packet parsing
  – Resource usage will be substantially less than separate AMP and flow capture programs.
Flomill

- Flomill is an implementation of a flow capture program used by one of our customers.
  - biflow collection – Ascii output records, heavy on packet statistics, payload & wire lengths, etc.
  - 32 pkt TCP flag history (27 early, 5 late) adjustable

- Work in progress.
  - Current performance for file playback is 0.6s-0.7s for a 512MB pcap file (Predict IARPA 2005 dataset) or about 6 gbs
  - Improvements clearly possible
    - hash tables and memory pool ops sub optimal
    - printf for output slower than binary
The workflow

• Fathom is based on workflow descriptions. These can be coded as C programs or YAML scripts
• Types: characterize information flowing in system
  packet: !type
    type: fm_net_packet
• Connections: carry a type 1:1, 1:M, M:1, M:M
  ticked-packets: !connection
    properties:
      type: !ref packet
• Computations: operate on data
  – Data may come from connection(s) or “outside” src
  – Data may go to connection(s) or outside sink
The workflow (computations)

• Computations can be parameterized (workflow or cmd)

  assign-flows: !computation
  computation: fm_net_assign_flows
  properties:
  in: !ref ticked-packets
  out: !ref flows
  inactive_timeout: 120
  hash_size: 512
  hash_stats: false

  aggregate-flows: !computation
  computation: aggregate_flos
  properties:
  in: !ref flows
  out: !ref ascii_flos
  site: "test"
  sensor: "flo30"
  active_timeout: 1800
  wrap: 27
  out_file_name: ""
  out_dir_path: "/tmp/"
  hash_size: 512
  hash_stats: false
  file_stats: false
flexibility

• The whole workflow contains computations for packet capture, parsing, assembly, punctuation, etc.
  – Equivalent computations can be interchanged
    • Read pcap file, device, list of files, compressed files, etc.
    • Produce SiLK uniflows, ipfix biflows, Argus flows, etc.
  – Multiple computations can use same connection
    • ampmill and flowmill on same packet stream.
  – Punctuation allows actions to be keyed to inserted events in data stream.
    • File Rotation
    • Periodic statistics or reports
More flexibility

• Computation boundaries are useful distribution points
  – In a single thread, pointers are passed (no copy)
    • Doing this in SiLK showed major performance increase
  – In a shared memory multicore box, disruptor queues can be used to improve efficiency, stabilize response
  – ZeroMQ like transport between platforms. Avro to reduce volume, serialize wire traffic
  – Publish / Subscribe mechanisms can be used to reconfigure computations on the fly adding or removing analytics without rebuild or restart of the computational pipeline.
Future Work

• Assist analysts in a transition to streaming data and away from the “collect, archive, analyze” mode, provisioning ever larger data centers.

• Extend to disparate streams, logs, messages, etc. and use the streaming results to guide or temper archiving.

• Adapt to stream various data types / sources (non-cyber) on Continuous Statistics.

• Deploy streaming analytics near or at high volume sensors to aid in the triage of large data streams, perhaps discarding that which is understood to be benign allowing analysts to concentrate on the needles rather than the haystack.
Questions?
Contact Information

John McHugh
john dot mchugh at redjack dot com