Ending the Finger-Pointing Between Apps and Network Admins

Using Splunk Stream™ for Fault Isolation

David J. Cavuto, CISSP | Principal Product Manager, Data Ecosystem

September 2017 | Washington, DC
Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.
My Bio

► Bell Labs
  • Principal Engineer - Lucent VPN Firewall

► AT&T
  • Network security and analytics

► Narus
  • Product Manager – Narus Cyber Analytics

► Splunk
  • Sales Engineer, Security SME
  • Principal Product Manager – Splunk App for Stream
  • Principal Product Manager – Data Ecosystem Area

► David J. Cavuto
dcavuto@splunk.com
Presentation Overview

1. Problem Statement
2. What is Wire Data? What is Splunk Stream?
3. Splunk Stream – Product Overview
4. Splunk Stream – Architecture
5. Fault Isolation Methodologies
Problem Statement

- Many different elements of networks
  - Hosts
    - OS
    - Enterprise Software
    - App Software
  - Infrastructure
    - Routers
    - Switches
    - Wireless
- Often those elements are managed by different teams
- How do you fault isolate?
Background on Wire Data and Stream

The Ground Truth
What’s Wire Data?

- Network Conversations
- Machine data
- Poly-structured data
- Authoritative record of real-time and historical communication between machines and applications
Open Systems Interconnect (OSI) model
Published in 1984 by ISO and CCITT (now ITU-T)
Forms the basis for all modern network communication models
Hierarchical messages encapsulated as they go down the stack, and get decapsulated as they go up the stack

<table>
<thead>
<tr>
<th>Layer</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Application</td>
<td>HTTP, SMTP</td>
</tr>
<tr>
<td>6. Presentation</td>
<td>TLS</td>
</tr>
<tr>
<td>5. Session</td>
<td>SCP</td>
</tr>
<tr>
<td>4. Transport</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>3. Network</td>
<td>IPv4, IPv6</td>
</tr>
<tr>
<td>2. Data Link</td>
<td>Ethernet</td>
</tr>
<tr>
<td>1. Physical</td>
<td>Ethernet, WiFi</td>
</tr>
</tbody>
</table>
Wire data represents capture of true conversations between endpoints
It has the “omniscient view” of what actually transpired
The conversations contain the details about each transaction, including the time of occurrence
Less chance of interference
• Intentional / Malicious
• Load or resource based
Multidimensional / Multiresolution Data
Why Splunk Stream™?

Flow-type Data

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Application</td>
<td>Traditional Wire Data flow-type records (such as NetFlow) generally contains only IP addresses and TCP or UDP ports.</td>
</tr>
<tr>
<td>6. Presentation</td>
<td>While this can show host-host connections, it doesn’t give any insight about the content of those conversations (like telephone call records)</td>
</tr>
<tr>
<td>5. Session</td>
<td>Splunk Stream parses wire data all the way up the stack and generates Events with information at every level (more akin to a written transcript of a phone call)</td>
</tr>
<tr>
<td>4. Transport</td>
<td></td>
</tr>
<tr>
<td>3. Network</td>
<td></td>
</tr>
<tr>
<td>2. Data Link</td>
<td></td>
</tr>
<tr>
<td>1. Physical</td>
<td></td>
</tr>
</tbody>
</table>

Splunk Stream

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Application</td>
<td></td>
</tr>
<tr>
<td>6. Presentation</td>
<td></td>
</tr>
<tr>
<td>5. Session</td>
<td></td>
</tr>
<tr>
<td>4. Transport</td>
<td></td>
</tr>
<tr>
<td>3. Network</td>
<td></td>
</tr>
<tr>
<td>2. Data Link</td>
<td></td>
</tr>
<tr>
<td>1. Physical</td>
<td></td>
</tr>
</tbody>
</table>
Wire Data Collection / Metadata Generation

- End Users
- TAP or SPAN
- Protocol Decoder (Deep Packet Inspection)
- Events
- Request/Response
- Decryption (If Necessary)
Splunk Stream™ (7.1 - GA) Features

- **Packet Metadata Collection**
  - Collects elements of the application conversation
  - Can use live data from a tap or SPAN port
  - Can extract from PCAP files
  - 1GbE and 10GbE link options
  - Can collect directly on host’s inband interface

- **Targeted Packet and File Collection**
  - Collects “sessionized” bidirectional PCAPs
  - Extracts reassembled File Attachments also
  - Based on L2/3/4/7 Target criteria
  - Saved to customer-supplied NAS
  - Retrieval proxied by SH

- **NetFlow Ingestion**
  - Explicit Flow Collector for other flow sources
  - NetFlow v5, v9, IPFIX, jFlow, cFlowd, sFlow
  - Can aggregate ingested Flow data

- **Estimate Mode**
  - Deploy without collecting data

- **Commercial App Detection (300+)**
  - Works even if the app is encrypted

- **TLS/SSL Decryption (with certs)**

- **Aggregation Mode**
  - Statistics generated at endpoint
  - Equivalent to “stats sum(field1), avg(field2)” in SPL

- **Filtering at Endpoint (BPF)**

- **Out-of-Box Content**
  - Dashboards for common protocols

- **Distributed Forwarder Management**
  - All config centrally managed
  - Forwarder Groups
ProtocolsParsed with Stream 7.1

**Simple Transport**
- TCP
- UDP
- IP

**Infrastructure**
- ARP
- DHCP
- SNMP
- DNS
- ICMP
- IGMP

**File Transfer**
- FTP
- HTTP

**File Service**
- NFS
- SMB

**Email**
- IMAP
- MAPI
- POP3
- SMTP

**Messaging**
- AMQP
- IRC
- SMPP
- XMPP

**Authentication**
- Diameter
- LDAP
- RADIUS

**Database**
- MYSQL
- Postgres
- TDS (Sybase / MS-SQL)
- TNS (Oracle SQL*Net)

**VoIP**
- SIP
- RTP
- RTCP
Commercial Application Detection

- Add the many hundreds of applications to be detected to the TCP stream type existing “app” field
- Help diagnose the problem of “what is going over port 80”? And also “what’s taking all of my bandwidth?”
- DOES NOT PARSE applications, simply detects them
  - Will detect encrypted protocols!
  - Will detect vendor-proprietary protocols!
  - Uses empirical patterns, DNS, Cert CNs and other methods
- Current feature supports 300+ applications, many more to be added
300+ Commercial Applications Detected 😊
<table>
<thead>
<tr>
<th></th>
<th>Application Detection Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Application Service</td>
</tr>
<tr>
<td>2.</td>
<td>Audio/Video</td>
</tr>
<tr>
<td>3.</td>
<td>Authentication</td>
</tr>
<tr>
<td>4.</td>
<td>Behavioral</td>
</tr>
<tr>
<td>5.</td>
<td>Database</td>
</tr>
<tr>
<td>6.</td>
<td>Encrypted</td>
</tr>
<tr>
<td>7.</td>
<td>ERP</td>
</tr>
<tr>
<td>8.</td>
<td>File Server</td>
</tr>
<tr>
<td>9.</td>
<td>File Transfer</td>
</tr>
<tr>
<td>10.</td>
<td>Forum</td>
</tr>
<tr>
<td>11.</td>
<td>Game</td>
</tr>
<tr>
<td>12.</td>
<td>Instant Messaging</td>
</tr>
<tr>
<td>13.</td>
<td>Mail</td>
</tr>
<tr>
<td>14.</td>
<td>Middleware</td>
</tr>
<tr>
<td>15.</td>
<td>Network Management</td>
</tr>
<tr>
<td>16.</td>
<td>Network Service</td>
</tr>
<tr>
<td>17.</td>
<td>Peer to Peer</td>
</tr>
<tr>
<td>18.</td>
<td>Printer</td>
</tr>
<tr>
<td>19.</td>
<td>Routing</td>
</tr>
<tr>
<td>20.</td>
<td>Terminal</td>
</tr>
<tr>
<td>21.</td>
<td>Thin Client</td>
</tr>
<tr>
<td>22.</td>
<td>Tunneling</td>
</tr>
<tr>
<td>23.</td>
<td>Web</td>
</tr>
<tr>
<td>24.</td>
<td>Webmail</td>
</tr>
</tbody>
</table>
Data Estimate Mode (per-Stream)

- **Stream Estimate**
  - Host: Last 60 minutes
  - Graph showing Splunk index Volume (MB)
  - Chart displays traffic trends over time

- **Mode Selection**
  - Options include Estimated, Disabled, and Disabled

- **Estimate Data Volume**
  - Table listing enabled streams with traffic details
  - Metrics: Total Events, Total Incoming Traffic, Total Outgoing Traffic, Total Traffic, Splunk Index Volume

Example stream details:
- **http**
  - Total Events: 18577
  - Total Incoming Traffic (MB): 19.55
  - Total Outgoing Traffic (MB): 40.28
  - Total Traffic (MB): 59.83
  - Splunk Index Volume (MB): 23.66

- **smb**
  - Total Events: 80
  - Total Incoming Traffic (MB): 0.00
  - Total Outgoing Traffic (MB): 0.00
  - Total Traffic (MB): 0.00
  - Splunk Index Volume (MB): 0.00
Prebuilt Reporting

Get visibility into applications performance and user experience

Understand database activity and performance without impacting database operation

Improve security and application intelligence with DNS analytics
Architecture and Deployment
Collect and Monitor Data with Stream

- Stream has two deployment architectures and two collection methodologies

**Deployment:**
- Out-of-band (stub) with tap or SPAN port
- In-line directly on monitored host

**Collection:**
- Technical Add-On (TA) with Splunk Universal Forwarder (UF)
- Independent Stream Forwarder using HTTP Event Collector (HEC)
Deployment: Dedicated Collector

- End Users
- Internet
- Firewall
- TAP or SPAN
- Servers
- Search Head
- Splunk Indexers
- Linux Forwarder
- Splunk_TA_Stream
Deployment: Run on Servers

End Users → Internet → Firewall → Search Head → Splunk Indexers → Splunk

Physical or Virtual Servers
Universal Forwarder
Splunk_TA_stream
Physical Datacenter, Public or Private Cloud

Internet
Firewall
Splunk
Splunk_TA_stream
Physical Datacenter, Public or Private Cloud
Stream Forwarder Options
Makes it easy to add Stream anywhere in your environment

1. Stream TA
   - Stream deploys as a modular input on top of your Splunk Forwarders.

2. Independent Stream Forwarder
   - Stream deploys as a stand-alone binary and communicates via HEC.
   - Requires >= Splunk 6.3.1
Splunk Cloud Support for Stream

1. Stream forwarders fetch their configuration from the Cloud SH (authenticated)
2. Stream sends metadata back to Cloud indexers via the UF or HEC
3. Analysts connect to Cloud SH to explore the data collected by Stream

- Cloud Indexers
- Cloud SH
- Stream Forwarder
- UF + Stream TA
- Independent Stream Forwarder
- Corporate
Gain more deployment flexibility
Increase management efficiency with per-forwarder protocol control
Tailor data collection by assigning different sets of protocols to groups of forwarders
New Features in Stream 7.0 and Stream 7.1
Major New Features in Stream 7.0

- Splunk Stream™ 7.0 was released GA in November 2016
- **NetFlow Collector**
  - NetFlow v5, v9 (with template support), IPFIX (with vendor extensions)
- **MD5 Hashing**
  - Any parsed Stream field, including SMTP attachments and HTTP files
  - Integrates with Enterprise Security – Threat Intelligence Framework
- **Flow Visualization** for all IPv4 space
- **PCAP Upload** via SH and Continuous Directory Monitoring via Forwarder
- **Enhanced Metadata** Fields (eg FlowID, Protocol Stack, Event Name)
- **Configuration Templates**
  - Easier integration with other Splunk products
Flow Collection

- Active Flow listening socket on Stream Forwarder
- Flexible Configuration Options
  - Selectable fields and filtering
  - Can configure multiple, distinct listening ports on each Stream Forwarder
- Supports most common versions of Flow protocols
  - Cisco NetFlow, Juniper jFlow, HP sFlow, cFlowd
  - NetFlow v5, v9, IPFIX
  - V9 with templates (standard and custom)
  - IPFIX with vendor extensions
- Aggregation of Flow records (pre-indexing) can dramatically reduce the number of Splunk Events created
- Performance > 465,000 flows/second (on a single Independent Stream Forwarder)
Flow Collector Data Flow

1. Netflow enabled devices
2. Export Netflow (over UDP)
3. NetFlow Metadata captured by Stream
4. Events in Splunk Indexer / Search Head

NetFlow Collector

- NetFlow listening sockets (UDP ports)
- Actively capture Flows from Netflow v5, v9, IPFIX
- Creates Splunk-compatible Flow Records
- Management from Stream Centralized UI
NetFlow and sFlow Streams UX
MD5 Hashing of Files

- File Hashing provides integrity verification of files, can be used for a number of security use cases
  - inbound malware detection
  - outbound data loss prevention
- Stream generates MD5 hashes equivalent to “md5sum” unix command after decoding content back to binary
- Specifically for SMTP file attachments and HTTP
- MD5 hashes generated with Stream integrate directly into the Threat Intelligence framework of Enterprise Security, and has been tested with ES
- As a bonus, *any* non-numeric field can be MD5 hashed using the “Extract New Field” option. Field can be length-truncated if desired.
MD5 Hashing Data Flow

1. File Transfer Traffic between Client and Server directed towards Stream
2. Stream generates MD5 hashes of files, sends to Splunk Indexers
3. MD5 hashes compared against Threat Intel from public databases

MD5 Hashing

- Used to enable DLP and Security use cases
- Examines both inbound and outbound data transfer
- Can be used to find IOCs as well as data exfiltration
- Better metric than file names or file types
Flow Visualization

- Designed to show limited Client->Server interaction for IPv4 address space. Overview and Detail views
- Can be used in real-time, interactive, and forensic modes
- Bubble chart that animates as flows appear (Detail view only)
Flow Visualization Detail View

The Bubbles animate in real-time or in play-back mode

Vertical Trends illustrate your internal host address space

Horizontal Trends show your externally-accessible hosts
Major New Features in Stream 7.1

Stream 7.1 was released GA in March 2017

1. Targeted Full Packet Capture
   - Use Case: ES analyst sees anomalous behavior with log or Stream metadata, requests full packet capture. Downloads full packet capture (PCAP) from Search Head into Wireshark for further analysis.

2. File Extraction
   - Use Case: File containing malicious attachment is downloaded via HTTP. MD5 hash automatically generated triggers ES Notable Event via Threat Intel framework. File is extracted and stored on disk for Analyst investigation.

3. SQL query parsing
   - Use case: Alert when a user is attempting to execute a SQL command to a table they shouldn’t be allowed to access
   - Use case: Look for SQL Injection or other SQL-based attacks
Stream 7.1: Targeted Full Packet Capture

Explanation and Inspiration

- Stream 7.0 and earlier transforms wire data into Splunk events, digesting many packets into a small number of events.
- Most of the time, this is advantageous for troubleshooting because it preserves the salient features of the packets but eliminates all the redundancy.
- Occasionally, for security and other reasons, analysts need to see the full packets in the conversation →
Stream 7.1 Targeted Full Packet Capture

Functional Concepts

▶ “Targeted” because it doesn’t capture every packet it sees. The analyst specifies a set of criteria to use for capturing data, and only conversations that meet those criteria are fully captured

▶ Full Packet Capture: The full fidelity of the original packet-level conversation observed on the wire is captured and stored to a File Server (ie NAS), **NOT the Splunk indexer**

▶ Packets are stored in a sessionized format – meaning, the PCAP files on disk represent a single SRC <-> DEST bidirectional conversation

▶ Metadata (Splunk Events) is still generated and sent to the Splunk Indexer. These events contain links to the File Server where the packet file is stored

▶ A workflow action is created in the Splunk Search Head to download the packets to the Analyst’s browser (and into a PCAP reader, like Wireshark)
Packet Storage Process

1. Packets are observed by Stream
2. Stream generates Splunk Events (Metadata) for all packets
3. Some packets match Packet Targeting Expression (“Packet Stream”)
4. Conversations containing matching packets are sent across the network from Stream to a File Server using a standard FS protocol (SMB/CIFS, NFS, etc.)
Packet Retrieval Process

1. Analyst explores Stream metadata in Splunk Indexer
2. For metadata that has Packet Stream data, Analyst requests Packet Data via Event Action in Search Head
3. Search Head contacts appropriate File Server, automatically retrieves associated PCAP file
4. Search Head passes PCAP file to browser, which opens file in registered app
Stream 7.1: File Extraction

- Works in the same manner as Packet Capture
- Extracts files from HTTP and SMTP protocol
- Can simultaneously extract files and generate MD5 hash
- Saves files on File Server and allows Search Head Retrieval
Stream 7.1: SQL Protocol Parser

- Stream now includes a full SQL parser
- Dissects statements 8 different variants of SQL
- Extracts:
  - Command (INSERT, UPDATE, DELETE, SELECT)
  - Stored procedures (XP_*; SP_* etc.)
  - Database DDL (CREATE TABLE, DROP TABLESPACE, etc)
  - Table name(s)
- User name, row count, return code are already included in Stream 7.0

User: Jim executes
DELETE from TBL_EMPLOYEES where VALUE="Tom Smith"
Fault Isolation
Ideally, we’d like to test each element in isolation, to see if any specific element is misbehaving individually.

Two practical problems:

• 1) Don’t usually have spare equipment to isolate
• 2) Often the problem is caused by interactions between elements
Isolation Solution Strategy

Use Stream probes to explore traffic between elements
Don't forget to rate this session in the .conf2017 mobile app