SilkWeb – Analyze silk data through API and Javascript frameworks

Silkweb – Flocon Jan 2017
Copyright and license

Copyright 2016 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING institute MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MER- CHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CAR- NEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sci.cmu.edu.

CERT® is a registered mark of Carnegie Mellon University.
Presentation agenda

- Introduction and Background
- SilkWeb in a nutshell
- SilkWeb components
- SilkWeb in a world of frameworks
- Silk CLI capabilities in SilkWeb
- Use cases from NOC and SOC
- Demo on live data
- Limitations and way forward
Introduction and Background

- Authors: Vijay Sarvepalli & Dwight Beaver
- Sponsors: DOD, DISA
- Collaborators: MPW (ISP)
- Recognition of roles and support

---

Architecture

== can go either way

Automation

== needs API

Visualization

== lower TTL

---

1 TTL – Time To Learn.
**SilkWeb in a nutshell**

- SilkWeb is a web application software.
- SilkWeb is designed to simplify access to a SiLK data repository through network data access (JSON/XML over webservices)
- SilkWeb is built with the modern design patterns (AJAX, View-Controller)
- SilkWeb is NOT a standalone web-UI for Silk, it is designed to work with multiple modern software frameworks (SIEM/Dashboards etc.)

1 SiLK - the System for Internet-Level Knowledge built by SEI CERT division.
Components of SilkWeb (3-tier)

Browser

API clients

Proxy/Cache

Proxy/Cache

Silkapi

Web server

Silkapi

Web server

SiLK data

Presentation

Caching

Processing

Data

XML-over-HTTP
JSON-over-HTTP

XML
JSON
CSV

Binary

Interpreted
Dynamic
Untyped
High-level
Content-oriented

Caching
Authentication
Reuse
Scale

Class-oriented
Config-based
Lambda-calculus
Common-Gateway
App engine model

Scalable data
File-based
Binary-optimized
Compressed
SilkWeb in a world of frameworks
SilkWeb in DISA lab at SEI

- Modes of integrating SilkWeb to Dashboard
- iFrame, IWC widget, component widget, JSON API, XML API
- Consideration browser XSS and authentication
Silk CLI capabilities built into SilkWeb

- **Rwfilter simple searches**
  
  ```
  $ rwfilter --type=out,outweb --start-date=2003/02/19
  --scidr=10.1.2.0/24 --pass=stdout
  
  
  ```

- **Rwstats group by searches**

  ```
  $ rwstats --fields=sip --count=4 data.rw
  
  INPUT: 549092 Records for 12990 Bins and 549092 Total Records
  OUTPUT: Top 4 Bins by Records
  
<table>
<thead>
<tr>
<th>sip</th>
<th>Records</th>
<th>%Records</th>
<th>cumul_%</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.1.1</td>
<td>36604</td>
<td>6.666278%</td>
<td>6.666278%</td>
</tr>
<tr>
<td>10.1.1.2</td>
<td>13897</td>
<td>2.530906%</td>
<td>9.197184%</td>
</tr>
<tr>
<td>10.1.1.3</td>
<td>12739</td>
<td>2.320012%</td>
<td>11.517196%</td>
</tr>
<tr>
<td>10.1.1.4</td>
<td>11807</td>
<td>2.150277%</td>
<td>13.667473%</td>
</tr>
</tbody>
</table>
  
  ```

- **Rwstats with time-bin**

  ```
  --bin-time
  --bin-time=SECONDS
  
  Adjust the key fields 'sTime' and 'eTime' to appear on SECONDS-second boundaries (the floor of the time is used). When no value is provided to the switch, 60-second time bins are used.
  ```
Use cases and live demo

- DDOS workflow (MPW use case)
- Build entity graphs of compromised home routers
- Dyn DDOS analysis
- Building qualifiers to move to “Analysis Pipeline”
- Find unauthorized port/protocols
- Call JSON/XML data from API
- Call JSON data from CLI
DDOS workflow

FILTER ddos_udp_filter
  
PROTOCOL == 17

END FILTER

EVALUATION ddos_udp_sources

FILTER ddos_udp_filter

FOREACH DIP
  
CHECK Threshold
  
SUM BYTES > 100000000
  
TIME_WINDOW 2 MINUTES

END CHECK

OUTPUT_TIMEOUT 2 MINUTES

OUTPUT LIST DIP ddos_udp_sourceList

ALERT 1 TIMES 2 MINUTES

CLEAR ALWAYS

END EVALUATION

LIST CONFIGURATION ddos_udp_sourceList

SEVERITY 5

SEED "/var/spool/ddos/ddos_udp_sources.set"

OVERWRITE ON UPDATE

UPDATE 5 MINUTES

END LIST CONFIGURATION
Compromised home routers – entity graph
Dyn DDOS analysis

➢ The day that your tweets died
Screenshots basic search
Screenshot pivoting from D3 graph
Demo of Stats and summary by time

<table>
<thead>
<tr>
<th>stime</th>
<th>sip</th>
<th>dip</th>
<th>protocol</th>
<th>sport</th>
<th>dport</th>
<th>bytes</th>
<th>packets</th>
<th>duration</th>
<th>application</th>
<th>sensor</th>
<th>ty</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016-02-15</td>
<td>173.194.204.95</td>
<td>192.168.5.166</td>
<td>6</td>
<td>443</td>
<td>49,640</td>
<td>6,315</td>
<td>1</td>
<td>11.16</td>
<td>-</td>
<td>asa02</td>
<td>in</td>
</tr>
<tr>
<td>2016-02-15</td>
<td>173.194.204.95</td>
<td>192.168.5.166</td>
<td>6</td>
<td>443</td>
<td>49,644</td>
<td>7</td>
<td>1</td>
<td>1.46</td>
<td>-</td>
<td>asa02</td>
<td>in</td>
</tr>
<tr>
<td>2016-02-15</td>
<td>173.194.204.95</td>
<td>192.168.5.166</td>
<td>6</td>
<td>443</td>
<td>49,648</td>
<td>4,005</td>
<td>1</td>
<td>33.29</td>
<td>-</td>
<td>asa02</td>
<td>in</td>
</tr>
<tr>
<td>2016-02-15</td>
<td>173.194.204.95</td>
<td>192.168.5.166</td>
<td>6</td>
<td>443</td>
<td>49,646</td>
<td>11,332</td>
<td>1</td>
<td>243.77</td>
<td>-</td>
<td>asa02</td>
<td>in</td>
</tr>
</tbody>
</table>
JSON/XML API for other widgets to consume

{"gdata": [{"bytes": 64332509, "packets": 1128, "rowid": 1, "records": 1128, "stime/300": "2016-02-15 01:55:00-2016-02-15 02:00:00"}, {"bytes": 43597295, "packets": 1262, "rowid": 2, "records": 1262, "stime/300": "2016-02-15 01:50:00-2016-02-15 01:55:00"}, {"bytes": 54490113, "packets": 1055, "rowid": 3, "records": 1055, "stime/300": "2016-02-15 01:45:00-2016-02-15 01:50:00"}], "rows": 3, "query_conditions": {"end": "2016/02/15:01", "istart": "0", "out_type": "json", "classname": "all", "start": "2016/02/15:00", "sortby": "time", "stats": "stime/300", "sensors": ["asa02", "kansascity", "squid", "squidkc"], "iend": "3", "types": ["in", "inweb", "inicmp"]}, "stats_totals": {"packets": 21059, "length": 24, "bytes": 1323440288, "records": 21059}, "header": {"timestamp": "1455556536", "version": "1.57", "time_execution": "1.936805 seconds"}, "rows_searched": 21059}

<?xml version="1.0" encoding="UTF-8"?>
<o>
<header><timestamp>1455556582</timestamp><version>1.57</version><time_execution>1.823312</time_execution></header>
<query_conditions><end>2016/02/15:01</end><istart>0</istart><out_type>xml</out_type><classname>all</classname><start>2016/02/15:00</start><sortby>time</sortby><stats>stime/300</stats><sensors>{u'asa02', u'kansascity', u'squid', u'squidkc'}</sensors><iend>3</iend><types>{u'in', u'inweb', u'inicmp'}</types></query_conditions>
<gdata class="array">
<record><bytes>64332509</bytes><packets>1128</packets><rowid>1</rowid><records>1128</records><stime/300>2016-02-15 01:55:00-2016-02-15 02:00:00</stime/300></record>
<record><bytes>43597295</bytes><packets>1262</packets><rowid>2</rowid><records>1262</records><stime/300>2016-02-15 01:50:00-2016-02-15 01:55:00</stime/300></record>
<record><bytes>54490113</bytes><packets>1055</packets><rowid>3</rowid><records>1055</records><stime/300>2016-02-15 01:45:00-2016-02-15 01:50:00</stime/300></record>
</gdata>
<stats_totals><packets>21059</packets><length>24</length><bytes>1323440288</bytes><records>21059</records><rows>3</rows><rows_searched>21059</rows_searched>
</o>
Limitations and Way forward

- JSON/XML is noisy throttle and use wisely
- Test with command line and understand limitations
- Be careful with calculus
  - In-memory IPSets are used in lambda functions
- Move to your graphics platform once you understand D-3
- Use asynchronous to keep user engaged not to fool the analysis.
- Try it!