Augment your Security Monitoring Use Cases with MLTK’s Machine Learning

Oliver Kollenberg, Philipp Drieger
Splunk .conf19, Las Vegas, Oct 24th 2019
Since 12 years
In 4 locations
8 Petabytes storage
Net-zero carbon footprint

Splunk at EAGLE: IT Ops and Security
70 indexes
3,200 data sources
210 data types
Agenda

• Intro
• Machine Learning Concepts
• Use Cases
  • Find Changes in Proxy Communication
  • Identify System Compromises
  • Baseline Privileged Account Behavior
• Wrap up, Q&A
Project setup

► Started Q1 2019
► Project staff of four
► Intention:
  • Can we extend SecMon with ML, to:
  • Baseline the data center, analyze behaviors?
  • Find security relevant patterns?
Project Team

Markus
IT Sec deepwaters

Michael
ML concepts and fundamentals

Philipp
ML project & science coach

Oliver
Splunk> that!
Machine Learning Concepts
Typical Machine Learning Scenarios

Anomaly detection
Deviation from past behavior
Proxy Communication

Predictive Analytics
Classify malware communication
Identify System Compromises

Clustering
Behavioral Analytics
Protect Privileged Accounts
Machine Learning Project Timeline

Frame the Problem
E.g. Detect Anomalous Proxy Traffic

Explore and tune models
Unsupervised approaches first
Supervised approaches second
Labels from context info, third

Present and Validate
Interface results, interpret findings. Challenge the model with simulated attacks / anomalous behavior

Get, explore and prepare data. Add context.
Collect and consolidate all relevant heterogeneous data into a dataset to start modelling work + consider adding context information (labels)

Launch and Operationalize
Put the models live on framed problem data scope and collect findings
Use Case 1: Proxy Communication
Use Case One: Proxy Communication
Idea, Concept, Results

Idea: Can we profile system communications outbound via webproxy and detect changes in behavior to ultimately find unwanted outliers?

Concept
• Collect eight features (each by Source IP) onto summary index.
• One DensityFunction per feature. | fit every night.
• | apply density models in intervals. Historize by again collecting onto summary index.
• Dashboard visualizes findings, guides investigation workflow.

Results
• Changes are reliably reported.
• Findings are consolidated into meta alarms (aka “change of change”).
• Change caused by unwanted software (HTTP tunneling of SSH traffic) correctly detected.
Use Case One: Proxy Communication
Detected Malware Communication

**Detect the Unwanted:** HTTP tunneling of SSH traffic

- **Meta Alert:**
  Triggers on first change, then change of change.

- **Alerted on outlying characteristics:**

![Graph showing anomaly details and meta alert triggers.](image-url)
Use Case One: Proxy Communication
Concept Building Blocks: SPL-Foo

Three scheduled Searches:

Search #1: Base Collector and SummaryFiller
- Create eight feature fields (dcDstIp, dcDstNetloc, dcCategories, categories, dcUserAgent, UserAgents, sumBytesIn, sumBytesOut), splitted by _time (1h) and src_ip
- Collect onto summary index.

Search #2: Training / Model Fitter
- Train one DensityFunction per feature.
- Full | fit every night.

Search #3: Alert Generator, Delta Fields Generator and SummaryFiller
- AlertGen: Find outliers by | apply 'ing density models over latest timeslice (every few hours).
- DeltaFieldsGen: Also cover last month of data to generate additional feature “newUserAgent”, via | streamstats implementing a historybrain.
- Collecting again onto summary index, to historize and make available for dashboard.
Use Case One: Proxy Communication
Concept Building Blocks:
Scheduling the ML System

Dashboarding

Alerting

Search #1

Search #2 | fit DensityFunction into M

Search #3 | apply M

index=summary_alarming

index=summary_filler

index=_raw
Use Case One: Proxy Communication

SPL-Foo: Search #1 Base Collector and SummaryFiller

Scheduled to run every few hours

Retain multivalues by tokenizing

PCR: Producer-Consu mer Ratio

Write features

| stats instead of | timechart to not zero-flood

| stats count as countConnas dc(r_ip) as dcDstIp, dc(ut_netloc) as dcDstNetloc, dc(cs_categories) as dcCategories, values(cs_categories) as categories, dc(viruses) as dcViruses, dc(UserAgent) as dcUserAgent, values(UserAgent) as UserAgents, sum(bytes_in) as sumBytesIn, sum (bytes_out) as sumBytesOut by _time, src_ip, sc_filter_result

| fields - categories, UserAgents

| eval categoriesZipped_mvjoin(categories, 'tok'), UserAgentsZipped_mvjoin (UserAgents, 'tok')

| collect testmode=false addtime=true index=demoSPLFoo index=demoSPLFoo

| source=stash source=stash

Earliest time:

Latest time:

Time specifiers: y, mon, d, h, m, s Learn More
Use Case One: Proxy Communication
SPL-Foo: Search #2 Training / Model Fitter

Fit all eight Density Functions, one by one

Ensure learning starts late, after grace period of 2 days.

Runs once per day, nightly.

```r
3 | eval UserAgents=split(UserAgentsZipped, `tok`), categories=split
categoriesZipped, `tok`
4 | table _time, src_ip, sc_filter_result, countConn, dcDistIp, dcDistNetloc,
dcCategories, categories, dcViruses, dcUserAgent, UserAgents, sumBytesIn,
sumBytesOut, pcr
5 | search sc_filter_result=observed
6 | fit DensityFunction countConn by "src_ip" into app:Md1
```
Use Case One: Proxy Communication
SPL-Foo: Search #3 AlertGen, DeltaFieldsGen and SmryFillr

Run longer for historybrain

Historybrain: Find new values

Throw away events which were only needed for historybrain

Apply to get outliers. Versionize. Collect to historize.
Use Case One: Proxy Communication Investigation Workflow

1. Derivate Alerts: Act on these

2. Deriving from sum(outliers) (=scoring) per src_ip

3. See anomaly_score trend (per src_ip)

4. Drildown on specific src_ip: Check outliers by visually presenting base values and its delta

5. Drildown on specific src_ip: Check raw values
Use Case 2: Identifying System Compromises
Use Case Two: Webserver Monitoring
Identify compromises by their behavior

Idea
• Can we use MLTK to ease alert fatigue?
• Instead of further evolving attack signatures:
  □ Let’s find signals that identify successful compromises
  • A hacked webserver will change its behavior. How?
    • Files written where they should not, e.g. Webshell
    • Webpage or DB table modified
    • OS Command executed, e.g. cmd.exe
    • …

Concept
• Use Windows logs (standard and sysmon) to generate features:
  • [Webserver: File Written into Webroot by Webserver Worker Process □ Classic SPL]
  • [All Systems: Lsass CrossProcs □ Classic SPL]
  • Webserver: Suspicious commands started by webserver worker process □ ML
  • …
Use Case Two: Webserver Monitoring
Find Suspicious Commands started by W3Worker Process

Observations
• How can we find unusual command from w3wp.exe?
  □ “unsual” can be tricky, CommandLines have a high variance (e.g. date/time parameter).
• Instead, how about suspicious commands? But not every process binary is bad
  □ Need to consider whole CommandLine.
• A single command may not be significant enough
  □ Need to consider multiple commands, their sequence.

Concept
• Define a list of suspicious words to extract matches from the CommandLine
• Collect extracted matches into bag of words pattern of suspicious sequences
• Classify malicious sequences:
  ▪ fit TFIDF from bag of words
  ▪ Classify with fit LogisticRegression from a set of positive examples
• Identify malicious sequences with a supervised learning approach
• Maintain lookup of positive examples to train and refine the model over time and adapt to new situations
Use Case Two: Webserver Monitoring
Find Suspicious Commands

Apache webshell usage

SQL Server CmdInjection
Use Case 3: Baseline Privileged Account Behavior
Use Case Three: Baseline Privileged Account Behavior
Idea, Concept, Gain

Idea: Can we profile Windows admin logon sessions and detect changes in behavior to ultimately find suspicious outliers?

Concept

• Collect features describing Windows admin logon sessions onto summary index
• Cluster training data (last [30;1] days) using fit XMeans models every night. One model each for focusing on source (ip address), session (behavior) and target (host)
• apply models each day to testing data (last day). Historize by collecting onto summary index.
• Detect anomalies based on changes in cluster_distances (fit DensityFunctions and outputlookup percXX statistics) and cluster memberships (outputlookup account to cluster relations)
• Dashboard visualizes findings and guides investigation workflow.

Results

• Three types of anomalies detected: Source, Target and Session Metadata
• Enables detection of credential theft, malicious account usage (e.g. lateral movement), …
• Differentiation between local and global outliers: New accounts trigger only global alerts
Use Case Three: Baseline Privileged Account Behavior Investigation Workflow

1. Alerts overview (graphical)

2. Alerts overview (tabular)

3. Drilldown on Alert: View Details

4. Compare sources

5. Compare targets

6. Compare session features
Graphs Special
Graphs Special: Consider Relations Between Use Cases

Scenario: Compromise, C2. Gives these Anomalies:

- System Compromise
- Accounts connects to unusual System
- Session Anomaly, Account hijacked
- New Outbound Connection, C2

Relate e.g. via Accounts’ Connections. Find ConnectedComponents in the Graph:

Calculate AnomalyScore per System, Connection and per ConnectedComponents. Map Score to Color:
Wrap Up
Summary, Key Benefits and Takeaways

Key Benefits
► New SecMon capabilities:
► 20 Mio Console Cmds classified every day by machine
► Pattern analytics of core infrastructure components: Communications, compromises, privileged accounts
► Efficiency gained: Smoother day-to-day ops, more time for use case development.

Takeaways
► Classic SPL is here to stay
► ML vastly expands SecMon capabilities:
► Allows to look at the “grey” between the black&white
► Enables feature-rich use cases
► Easy tuning
► New data insights may impact project plan. Stay flexible.
Thank you!

Q&A

Oliver Kollenberg
Security Consultant
EAGLE DataCenter
SIEMENS AG
oliver.kollenberg ( a t ) siemens.com

Philipp Drieger
Staff Machine Learning Architect
Splunk Inc.
philipp ( a t ) splunk.com

http://siemens.com/datacenters