Characteristic Based Alerting

Requirements

- High sensitivity alerting to detect subtle attack techniques
- Low false positive alert creation
- Rules specific to individual devices for more accurate base lining

Premise

- Specific attack tactics and techniques will cause a detectable characteristic. These characteristics can be detected through targeted specific detection rules that baseline normal usage and detect outliers.
- The detection of a single outlier may be a false positive but the detection of two or more characteristics for a single tactic or technique on a single device is more likely to be a true positive.

Solution

- Detections based on attack tactic characteristics.
- Alerts requiring 2 or more characteristics detections per tactic/technique per device.
- Characteristic detections are grouped by attack technique or tactic. This enables greater confidence in alert generation by requiring more than one type of characteristic per tactic per host.

Characteristic Based Alerting

Characteristic based alerting is a strategy designed to detect more subtle attach techniques and tactics. As such it requires that the foundations of threat alerting are complete, that alerting levels are reasonable and manageable.

The Security Industry has conducted great efforts to reduce the alert fatigue caused by excessive alerting experienced by most security teams. Many of these efforts are not aimed at the root causes of excessive alerting but provide a remedy for the symptom to enable security teams to function in the near term. Most excessive alerts are either; false positives, or alerts that cannot be effectively responded to due to a conflict between policy, architecture, process and/or practice. These ultimately stem from 2 root causes:

1. Insufficiently tuned alerts/notables
    - Alerts need to be tuned/created such that the ratio of positives to false positives is extremely low.
    - Tuning needs to be conducted continuously and vigorously to ensure alerts are as refined and current as possible.
2. Inability to tune alerts/notables due to network design/use limitations
    - Networks that are inherently insecure or where policies are lax or not enforced will cause even perfectly tuned alerts to have a high false positive rate.
    - Security operating in isolation will not overcome this shortfall, SecOps working together with enterprise architecture will need to investigate and alleviate the experienced issues.

Overcoming these causes is extremely difficult to achieve for most enterprises but is essential before any sensitive detection strategy can be successfully implemented.

Alerts often focus on specific signature traits used in the attack. To avoid detection attackers will monitor threat intelligence and detection rules in security devices to determine the discovered signatures, then slightly alter their tools and infrastructure to avoid future detection.

Detection of attacker techniques by focusing on the characteristics of the technique will provide more sensitive alerting that is not susceptible to minor signature changes. To improve the sensitivity the characteristic should then be compared to the individual device's baseline. Combining both provides a targeted, with a dynamic threshold dependent on each device's normal operations. To accomplish this attack techniques, need to be dissected to determine what noise (above normal operations) can be detected and in what log entry these can be found.

As the detection of individual outliers can cause excessive alerting through false positive generation, detections should only be alerted on when 2 or more from the same technique are observed on a single device. This form of aggregation is very similar to the Splunk UBA threats and Splunk RBA, in that it requires an aggregated alert to be created from basic alerts before notifying the SOC, but differs in that it focuses on attack characteristics, not all alerts and user behaviours.

This strategy should be viewed as an augmentation, not a replacement for other strategies.

Defining Characteristics

Entity

Defining an attack's characteristics must look at the nature of the attack method. Characteristics need to be focused on a single aspect of a service and this becomes the characteristic's entity, in the case of web shells the entity can be the HTTP endpoint expressed as the URL.

Group

Comparisons of an entity to its peers is useful to determine if the entity is malicious. Entity groups need to be defined as narrowly as possible to ensure that their behaviours are sufficiently similar, but broad enough to create a population. For web shells the group could be defined as all the HTTP endpoints on a given server.

Characteristics

An attack method's characteristics should be limited to a single value for each detection alert when using simple conventional alerting methods like SIEM rules and SPL. Combining characteristics is possible when implementing multi-dimensional machine learning techniques. Characteristic could include such things as:

- the volume of bytes in a session
- the total length of a URI
- The number of devices that connect to a specific service, etc

Data Set

Alerts based on population statistics such as IQR or standard deviations require batches of events to generate sufficient statistics and are considered batch. For batch data sets the default suggested is to execute the query daily on a batch size of a month. This cadence and batch supplies offers sufficient data to create a decent population whilst offering sufficient response time for detection. These parameters can be tuned to accommodate individual enterprise requirements, such as it may be necessary to execute the search every 8 hours on a data set of 1 month.

Alerts that can be generated from a single event without the need for comparison against a population, or using pre calculated population parameters, are considered stream. Stream processing requires additional compute as the comparison is executed against each event separately. If time to alert is of considerable importance it is possible to convert all alerts to stream by pre-processing the population parameters, storing the results as a separate table, and referencing them individually using a real time search.

Detecting Outliers

Alerts are created detecting outliers from normal behaviour, focusing on the behavioural characteristic of the attack method. Sample distribution is first assumed to be normal, with outliers detected as beyond 2 standard deviations. Distributions found not to be normal (having a negative -2stdDev value) have outliers calculated as outside of 80% of the population (lowest 10% and highest 90%).

Future work will look towards the incorporation of ML to augment statistical baselining.

Use Case: Web Shell detection

Web Shell Alert

a. Characteristic: 2 or more Web Shell characteristic detections
   Entity: host/device
   Group: None
   Data Set: Batch
   File: cbaWebShellsAlert
b. Detecting outliers with the characteristics of a web shell may include legitimate outlier behaviour. Requiring the existence of 2 or more different characteristics greatly increases the probability that the behaviour is a web shell.
c. Tuning:
   i. Tuning should be conducted on the individual detection rules.
   ii. Where a higher degree of sensitivity is required, alerting should be conducted on individual characteristic detections.
d. Noise Reduction:
   i. Requiring 2 different characteristics
e. Noise: Poorly tuned characteristic detection rules will result in noise alerting from this rule.
f. Execute directly after the characteristic rules have executed and the results are created in a summary index. Count distinct characteristic tactics/techniques for each host, where the characteristic is limited to those that describe web shells.

Web Shell Techniques

1. C2 and/or Exfil through URI encoding.

a. Characteristic: URI Length
   Entity: URL
   Group: All URLs on the Server
   Data Set: Batch
   File: unusualUriLength

b. Web shells can use the URI to carry their C2 and Exfil. Using this method may create URIs with unusually long URI for that server URL. As systems generally operate using methods with a low deviation detecting URIs that exceed 2

standard deviations or exceed 90% of the population may indicate the use of URI encoded C2/Exfil.

c. Tuning:

    i. Particularly consistent systems/environments can have increased sensitivity by reducing the outlier factor from 2 standard deviations to 1, and from 90% of population to 80%, or as required.

d. Noise reduction techniques:

    i. Remove events where the distribution is narrow making clear outlier detection unreliable. range/avg*100 > 25

    ii. Only alert on events created on this day. eventDate = nowDate

e. Noise:

    i. Systems that heavily use REST may have a highly divergent URI length

    ii. Human generated ad hoc REST queries could create URIs of unusual length.

f. Baseline per server URI for normal length using normal distribution and 90 percentile. Alert for URI that exceeds 2 stdDev above norm or 90%.

2. HTTP Endpoints without Referrers

a. Characteristic: Referrer event count
Entity: URL
Group: All URLs on the Server
Data Set: Batch
File: lowReferrerCountByHost

b. Most HTTP traffic is referred from other pages. As web shells are attempting to remain unknown, referrals from other web pages are typically a characteristic for them.

c. Tuning:

    i. Systems that do not use referrers for HTTP connectivity should be excluded.

    ii. Specific whitelisted referrers can be added to the () in the search statement

d. Noise Reduction Techniques: Allow listing known services that do not use referrers. These can be discovered during a tuning phase.

e. Noise: Services that do not use referrers

f. Baseline each server across all ULSs for referrer count using normal distribution and 90 percentile. Alert for URI that exceeds 2 stdDev above norm or 90%

3. Uncommon, mistyped, or outdated user agents
    a. Characteristic: Unusually low UA count for a server
       Entity: URL
       Group: All URLs on the Server
       Data Set: Batch
       File: lowUAcountByHost

    b. Normal system user agents are automatically updated as the software is patched. Further as browsers and other systems are fairly standardised the user agents from these systems form a consistent group. Connections from user agents not within this group may indicate a new service, non-standard system, or potential malicious activity.

    c. Tuning:

        i. Known and acceptable user agents can be added to the () list in the search for exclusion from alerts.

    d. Noise Reduction Techniques:

        i. Connections referred by itself are removed.

        ii. Connections to/from home are removed.

    e. Baseline each server across all servers for UA count per server using normal distribution and 90 percentile. Alert for any URI where connections using a user agent exceeds 2 stdDev below norm or less than 10%

4. Web shells often use a separate URL from other services (newly created endpoint)
    a. Characteristic: Newly created URL on a server where this is unusual
       Entity: URL
       Group: All URLs on the Server
       Data Set: Batch
       File: newURIonStableServers

    b. Many web shells require the creation of a new URL for attackers to connect. For servers that heavily use REST this sort of behaviour may be normal and detection not possible.

    c. Tuning:

        i. Particularly consistent systems/environments can have increased sensitivity by reducing the outlier factor from 2 standard deviations to 1, and from 90% of population to 80%, or as required

        ii. Unstable servers have been removed by requiring an average age of more than one day – this threshold can be adjusted up if required.

    d. Noise Reduction Techniques:

i. Servers that heavily leverage REST may have URIs with an average age of 1 day or less. As the rapid creation of URIs are normal behaviour these servers are removed from detection of this characteristic.

e. Noise:

i. Systems that use REST but have a longer average age may result in excessive alerting. This can be overcome by adjusting the average age threshold for those specific servers.

f. New URL created on a server where this behaviour is unusual. Baseline URL age across the server, do not activate on servers that have a URL average lifespan of less than 2 days. Alert for new URLs detected with an age of 1 day.

5. Web shells often use a separate URL from other services (uncommon connections to the endpoint compared to group)

a. Characteristic: URI with low connection source count
Entity: URL
Group: All URLs on the server
Data Set: Batch
File: unusuallyLowSrcCountFromHostURI

b. Web shell endpoint are often only connected to by the attacker, resulting in a much lower count for connected src Ips.

c. Tuning:

i. Thresholds are set to 5% and -2StdDev, these can be adjusted as required

d. Noise Reduction Techniques:

i. Averaging by server should address unusually low administration for some devices

e. Noise:

i. Servers that are administered by a small number of sources very infrequently may result in false positives.

f. Baseline the number of src IP's connecting to a URL. Alert for URLs that exceeds 2 stdDev below norm or 05% of the server average.

6. Web shell C2 and/or exfil will produce increased session volume

a. Characteristic: HTTP session volume
Entity: URL
Group: All URLs on a server
Data Set: Batch
File: unusuallyLargeSessionHost

b. Web shells sessions may include data exfiltration and C2, this will result in an increase in session volume when compared to other HTTP sessions to the server.

c. Tuning:

    i. Thresholds are set to 95% and +2StdDev, these can be adjusted as required.

    ii. To increase sensitivity, alter the threshold to detect low session volumes, particularly in servers where large sessions are normal (file transfer servers, streaming services etc)

d. Noise Reduction Techniques:

    i. Detection uses both session time and/or byte count. Devices can be restricted to either, or require both for a detection as required.

e. Baseline per server session volumes. Alert for URI that exceeds 2 stdDev above norm or 90%