Characteristic Based Alerting

# Premise:

High sensitivity alerting based on attack tactic characteristics, requiring 2 or more characteristics of an attack tactic to trigger an alert.

Attack techniques will cause a detectable characteristic. These characteristics can be detected through the creation of detection rules that baseline normal usage and can detect outliers.

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

The Security Industry has conducted great efforts to reduce the alert fatigue 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.

Alert fatigue often results 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.

1. 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, as such strategies have been developed to sort the wheat from the chaff. Strategies such as risk-based alerting can greatly assist in sorting through the chaff, but the addition of an abstraction layer can cause particularly subtle attacks to go unnoticed.

Alerts often focus on specific signature traits used in the attack to detect the action. To avoid detection attackers will monitor threat intelligence to determine the discovered signatures and slightly alter these to avoid future detection.

Detection of attacker techniques by focusing on the characteristics of the technique, where this is not normal behaviour for the device will provide more sensitive alerting that is not susceptible to minor signature changes.

Techniques need to be dissected to determine what noise (above normal operations) will be made and in what log entry these can be found.

Tuning to reduce sensitivity can be achieved by employing a requirement for more than one characteristic alert for an attack technique to be present. 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.

# 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%).

# Use Case: Web Shell detection

## Web Shell Alert

1. Characteristic: 2 or more Web Shell characteristic detections  
   Entity: host/device  
   Group: None  
   Data Set: Batch  
   File: cbaWebShellsAlert
2. 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.
3. Tuning:
   1. Tuning should be conducted on the individual detection rules.
   2. Where a higher degree of sensitivity is required, alerting should be conducted on individual characteristic detections.
4. Noise Reduction:
   1. Requiring 2 different characteristics
5. Noise: Poorly tuned characteristic detection rules will result in noise alerting from this rule.
6. 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

### C2 and/or Exfil through URI encoding.

* 1. Characteristic: URI Length  
     Entity: URL  
     Group: All URLs on the Server  
     Data Set: Batch  
     File: unusualUriLength
  2. 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.
  3. Tuning:
     1. 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.
  4. Noise reduction techniques:
     1. Remove events where the distribution is narrow making clear outlier detection unreliable. range/avg\*100 > 25
     2. Only alert on events created on this day. eventDate = nowDate
  5. Noise:
     1. Systems that heavily use REST may have a highly divergent URI length
     2. Human generated ad hoc REST queries could create URIs of unusual length.
  6. Baseline per server URI for normal length using normal distribution and 90 percentile. Alert for URI that exceeds 2 stdDev above norm or 90%.

### HTTP Endpoints without Referrers

* 1. Characteristic: Referrer event count  
     Entity: URL  
     Group: All URLs on the Server  
     Data Set: Batch  
     File: lowReferrerCountByHost
  2. 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.
  3. Tuning:
     1. Systems that do not use referrers for HTTP connectivity should be excluded.
     2. Specific whitelisted referrers can be added to the () in the search statement
  4. Noise Reduction Techniques: Allow listing known services that do not use referrers. These can be discovered during a tuning phase.
  5. Noise: Services that do not use referrers
  6. 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%

### Uncommon, mistyped, or outdated user agents

* 1. Characteristic: Unusually low UA count for a server  
     Entity: URL  
     Group: All URLs on the Server  
     Data Set: Batch  
     File: lowUAcountByHost
  2. 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.
  3. Tuning:
     1. Known and acceptable user agents can be added to the () list in the search for exclusion from alerts.
  4. Noise Reduction Techniques:
     1. Connections referred by itself are removed.
     2. Connections to/from home are removed.
  5. 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%

### Web shells often use a separate URL from other services (newly created endpoint)

* 1. Characteristic: Newly created URL on a server where this is unusual  
     Entity: URL  
     Group: All URLs on the Server  
     Data Set: Batch  
     File: newURIonStableServers
  2. 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.
  3. Tuning:
     1. 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
     2. Unstable servers have been removed by requiring an average age of more than one day – this threshold can be adjusted up if required.
  4. Noise Reduction Techniques:
     1. 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.
  5. Noise:
     1. 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.
  6. 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.

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

* 1. Characteristic: URI with low connection source count  
     Entity: URL  
     Group: All URLs on the server  
     Data Set: Batch  
     File: unusuallyLowSrcCountFromHostURI
  2. Web shell endpoint are often only connected to by the attacker, resulting in a much lower count for connected src Ips.
  3. Tuning:
     1. Thresholds are set to 5% and -2StdDev, these can be adjusted as required
  4. Noise Reduction Techniques:
     1. Averaging by server should address unusually low administration for some devices
  5. Noise:
     1. Servers that are administered by a small number of sources very infrequently may result in false positives.

### 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.

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

* 1. Detecting a session that has as higher session byte count than is normal for that server

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