Tenable Event Correlation

May 5, 2015
Revision 2

Table of Contents

I. Introduction

Tenable Log Correlation Engine® (LCE®) component of SecurityCenter Continuous View® (SecurityCenter CV™) offers a variety of event correlation types to detect abuse, anomalies, compromise and compliance violations. It does this because logs often do not have enough information in them, or there are simply too many of them, to gain a complete understanding of what is occurring on your network right now.

This paper outlines the various types of event correlation available through SecurityCenter CV, what information is leveraged by the correlation and how this can be used to monitor security, compliance and risk on enterprise networks.

For each correlation technology considered, a high-level description of the data used for the correlation as well as why this correlation matters is included. Screen captures of SecurityCenter CV interfaces, example logs and dashboards are included as well.

An understanding of SecurityCenter CV and LCE terminology and basic operation is assumed as well as familiarity with real-time network logs produced by Tenable Passive Vulnerability Scanner® (PVS™) based on network monitoring of DNS, SSH, SSL and many other types of network activity.

As a standard in this paper, all event types are italicized, as in *network*. The different types of events are discussed with high-level examples in depth in the LCE Event Types section.

II. LCE Correlation Overview

Normalization

All LCE correlation is applied to normalized events. Logs arrive at the LCE via syslog messages or through one of the LCE's many clients such as the LCE Windows Client or the Tenable NetFlow Monitor. An LCE may be configured to drop logs that are not normalized or keep them on hand for full log searching, but only logs that are normalized will be processed for correlation.

A key concept of log normalization and events is that one log might be normalized into one event, but that event could in turn be consumed by many other correlation engines, which could create more logs and events. For example a login failure log could be normalized to an SSH-Login_Failure event, but this log could be the final event that causes the statistical anomaly engine inside the LCE to create a Statistics-Login_Failure_Anomaly event.

Another key concept is that each normalized log has a unique event name but a common event type. All events related to network traffic are given the event type of *network*. All events that have to do with system errors are given the event type of *error*.

It is important to understand the event normalization process that categorizes logs into generic categories because many of the LCE's correlation functions apply generically to all events or to just one event category. For example, the LCE's statistical anomaly engine applies to every possible event that the LCE can generate, but it only produces event names based on the event type names. Similarly, detection of brute force password guesses only considers *login-failure* event types and creates new alerts when a threshold of potential password guesses is reached.

Core Event Correlation

This paper divides LCE correlation into three major types. The first is Core Event Correlation. These engines are generic and apply to very large classes of event types. In many cases, they also apply to each other and have their own forms of feedback that create very unique types of events.

The following types of core correlation capabilities exist within the LCE (click on the header of each section to see more details):

First Time Seen Events

The LCE tracks every possible normalized event and creates a “never before seen” alert and log for each host when new events occur. This is useful to identify new types of errors and attacks on systems that have been running for a long time.

Continuous Activity Detection

The LCE will track the IP addresses that have had activity occurring continuously for periods of time of approximately 30 minutes for various event types and some discrete events. This is a remarkable method to detect determined attackers, systems sending spam, systems with numerous errors, virus outbreaks and much more.

Statistical Anomalies

The LCE models the statistical frequency of connections and events for each configured host and compares it to the history of that host over time. For sequences of events that indicate a large number of deviations away from normal, the LCE will generate Minor, Anomaly, Medium and Large alerts.

IP & DNS Reputation (Botnet Detection)

The LCE receives a daily update of highly accurate IP address and DNS names known to be associated with botnet propagation and command and control. The LCE then applies this information to normalized events to create new events that indicate botnet connections into and outbound from your network. Most log event types are supported for this type of correlation including IDS, web server, NetFlow, sniffed network traffic and authentication logs.

Detecting Valid Attacks in Intrusion Detection Logs

The LCE, as managed through SecurityCenter CV, will receive the latest list of vulnerabilities available for each host as determined by Nessus® scans, Nessus patch audits and continuous network monitoring with PVS. As LCE processes IDS events, it will consider if each attack could potentially exploit the target based on the vulnerabilities present. This facilitates identification of highly critical IDS alerts.

Change Detection

The last core type of correlation performed by the LCE identifies normal system logs that indicate change. Change comes in many flavors such as the detection of new software, a new user, new open ports, new systems, new running programs, modified Windows registry settings and more. Tenable constantly updates LCE’s list of what is considered “change” based on new logs that it can normalize.

Tactical Event Correlation

The second type of LCE correlation is tactical in nature. These correlations are narrowly focused on finding one type of distinct behavior whereas the “core” correlation functions apply to many different types of events and types. The types of behaviors are significant and dramatically add value to the security monitoring of any organization (click on the header of each section to see more details).

Determined Scan and Attack Detection

The LCE has a variety of “one to many” and “many to one” correlation rules that look for one host stimulating multiple behaviors on one target or one host stimulating one behavior on one target. When applied to

intrusion detection, login failure or web error logs, the LCE can identify a wide variety of “determined” and “low and slow” attacks.

SQL Injection Detection

The PVS passively logs SQL queries and sends these as log messages to the LCE. The LCE has normalization rules that parse the SQL statements to look for potential SQL injection attacks.

Network Outage and Crash Detection

As the LCE parses error messages and system reboots, it attempts to correlate network wide crashes. These not only indicate when there is an network-wide outage in a certain type of application, such as when everyone’s version of Thunderbird crashes, they can also indicate failed (or potentially successful) virus and malicious content attacks against web browsers, email clients and chat tools.

Automatic User Account Recognition

As the LCE parses logins for a variety of applications it will automatically learn valid accounts that have logged into services such as Secure Shell (SSH), Windows, VMware, Nessus and dozens of others. When new users are recognized, an alert is generated indicating the new account. For some applications, the LCE will recognize when certain accounts are locked out or invalid and alert when attempts are made to use those accounts.

New Hosts Port Scanning

As the PVS discovers new hosts on the network, it sends logs that are recognized and parsed by the LCE. The LCE also normalizes logs from network IDS and other devices that indicate port scanning. When the LCE recognizes a new host and then also sees that it is performing port scanning, it creates an alert. This can be used to recognize hosts that are added to the network that may have been infected by worms and viruses.

Worm Outbreaks

As the LCE parses port scanning logs, it tracks the source and destination of the systems involved. If the LCE observes a system that was just port scanned then starts scanning other systems, it reports a possible worm outbreak.

Successful and Unsuccessful Password Guessing

As the LCE records failed and successful login attempts, it tracks when a remote IP address crosses a threshold of invalid login attempts. High numbers of login failures can indicate brute force password guessing attacks. If the LCE observes a system that was flagged as performing brute force password guessing eventually succeed with a valid login, it will report a successful password guess.

Suspicious Proxy Detection

As the LCE parses network traffic from applications, NetFlow data and sniffed network sessions it looks for sequences of connections that indicate that a host is acting as a temporary proxy. As attackers break into the network and compromise systems, they often use these systems to attack other systems on the local network.

Questionable Outbound Connection Spikes

LCE uses the Crowd Surge plugin to watch various connection types and network event logs. Crowd Surge will alert if a large number of hosts in your network connect to a single external IP address. This could indicate spyware, malware, a worm or a botnet on your network, reaching out to phone home to a command and control server.

Summary and Activity Reporting

The final type of correlation provided by the LCE includes a variety of categorization, event summarization, network session tracking and user ID tagging. Each of these correlation types enriches the data that can be filtered, which provides a more comprehensive analysis of events (click on the header of each section to see more details).

User IP Address Correlation

As the LCE processes authentication logs, it can associate the source IP address of the login with the user account. Logs that do not have user IDs in them such as NetFlow, anomalies and even web browsing, can be associated with a given user for analysis.

Domain Query and SSL Certificate Summary Reporting

The LCE processes DNS lookups from BIND, Windows servers, web queries and from passive sniffing with the PVS. It also parses SSL certificates used in secure connections logged by the PVS. To aid in analysis of which domains and secure sites a host or user has visited, the LCE will periodically create a summary log of all visited information.

Process Executable Summary Reporting

The LCE receives information about program execution from Windows audit logs, Unix process accounting records and the Unix audit trail. For each system, the LCE will summarize all programs that have run or crashed in the past hour, and also all programs that have run in a 24 hour period.

III. LCE Event Types

The LCE normalizes events into a variety of types. New log parsing rules are written by Tenable's Research team and are constantly updated.

When support is added for a new log source, such as a new application, firewall or intrusion detection system, the Tenable analysts who write the rules assign each normalized log to one of these event types.

The types and event names are used generically by other forms of LCE correlation. For example, one type of botnet behavior the LCE can detect is when an IP known to have been turned into a botnet successfully logs into a system by associating a *login* normalized event with an IP address of a botnet. If Tenable adds new support to process login events from a device such as a new Cisco router, the LCE's correlation engine will automatically process *login* event types from this source for IP addresses that are known to be botnets.

access-denied

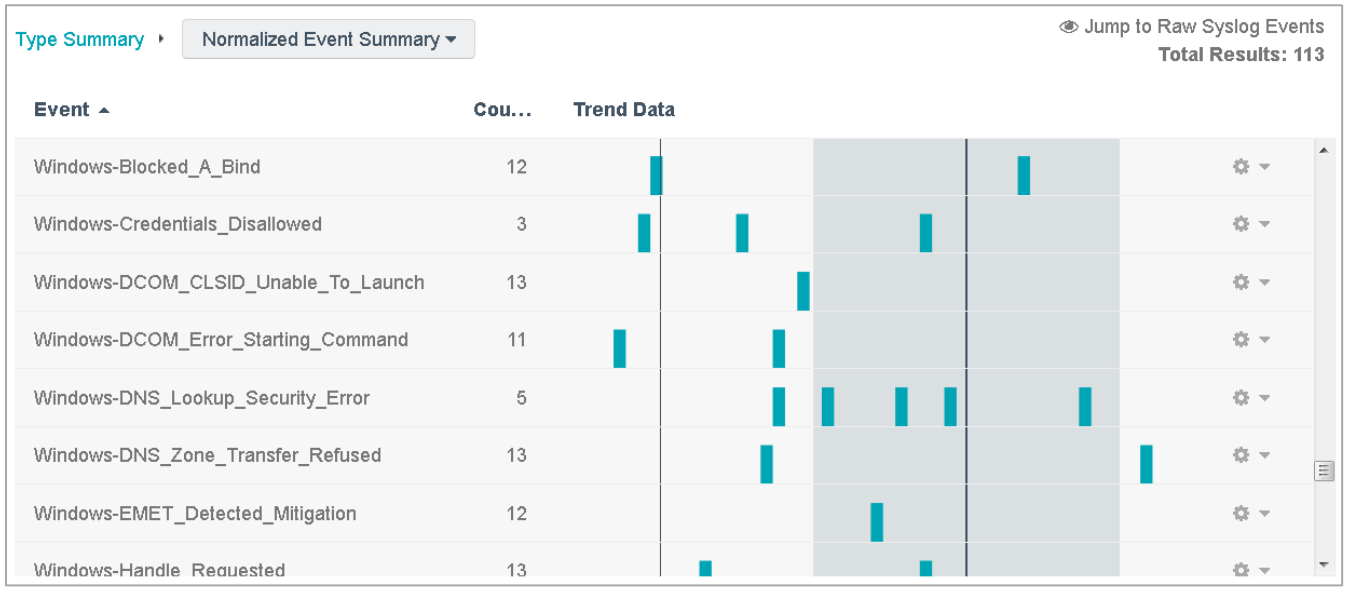
Flags attempts to retrieve objects, files, network shares and other resources that are denied. These events are distinct from authentication failures, blocked firewall connections and attempts to access web pages that do not exist that are respectively normalized to the *login-failure*, *firewall* and *web-error* event types.

Examples of some event types:

Event	Description
Juniper-Command_Execution_Error	A user was prevented from running a command.
FTP-Directory_Create_Attempt	A user was denied in an attempt to create a directory on an FTP server.
Sidewinder-Allowed_TCP	A Sidewinder firewall allowed and logged a TCP connection.
Bind-Denied_Version_Query	An attempt to perform a version query against a BIND DNS server was denied.
Windows-Privileged_Object_Operation_Failure	An attempt to access a Windows Object was denied.

Below is a partial screen capture of a 24 hour trend of *access-denied* events for an office network:

Drilling into the 1344 events, the following distinct events occurred:



Each of these events came from Windows event logs and indicated that some sort of action on the device was blocked.

application

Denotes logs from any application such as Nessus, Symantec Anti-Virus, SecurityCenter, the WU-FTP server, Exchange, sendmail, etc. that is noteworthy but does not indicate an error, a login failure, a connection, a restart of the application, an operating system event or a major function of the device.

For example, sendmail logs that indicate authenticated users are normalized to the login event type and sendmail logs that indicate spam email are logged to the spam event type.

Examples of some event types:

Event	Description
Bind-Transfer	A DNS server delivered a zone transfer.
Filezilla-Directory_Listing	A Windows-based FTP server had a user perform a directory listing.
Sendmail-Verify_User_Attempt	The sendmail application encountered an attempt to verify the existence of a user account.
MYSQL-Total_Allocated_Space	A MySQL database logged its allocated amount of disk space.
MSSQLSVR-Database_Unfrozen	An MS SQL server became unfrozen.
Nessus-Scan_Started	A Nessus scan started.
Windows-Defender_Scan_Started	Windows Defender started a virus scan.

connection

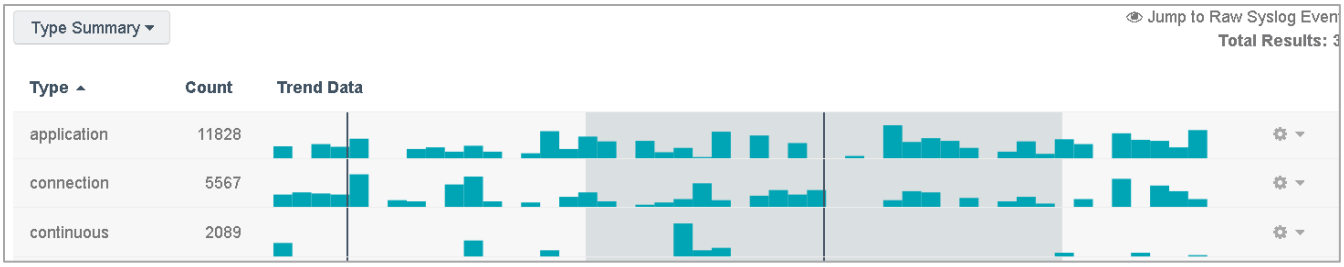
Notes any type of audited network connection logged by software that is not directly logged via the Passive Vulnerability Scanner (PVS), Tenable NetFlow Monitor (TFM) or the Tenable Network Monitor (TNM).

Event sources include allowed connections through firewalls, established VPN sessions and connections by applications such as Postfix. Web logs are explicitly normalized to event types of *web-error* and *web-access*.

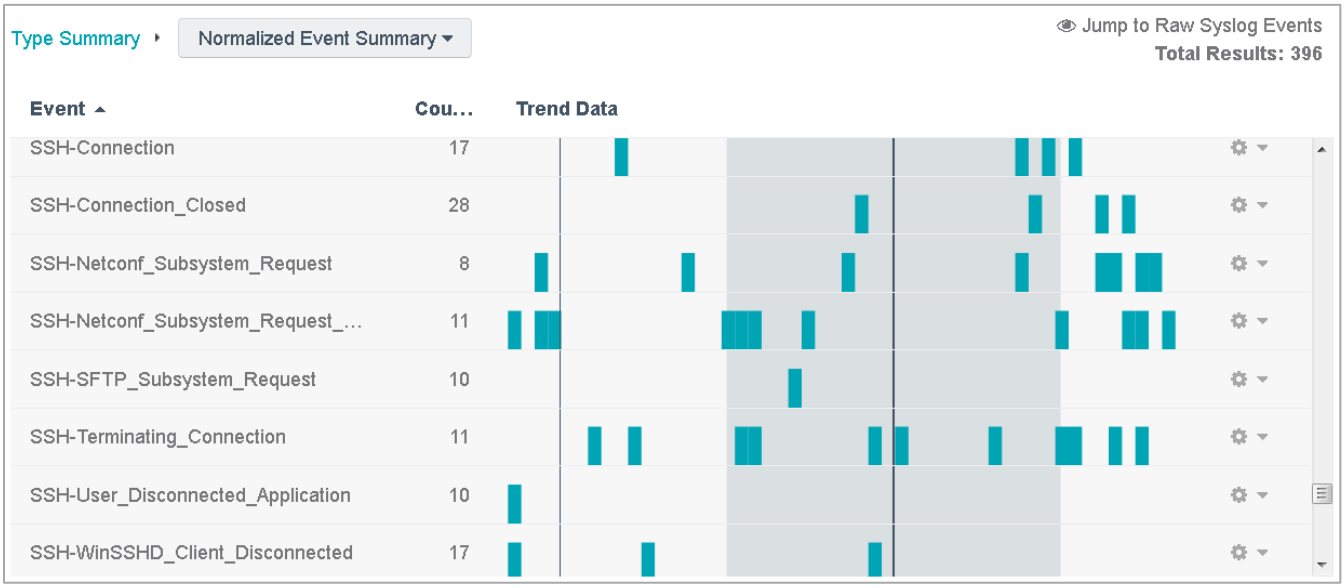
Examples of some event types:

Event	Description
Juniper-VPN_Session_Started	A VPN session has started and been logged by a Juniper VPN device.
Postfix-SMTP_Connection	A Postfix email application received a valid SMTP connection.
Sidewinder-Allowed_TCP	A Sidewinder firewall allowed and logged a TCP connection.

Below is an example type summary showing how *connection* events occurred during a 24 hour period on a server farm consisting of Unix, Linux and Windows servers, as well as multiple network device types:



Below is a screen capture of some of the corresponding *connection* events that contributed to the previous type summary:



These events indicate that the SSH process on one or more Linux servers are creating logs continuously and there are a few stray connections events from SSH and Windows servers during that same time period.

continuous

The LCE can identify hosts that are generating specific event types for periods of 20 minutes or longer. For example, a host may be infected with a worm and attacking small numbers of targets every five minutes.

Events of this type are a major form of LCE correlation and are covered in depth in the Continuous Activity Detection chapter.

Examples of some event types:

Event	Description
Long_Term_DNS_Failures	A host is likely performing vulnerability scans, attempting to send a large volume of spam email or has its DNS information misconfigured as it has been encountering DNS lookup errors for periods of 20 to 120 minutes continuously.
Long_Term_Error_Activity	A host has had errors reported from it for periods of 20 to 120 minutes continuously, which can indicate a major error.

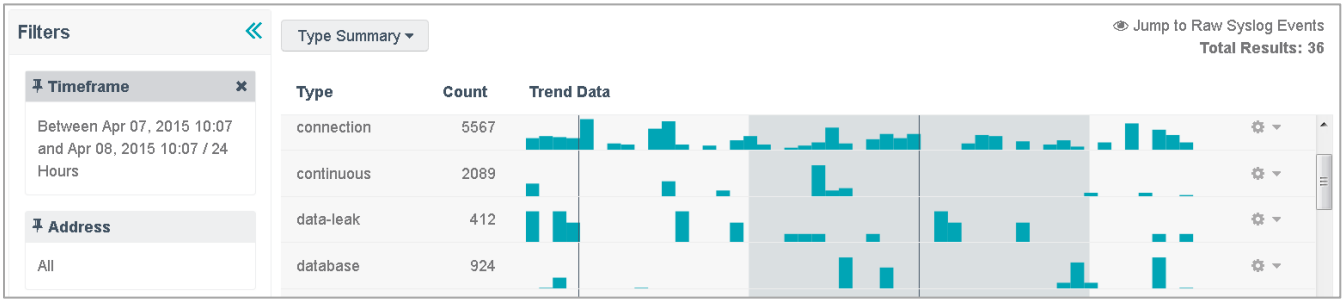
data-leak

Flags logs from the Passive Vulnerability Scanner or other Data Leak Prevention (DLP) products that indicate the presence of sensitive data such as a credit card or social security number. PVS must be specifically configured with DLP rules available from the Tenable Support Portal.

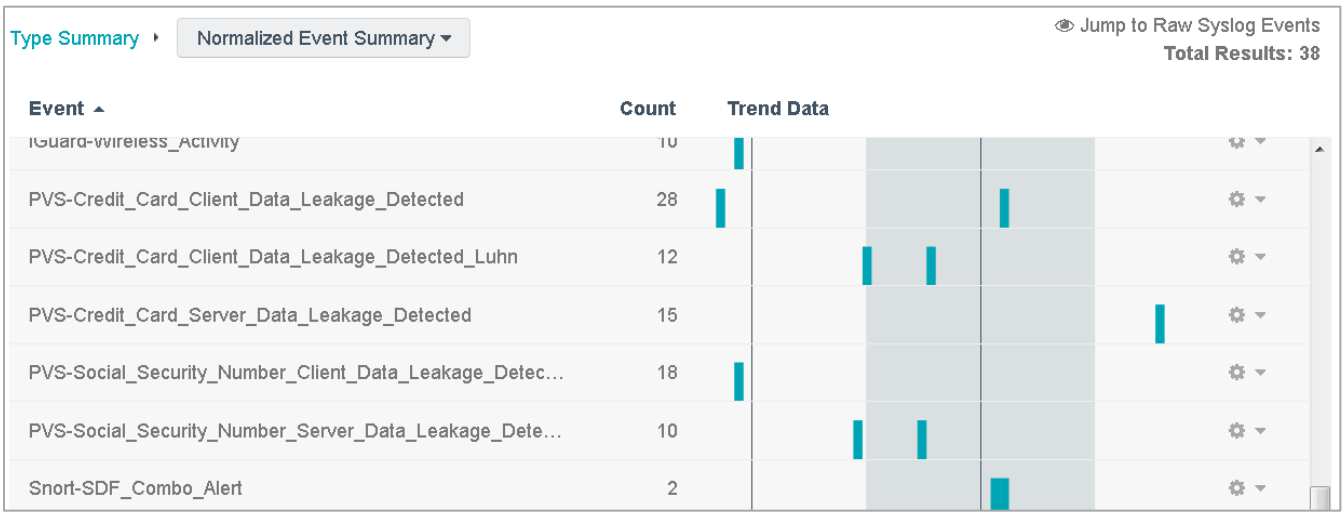
Examples of some event types:

Event	Description
PVS-Credit_Card_Detection	The PVS has detected network content that contained a credit card number.
iGuard-Skintone_Image	The McAfee DLP product detected an image that likely contains human skin tones and could be related to adult content.

Below is a screen capture from a network that has had 412 *data-leak* events occur throughout the past 24 hours:



A portion of these events were generated through traffic analysis and pattern matching by the PVS in a search for credit card and Social Security numbers:



These specific event types are associated with PVS logs that indicate credit card and Social Security number data was observed being sent over HTTP.

database

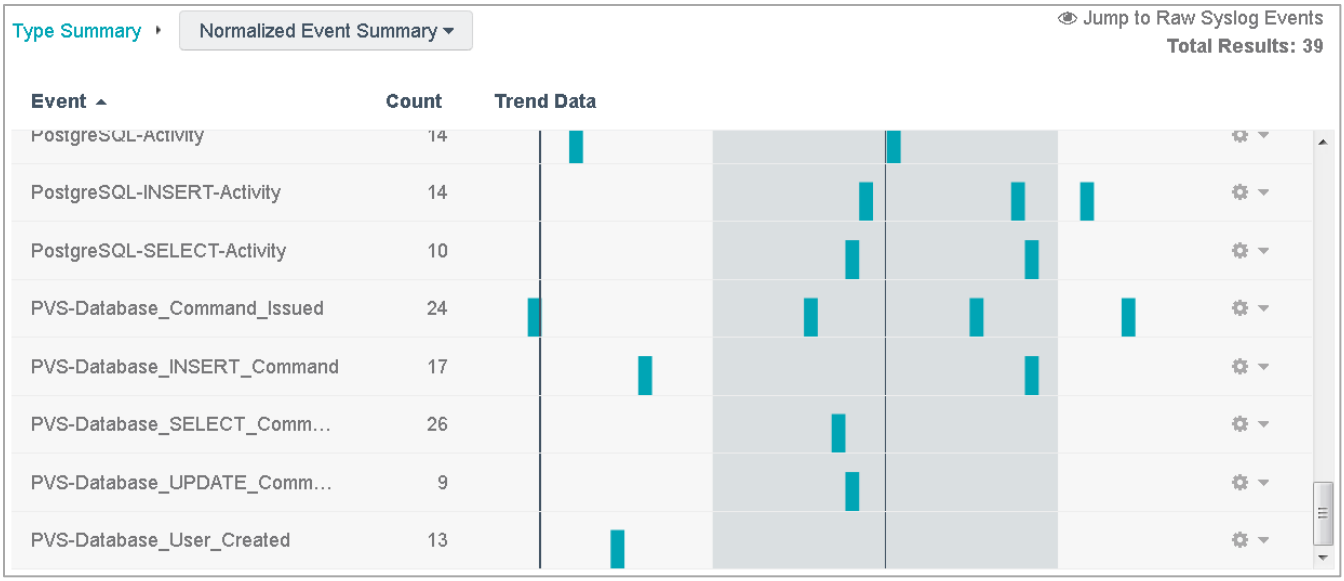
Denotes logs generated by the PVS from observed SQL queries. As the PVS monitors Oracle, MySQL and MS SQL network transactions, it creates logs that indicate a variety of database actions such as insertions and select statements.

Some observed SQL queries are processed by the LCE to look for potential SQL injection and other types of attacks against databases. These events are logged to the *intrusion* event category.

Examples of some event types:

Event	Description
PVS-Database_INSERT_Command	The PVS has detected an INSERT event into the database.
PVS-Database_CREATE_Command	The PVS has detected a CREATE event into the database.

Below is an example screen capture of set of *database* events from a server farm running Oracle and PostgreSQL, which is being monitored by the PVS:



The PVS monitors unencrypted SQL queries and database insertions through traffic analysis. In this screen capture, the bulk of the events are generated from the SELECT commands.

detected-change

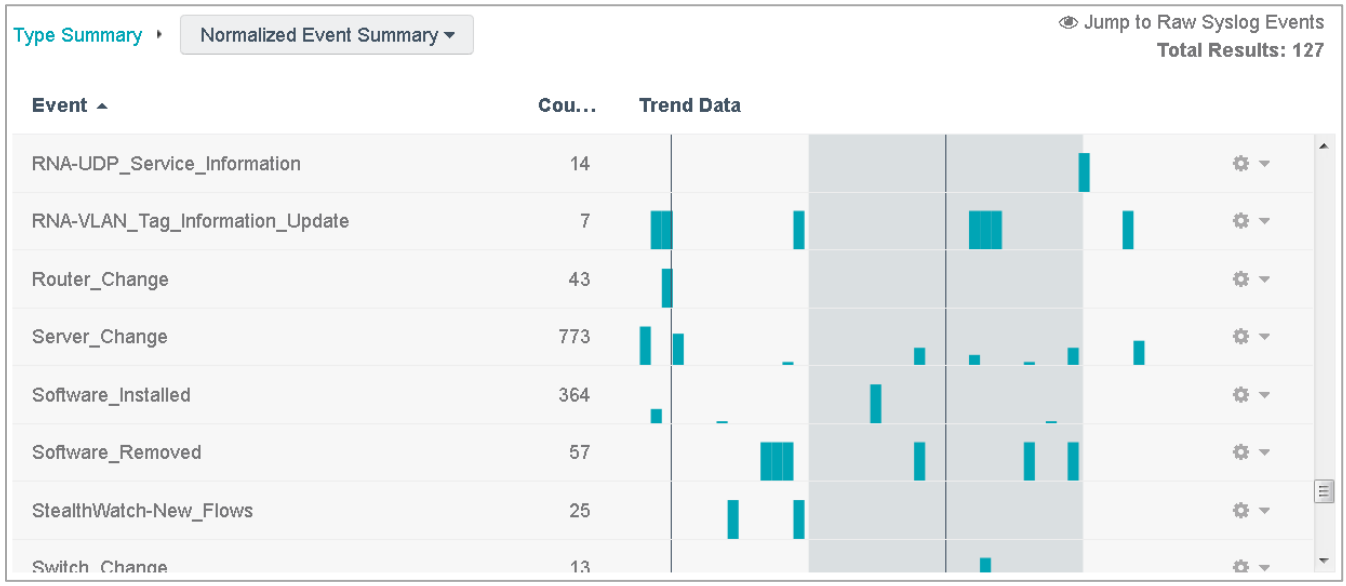
The LCE automatically recognizes many types of system events that indicate change and creates secondary higher level events. These events can be used to aid in reporting, alerting and creating dashboards.

Change detection is a major form of correlation performed by the LCE and is covered extensively in the Change Detection chapter.

Examples of some event types:

Event	Description
LCE-Windows_Executable_File_Modified	An LCE Client detected a file modification of an executable.
New_MAC	A new Ethernet address was encountered that had not been previously seen.
PVS-New_Port_Browsing	The PVS has detected a host that is browsing the network or Internet on a new port.
Router_Change	A configuration change to a router was encountered.
Software_Installed	Some sort of application or package was installed.

Below is a screen capture of 24-hours of detected-change events:



The majority of these events occurred by detecting network level changes, specifically from the PVS. The highest count of these *detect-change* events was Server_Change. Many Windows operating systems that make registry or Group Policy changes create logs that are normalized by the LCE as detected change.

dhcp

Logs from DHCP servers that indicate new leases are given the *dhcp* event type. Any type of other logs from DHCP servers such as operating systems errors, login failures, system status messages, etc. are normalized to other LCE event types. The events in the *dhcp* type events focus solely on monitoring of DHCP activity.

Examples of some event types:

Event	Description
DLink-Network_Computer_Assigned_IP	A D-Link router issued a DHCP lease.
DHCP-Request	A generic DHCP lease request was received.
Fortinet-DHCP_Config_Offer	A Fortinet firewall offered a DHCP lease to a host.

dns

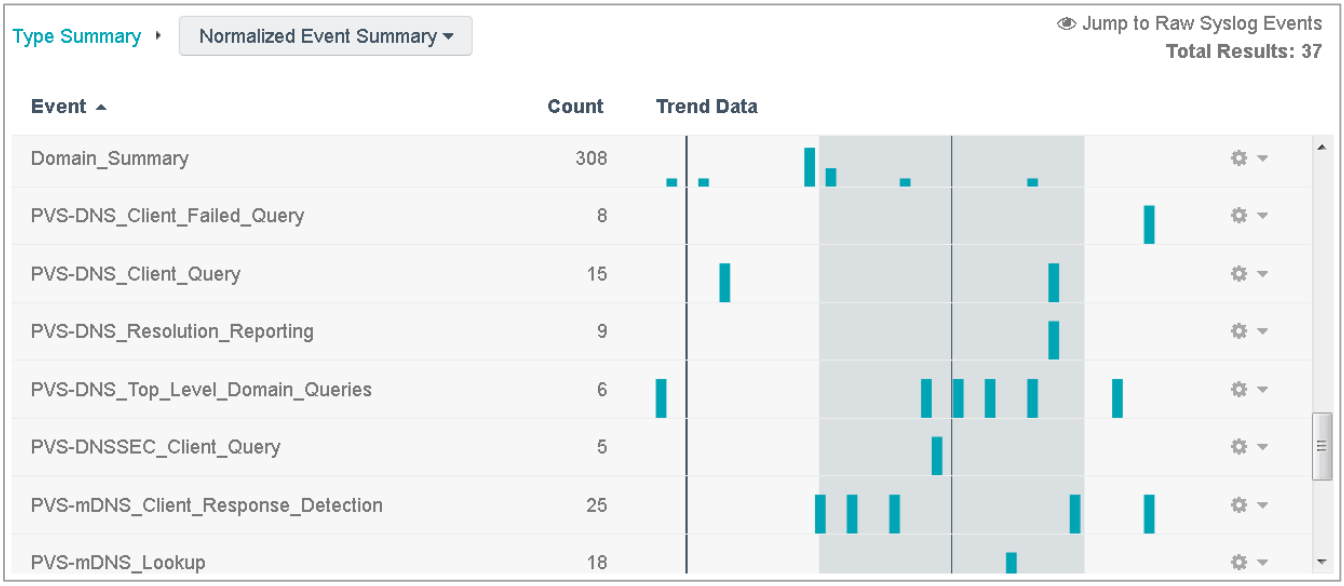
Denotes any type of log from a DNS server or from real-time network monitoring by the PVS, which indicates a DNS query or a DNS query lookup failure. LCE summary information, such as top domains visited by a host, is also logged in this event type.

Examples of some event types:

Event	Description
Bind-Query_IPv6	The BIND DNS server logged an IPv6 address request.
Domain_Summary	The LCE has summarized unique domain names queried for a host.
PVS-DNS_Client_Query	The PVS sniffed a DNS query and logged it.

Below is a screen capture of 24 hours of *dns* event type activity from a network that handles light amounts of email:

The specific event patterns that went into these 789 *dns* event types is shown below as well:



The DNS queries in this screen capture were detected by the PVS. Similar DNS query activity is recorded by BIND DNS servers. The Domain_Summary event is generated by the LCE and consists of recently visited top level domains that have been queried by a specific host. Summarizing DNS domain queries is covered in depth in the Domain Query Summary Reporting section.

dos

Denotes logs that indicate a denial of service event has occurred. These typically occur from network IDS detection engines such as Snort.

Examples of some event types:

Event	Description
NetscreenIDP-DDOS_Activity	The NetScreen (Juniper) IDP has detected a distributed denial of service attack.
WebTrends-Possible_SYN_Flood_Attack	A SYN flood attack was detected by the WebTrends application firewall.

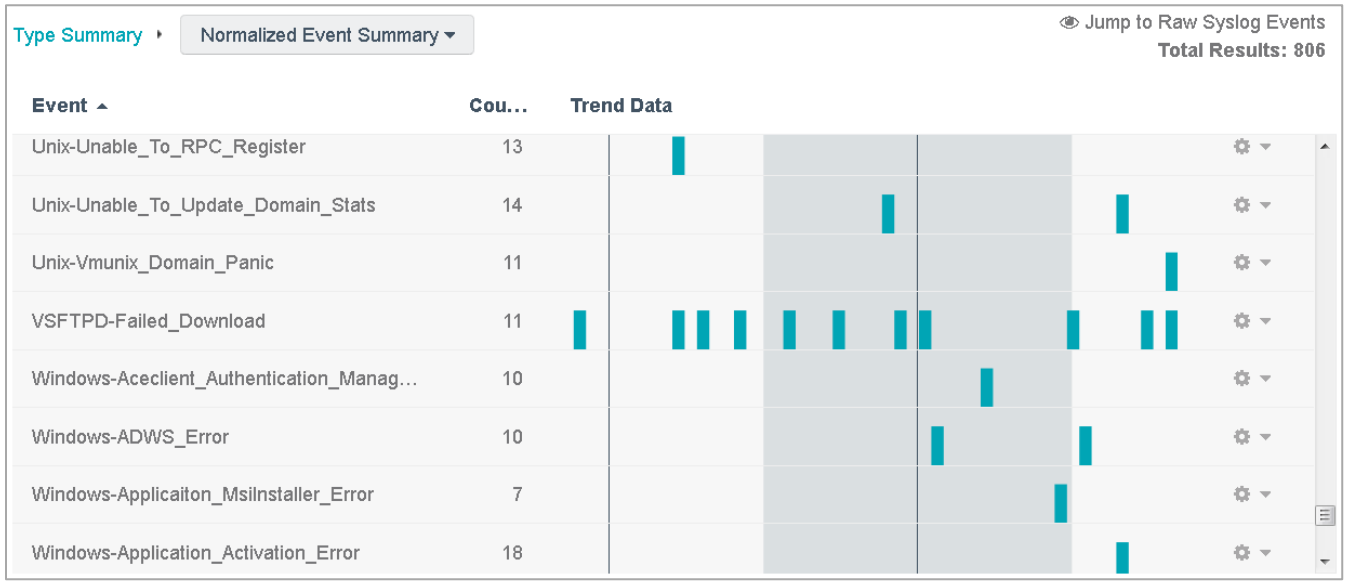
error

This event type denotes any type of system, application, router or switch log that indicates some sort of error. Logs that indicate crashes and hung processes from executable programs or daemons are sent to the *process* event type.

Example same event types:

Event	Description
CiscoWireless-Config_Error	A Cisco wireless access point (WAP) has encountered a configuration error.
CiscoASA-High_CPU	The CPU utilization level on a Cisco ASA firewall is high.
Fortinet-Firewall_Update_Failed	An update performed by a Fortinet firewall has failed.
Exim-Empty_SMTP_Message	The Exim email server encountered an email that was empty.
Linux-User_Exists	An attempt to add a user failed because the user account name already exists.
Windows-Print_Warning	An attempt to print a document from Windows encountered an issue.

Below is an example screen capture of errors from a mixed Unix, Linux and Windows server environment:



The errors in this section originated from a variety of sources including a Unix server, an error log from a VS FTP server and Windows error logs.

file-access

This event type denotes any type of sniffed PVS network session or log that indicates that a file was accessed, modified or likely retrieved.

Examples of some event types:

Event	Description
FTP-File_Upload	An FTP server logged a file being uploaded.
FTP-File_Renamed	An FTP server logged a file being renamed.
PVS-SMB_Client_DLL_File_Download	The PVS sniffed a file with a `.dll` extension being downloaded through Windows file sharing.
PVS-Web_File_7Z_Request	The PVS sniffed a file with a `.7z` extension downloaded over HTTP.
PVS-Web_Executable_RPM_Request	The PVS sniffed a file with a `.rpm` extension downloaded over HTTP.

Below is a screen capture from a small server network that hosted an anonymous FTP server:

Passively, both the SMB and FTP protocols were observed moving a variety of file types at various times throughout the day.

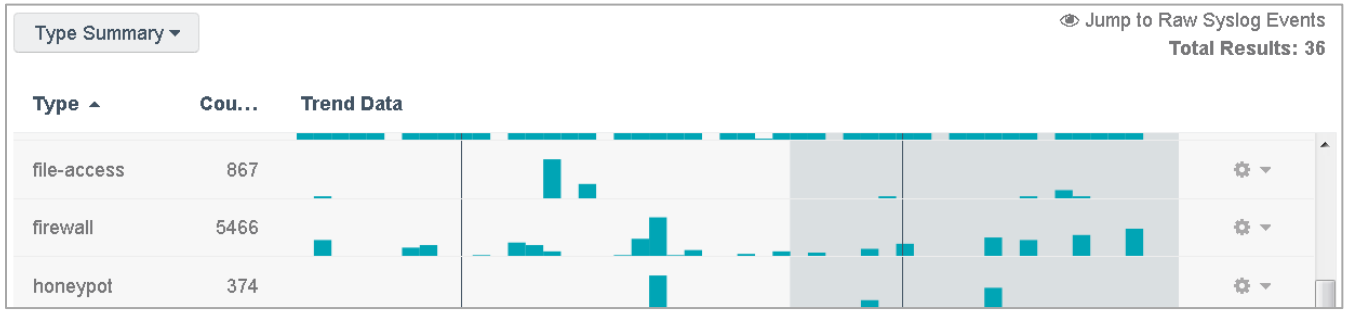
firewall

The event type denotes any type of log from a firewall, an intrusion prevention device, a router or a firewall, or application configured at the local host to specifically deny connections. Logs from a firewall about an incorrect configuration, administrator logins, port scan detection or errors would be normalized to other event types. Some web application firewalls (WAFs) have their events normalized to the *web-error* or *web-access* event types and not the firewall event type.

Examples of some event types:

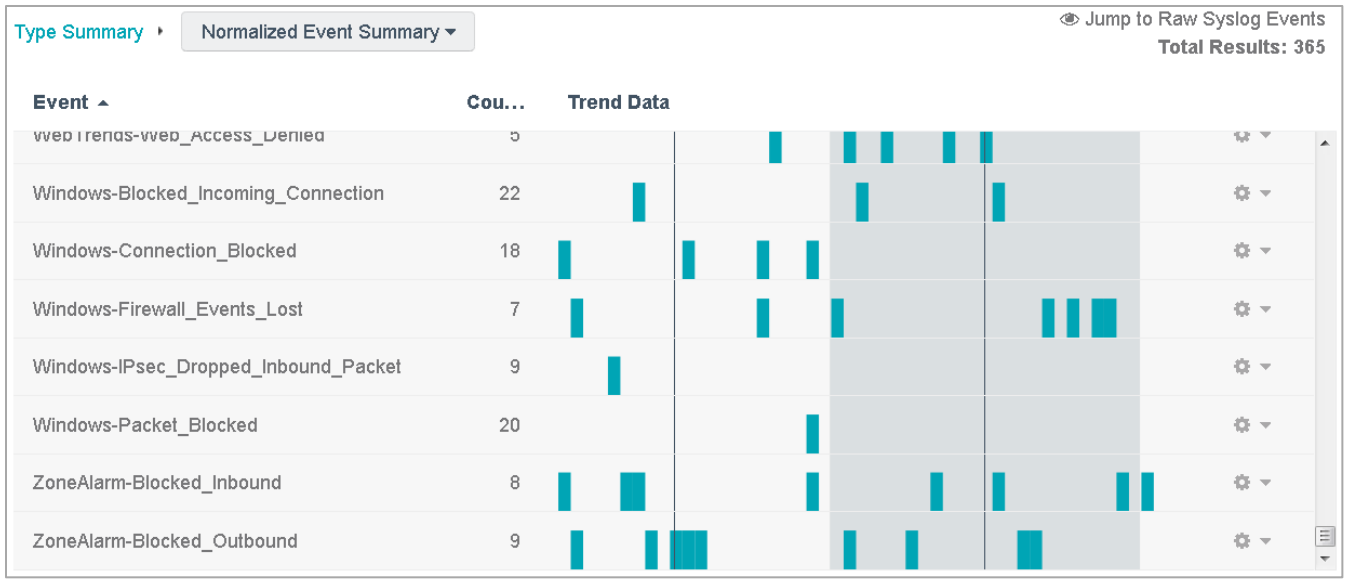
Event	Description
Checkpoint-Blocked_TCP	A Check Point firewall blocked a TCP connection.
CiscoASA-Blocked_ICMP	A Cisco ASA firewall blocked an ICMP packet.
Fortinet-Firewall_Virus_Oversized	An email with an attachment larger than a size set by policy was blocked by a Fortinet firewall. Fortinet firewall messages that indicate virus activity are normalized to the virus LCE event type.
Microsoft_Drop_TCP	A local Microsoft firewall denied a TCP connection.
TippingPoint-Block_UDP_Critical	The HP TippingPoint IPS blocked a UDP network session.

Below is a partial screen capture of *firewall* event types from a server farm for a 24 hour period:



The 5466 *firewall* events were comprised of the following specific event types:

In this case, firewall logs from a Cisco ASA device were being centralized and normalized by the LCE. Host-based firewall logs were also collected and normalized from the Windows operating system firewall and a ZoneAlarm application-based firewall, as shown below:



honeypot

This event type is reserved for normalized logs from applications designed to simulate networks, hosts and applications for the purpose of detecting intruders.

Examples of some event types:

Event	Description
Nepenthes-Critical_Alert	A critical log from the Nepenthese malware research toolkit has occurred.
Honeyd-TCP_Connection_Reset	The **honeyd** application reset a TCP connection.
Forescout-User_Mark	The ForeScout product has marked a user and will track if they return to the network.

Indicator

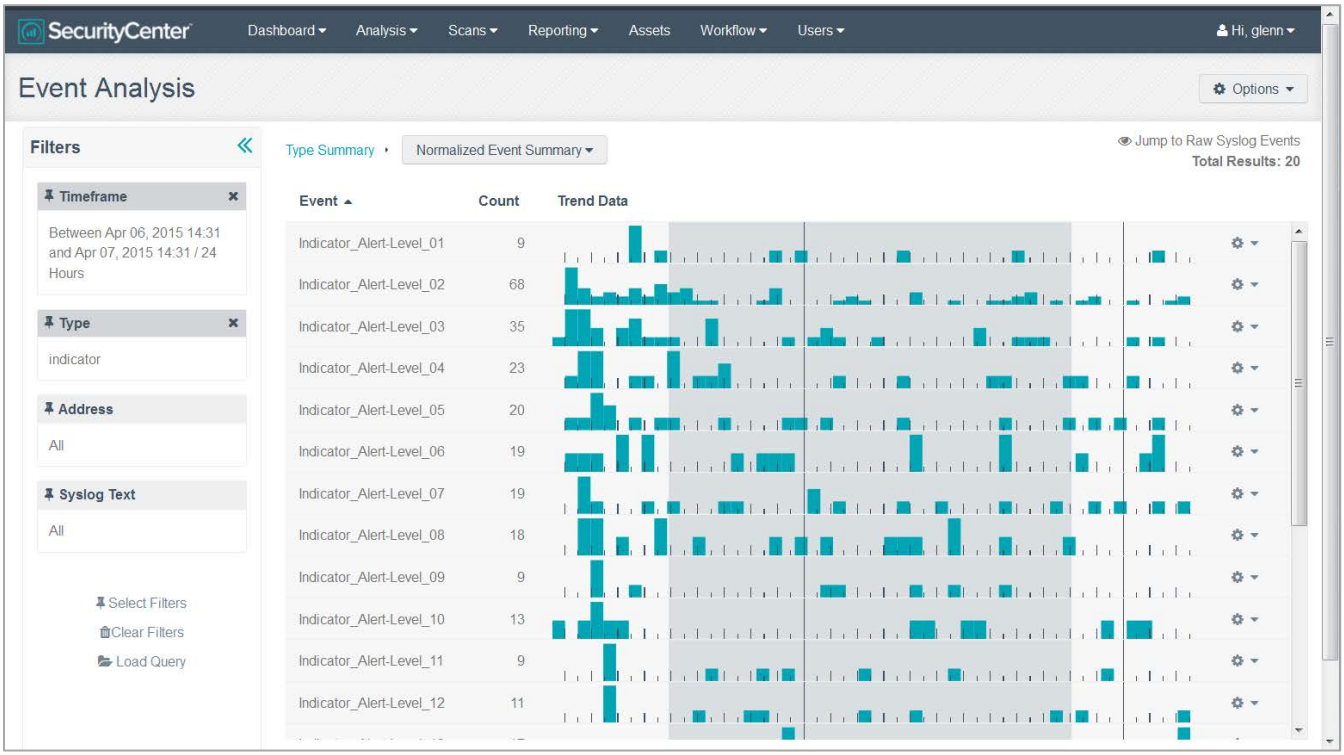
The "indicator" event type is used by LCE to track correlations associated with scanning, compromises, anomalies and other behaviors that indicate the presence of determined attackers, advanced malware and other forms of potentially malicious activities.

The tracking occurs for any IP address that has had at least two unique events occurring in a short period of time. The alert level is increased for each new type of event added into the tracking of the IP address. Each time an alert level is increased, a new log is produced summarizing the sequence of events.

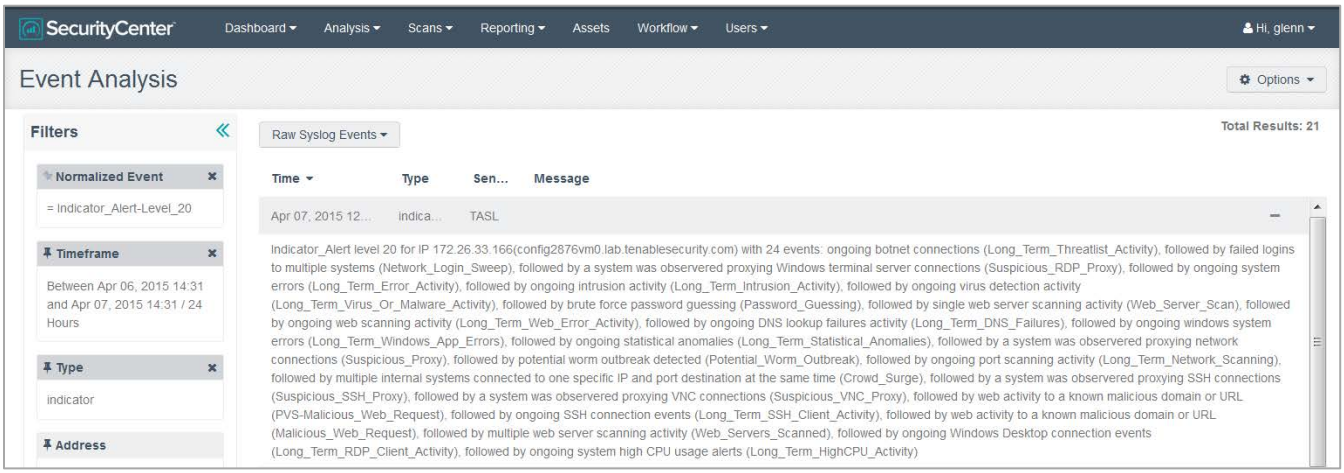
LCE tracks various different normalized event types that are shown in the “Indicator” type, including large anomalies. Also included is the threatlist (botnet) inbound, and outbound activity that indicates file transfers or proxy traffic. The “Indicator” type also tracks IDS events, continuous events, changes (network, account), downloads and other behaviors of interest.

When “Indicator” is selected, events will be displayed that look similar to the Indicator_Alert-Level_02 example below, which contains two events related to a single IP address:

```
Indicator_Alert level 2 for IP 10.31.15.203 with 2 events: ongoing virus detection
activity (Long_Term_Virus_Or_Malware_Activity), followed by brute force password
guessing (Password_Guessing)
```



The number at the end of an “Indicator Alert” is associated with the number of events that occurred related to one IP address. Selecting the “Raw Syslog Events” filter from the filter drop-down menu will reveal the raw syslog of the indicator type of events. Selecting the plus (+) symbol next to the specific indicator event will show the complete detail of the events that occurred related to a single IP address. It is suggested that “Indicator” type events be reviewed in descending order. The following example below shows an “Indicator Alert” with a level of 20, which is the highest type of indicator event that is currently possible in SecurityCenter CV:



intrusion

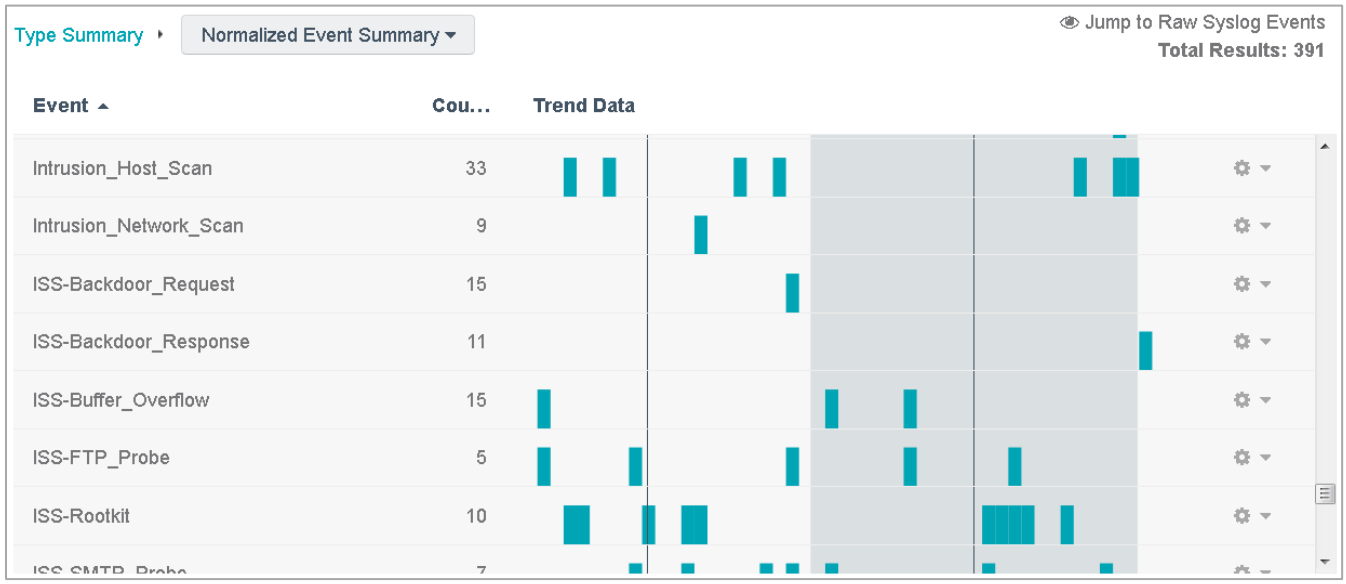
Denotes logs from network IDS, IPS, firewall, application and operating systems that indicate some sort of network attack. Post scans, denial of service and logs that indicate virus probes are normalized to their own LCE event types. Many applications will generate logs indicating they are under attack or have recognized an attack. The LCE’s normalization process assigns this type to any log that indicates an attack.

The LCE performs several types of correlation with events normalized from intrusion logs. These include correlating intrusion logs with known botnet addresses (see the IP & DNS Reputation section for more information), correlating vulnerabilities with intrusion attacks (see the Detecting Valid Attacks section) and identifying determined scans, regardless if they are low and slow or massive (see the Determined Scan and Attack Detection section).

Examples of some event types:

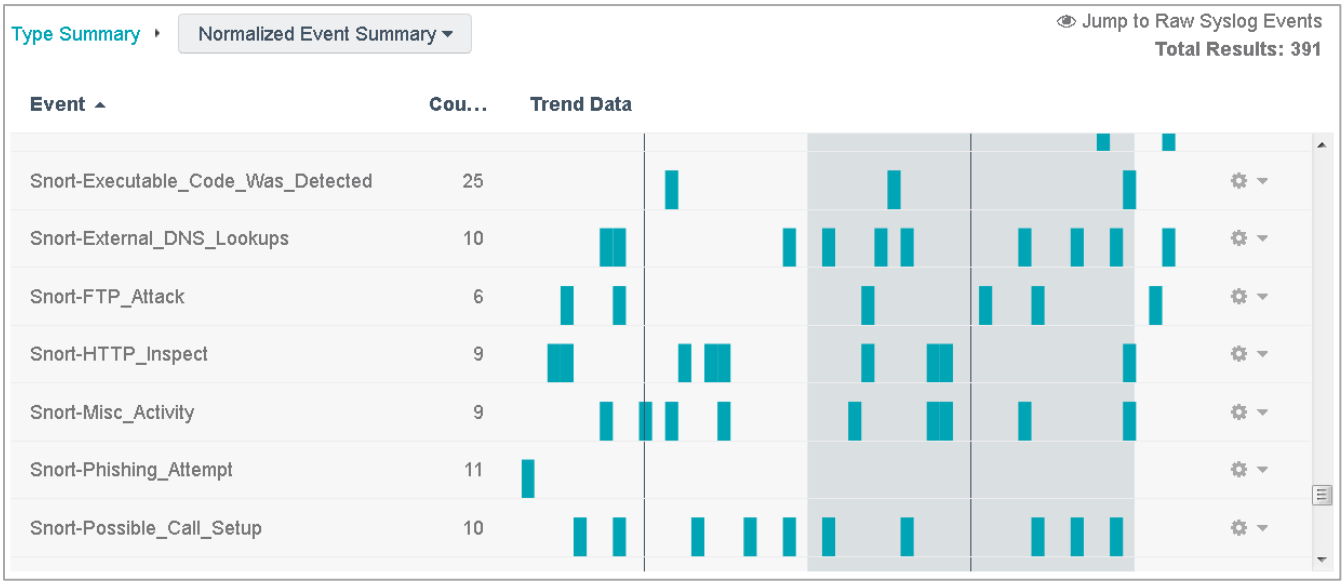
Event	Description
Bind-Potential_Attack	The BIND DNS server processed a DNS request that indicated a known attack type.
CiscoPIX-Potential_SNMP_Overflow_Attempt	A Cisco PIX firewall encountered an SNMP query that was likely a buffer overflow attack.
Fortinet-TCP_IDS_Event	The Fortinet firewall encountered a generic IDS event occurring over TCP.
IMAP-User_Overflow	The WU IMAP server encountered a very long user name that likely indicates a buffer overflow attempt.
Bro-SMTP_Event	The Bro IDS encountered an email-based attack.
Dragon-Compromise_Event	The Enterasys Dragon IDS encountered a compromise attack attempt.
Intrushield-Buffer_Overflow	The McAfee Intrushield IPS encountered a buffer overflow attempt.
Snort-TCP_Attempted_Information_Leak	The Snort IDS encountered an event classified as an Attempted Information Leak.

Below is a screen capture of normalized *intrusion* events:

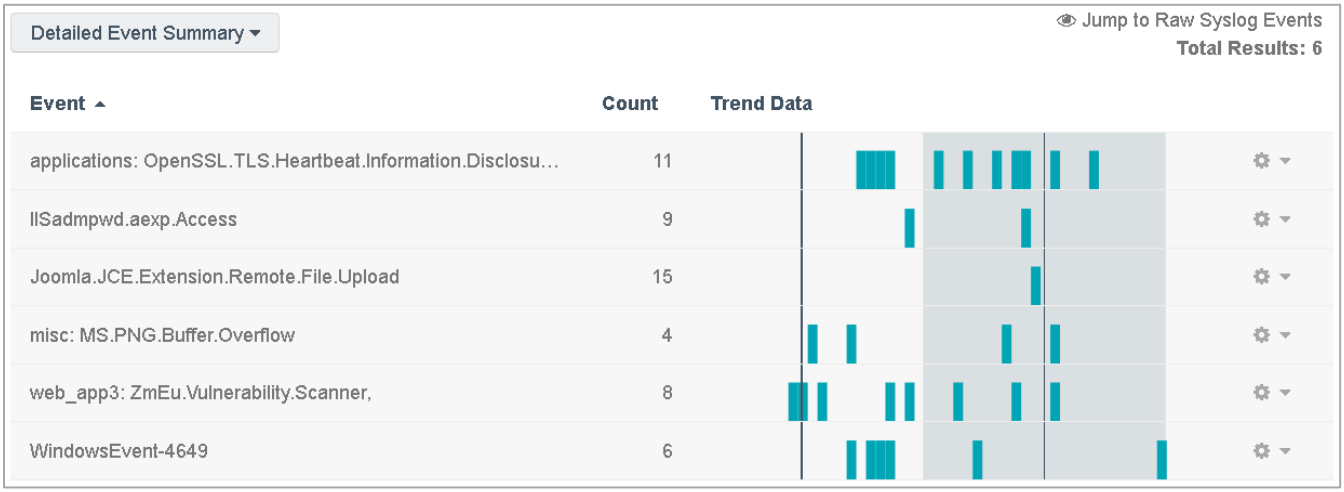


The Intrusion_Host_Scan and Intrusion_Network_Scan events indicate higher order correlated events from the LCE. They show that one IP address has done a large scan of a target and generated multiple unique IDS

events against it, or that one IP address has swept the network and generated a footprint of identical IDS events against local targets. Both correlations are covered in the Determined Scan and Attack Detection section.



The majority of these events are from Snort sensors. They have been normalized to the LCE's nomenclature of event names, which tries to honor as much of the log source technology as possible. In this case, Tenable has normalized Snort events to higher level names based on the Snort rule classifications. The LCE can also present intrusion events with their original event names designed by the various vendors. In this same dataset from above, the following screen capture was obtained by selecting the "Detailed Event Summary" tool:



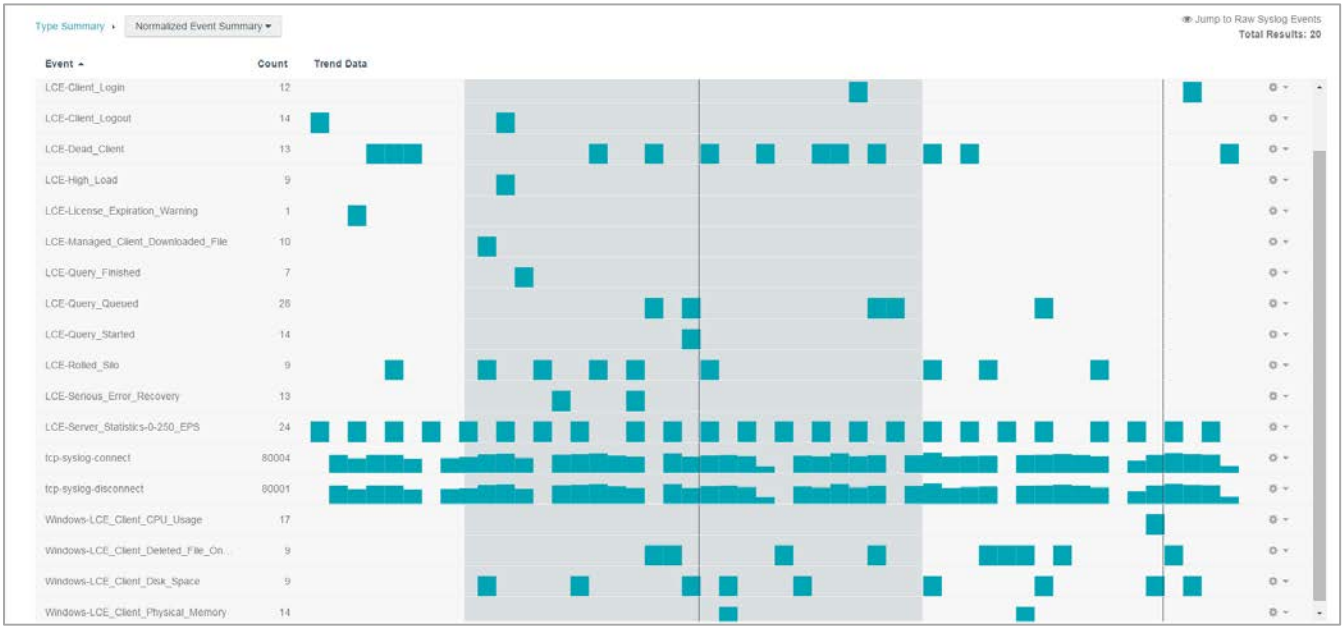
lce

This event type is used to help track information about LCE clients and their operating system statistics.

Examples of some event types:

Event	Description
LCE-Client_Login	An LCE Client has logged into the LCE.
LCE-High_Load	A system with an LCE Client running on it is under heavy CPU utilization.
Windows-LCE_Client_Disk_Space	Report of current disk space available on a Windows system.

Below is a screen capture from a network collecting logs from LCE Clients deployed across a couple hundred servers and desktops:



This screen capture displays logs of a variety of LCE Client connections and disconnections as well as status and information about the host systems (CPU, disk space and memory).

login

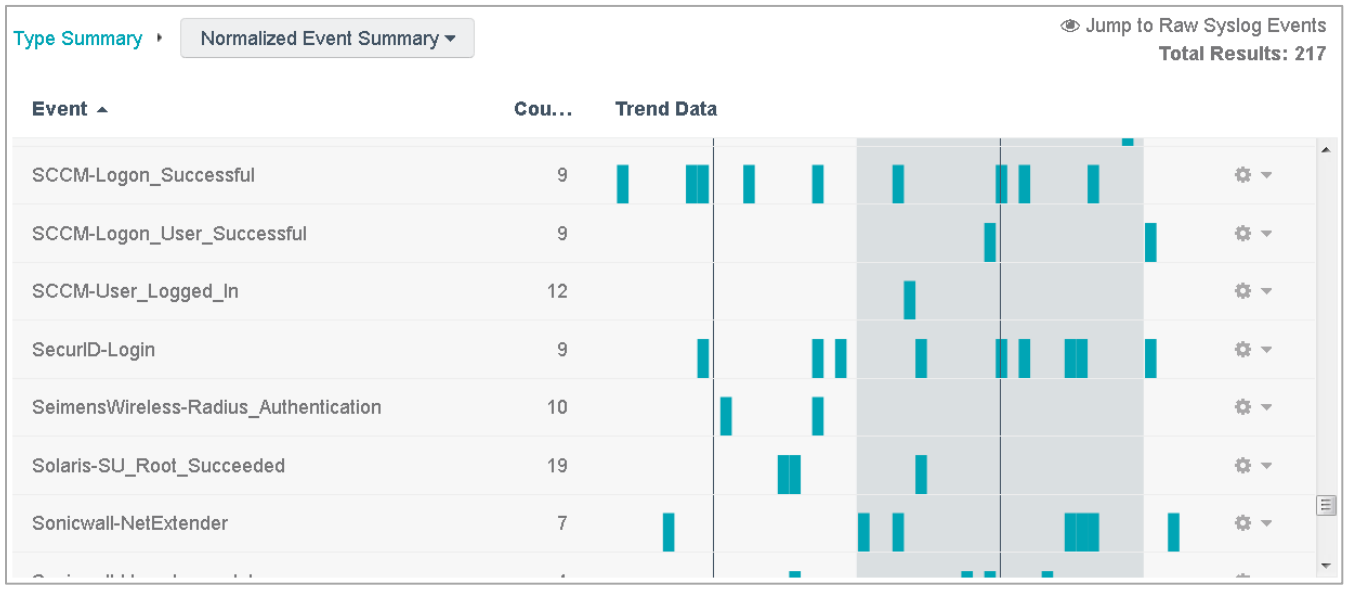
This event indicates any type of login event to an application, operating system, VPN, firewall or other type of device.

Valid logins are used to associate network users with IP addresses found in logs from firewalls, IDSes, NetFlow and other log sources. This is discussed in the User IP Address Correlation section. Authentication logs are also used to determine successful brute force password guesses, which is covered in the Successful and Unsuccessful Password Guessing section.

Examples of some event types:

Event	Description
CiscoASA-Admin_Permited_Console	A Cisco ASA had an administrator successfully authenticate via the physical console interface.
FTP-Valid_Login	An FTP server had a valid user authenticate.
IMAP-User_Login	An IMAP server had a user successfully authenticate to receive their email.
Unix-SU_Event	A Unix system had a user use the switch user “su” command.
Windows-Successful_Network_Login	A Windows system logged in a user from the network.

Below is a screen capture of a typical 24-hour period of a sample of *login* event types:



login-failure

Denotes any type of authentication log that indicates credentials were presented and were incorrect. This is distinct from application logs that block an IP address or access to resources that were denied. Those logs would normalize to event types of *firewall* or *access-denied* respectively.

Authentication logs are also used to detect brute force password guessing, which is covered in the Successful and Unsuccessful Password Guessing section.

Examples of some event types:

Event	Description
DLink-Admin_Login_Failure	A D-Link home router had a login failure for the administrator account.
Filezilla-Incorrect_Password	The FileZilla FTP server encountered a login failure for a user account.
Password_Guessing	The LCE has correlated multiple password login failure events.
Successful_Password_Guess	The LCE has observed multiple Password Guessing events followed by a successful login.
Cisco-NAC_Invalid_Login	A Cisco NAC appliance has had a user unable to authenticate.
Unix-Su_To_Root_Failed	A Unix system had a user account attempt to run the “su” command but was unable to provide the proper credentials.
Windows-Logon_Failure	A Windows system had a user unable to authenticate.

logout

The LCE normalizes events for applications, operating systems, VPN sessions and devices that detect when a user’s session is finished to the logout event type.

Examples of some event types:

Event	Description
CiscoASA-SSH_Disconnect	A Cisco ASA firewall had an established SSH session finish.
WatchGuard-VPN_User_Logged_Out	A VPN user of a Watchguard firewall has finished their session.
SC4-Logout	A SecurityCenter user has logged out of the application.

nbs

The LCE tracks all normalized events that have occurred for each host. As new normalized events are logged for the host, the LCE will generate secondary events based on the event type. This is a core type of correlation performed by the LCE.

For example, perhaps a Linux server is set up with SSH and has only ever had users login with passwords. The first time a user tried to login with an incorrect certificate, the resulting SSH-Failed_Publickey event would cause the LCE to generate a Never_Before_Seen-Login-Failure_Event. This concept is discussed in much greater detail in the First Time Seen Events section.

Examples of some event types:

Event	Description
Never_Before_Seen-Access_Denied	The host generated a normalized access-denied event log that had never been seen prior to this log.
Never_Before_Seen-Firewall	The host generated a normalized firewall event log that had never been seen prior to this log.
Never_Before_Seen-System	The host generated a normalized system event log that had never been seen prior to this log.

network

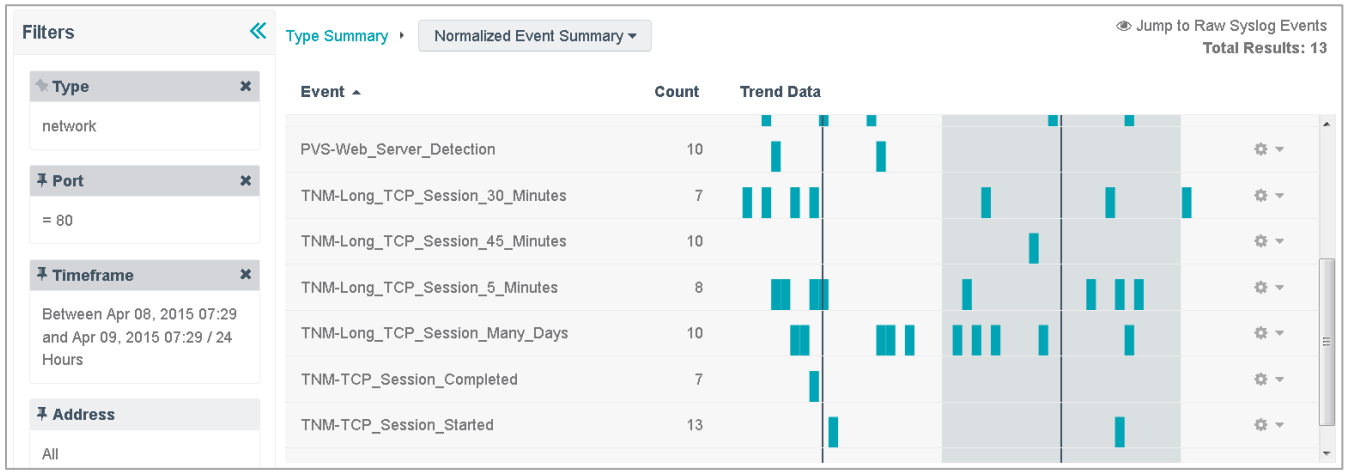
Observed application logs from the Passive Vulnerability Scanner as well as logs from the Tenable NetFlow Montior (TFM) and the Tenable Network Monitor (TNM) are logged to this LCE event type. Event names are used to designate the collection type (PVS, TNM or TFM) as well as session length and amount of bandwidth transferred.

Real-time logs from the Passive Vulnerability Scanner, Sourcefire's RNA, ArpWatch and some other sources that indicate network changes are also logged. The PVS will log application sessions based on protocols such as SSH, SSL, VNC, RDP and other applications.

Examples of some event types:

Event	Description
PVS-SMTP_Proxy	The PVS has observed a host proxy SMTP emails.
PVS-SSL_Session_Starting	The PVS has observed an SSL session.
Suspicious_Proxy	A host was generically observed to have multiple network connections that could indicate a proxy.
TFM-Long_TCP_Session_15_Minutes	A NetFlow session lasting about 15 minutes was observed.
TFM-Long_TCP_Session_Many_Hours	A Netflow session lasting several hours was observed.
TFM-TCP_Session_Whole_1-10MB	A sniffed session that transferred between 1 and 10 MB of data was observed.

Below is a screen capture of network events filtered for TCP port 80 over a 24 hour period:



Network sessions monitored by the LCE from the TNM or TFM are normalized for both length of session as well as size of data sent. This type of information can be used to detect large transfers of data where there should not be as well as long network sessions that could indicate compromise. Network logs are also used to detect ad-hoc proxy systems that may have been used to leapfrog into your network by determined hackers. This is discussed in the Suspicious Proxy Detection section.

process

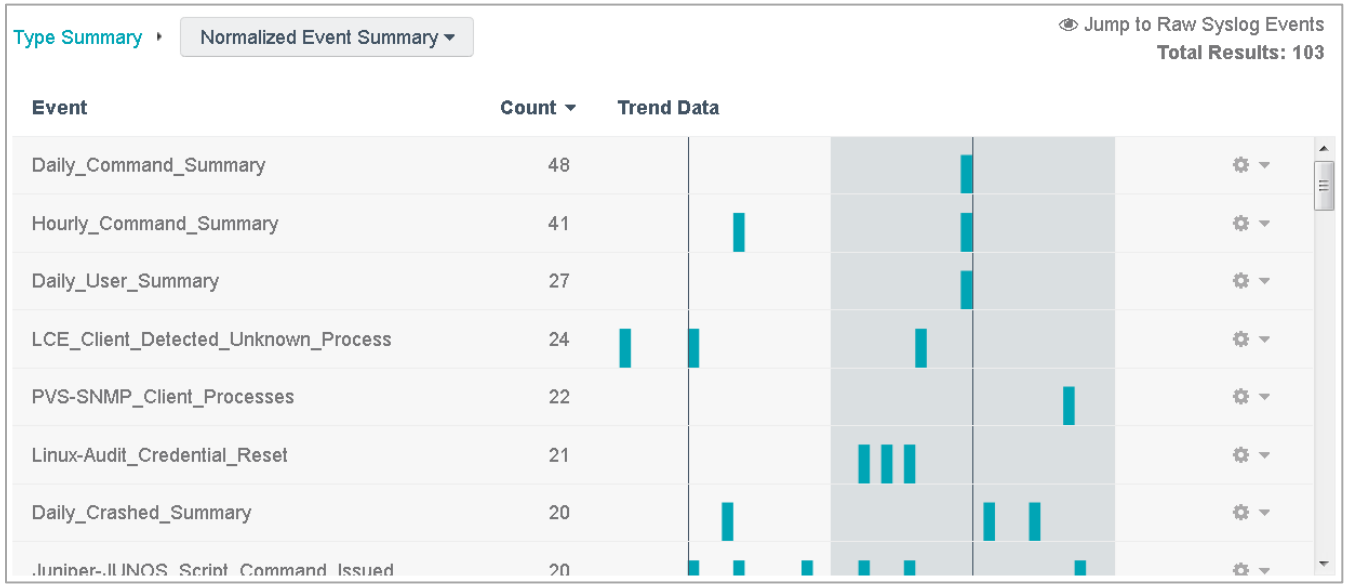
Logs from Unix process accounting, Unix audit logs and Windows event logs that indicate process starts and stops, as well as executable crashes, restarts, hung states and segmentation faults are logged to this LCE event type.

In addition, *process* logs are used to track crashes that occur network-wide. This is discussed in the Network Outage and Crash Detection section.

Examples of some event types:

Event	Description
Bind-Fatal_Exit	The DNS BIND application had a fatal crash.
FreeBSD-Root_Command_Issued	A FreeBSD system had a command issues by root.
Hourly_Hung_Summary	The LCE generated an hourly list of programs that have hung for a given system.
Windows-New_Process_Created	A Windows system logged the start of a new process.
New_Command	The LCE has detected a Unix or Windows command that was never previously run on a system.

Below is a screen capture of *process* events for a lightly used network over a 24 hour period:



In this case, there were summary events for a variety of daily and hourly command summaries – 48 and 41, respectively. There are also distinct events for a variety of crashed and unknown processes, as well as SNMP client processes and JunOS script commands.

restart

The LCE will normalize logs that show when applications, services, router, switches, devices and operating systems reboot, restart and are shutdown to the *restart* event type. Applications that have crashes, core dumps and other types of specific process related issues are logged to the *process* event type.

Some *restart* events are considered by the LCE to detect network wide reboots and system crashes that can indicate malicious software. This is covered in greater detail in the Network Outage and Crash Detection section.

Examples of some event types:

Event	Description
Unix-Syslog_Restarted	The Unix syslog process has restarted.
Windows-Unexpected_Shutdown	A Windows system has logged a shutdown that was unexpected.
MSSQLSVR-Pause_Request	An MS SQL server received a request to pause operation.
Nessus-Restarting	The Nessus vulnerability scanner is restarting.

scanning

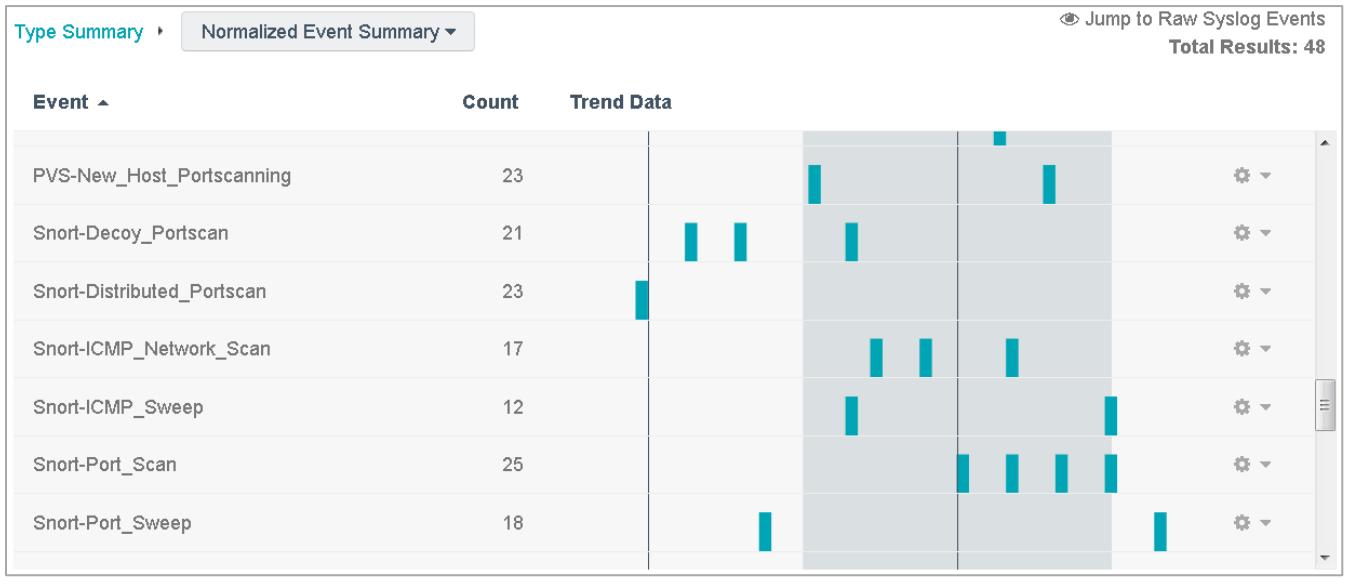
Network IDS, firewall, anti-virus and other log sources that detect port scans, port sweeps and probes are logged to the LCE *scanning* event type.

Port scanning events are used for a variety of correlations by the LCE including detecting worm infections, detecting port scans from botnets, and new hosts port scanning. Port scanning events are also processed by core LCE correlations engines to find never before seen, continuous and statistical event patterns. These are covered in detail in many other sections including the Worm Outbreaks the New Hosts Port Scanning sections.

Examples of some event types:

Event	Description
Bro-PortScan	The Bro IDS has discovered a port scan event.
NetScreenIDP-Port_Scan_UDP	The NetScreen IDP has discovered a UDP port scan.
SnortET-Scanning	A Snort Emerging Threats rule has alerted on a type of port scanning activity.
OSX-Limiting_RST_Response	A Mac OS X system encountered many network connections, which is typically the result of port scanning.
PVS-New_Host_Portscanning	The LCE has detected a brand new host on the network that immediately started to perform scans.

Below is a screen capture of detected port scans for a 24 hour period for an Internet services media company:



In this case, both Snort and the Passive Vulnerability Scanner were used to monitor network traffic. Snort had detected a variety of different types of port scans. Hosts that are added to the network and immediately start to perform scans may have been infected outside of the network (for example, connecting a laptop that became infected on a conference network to the office network) and are now trying to infect others locally.

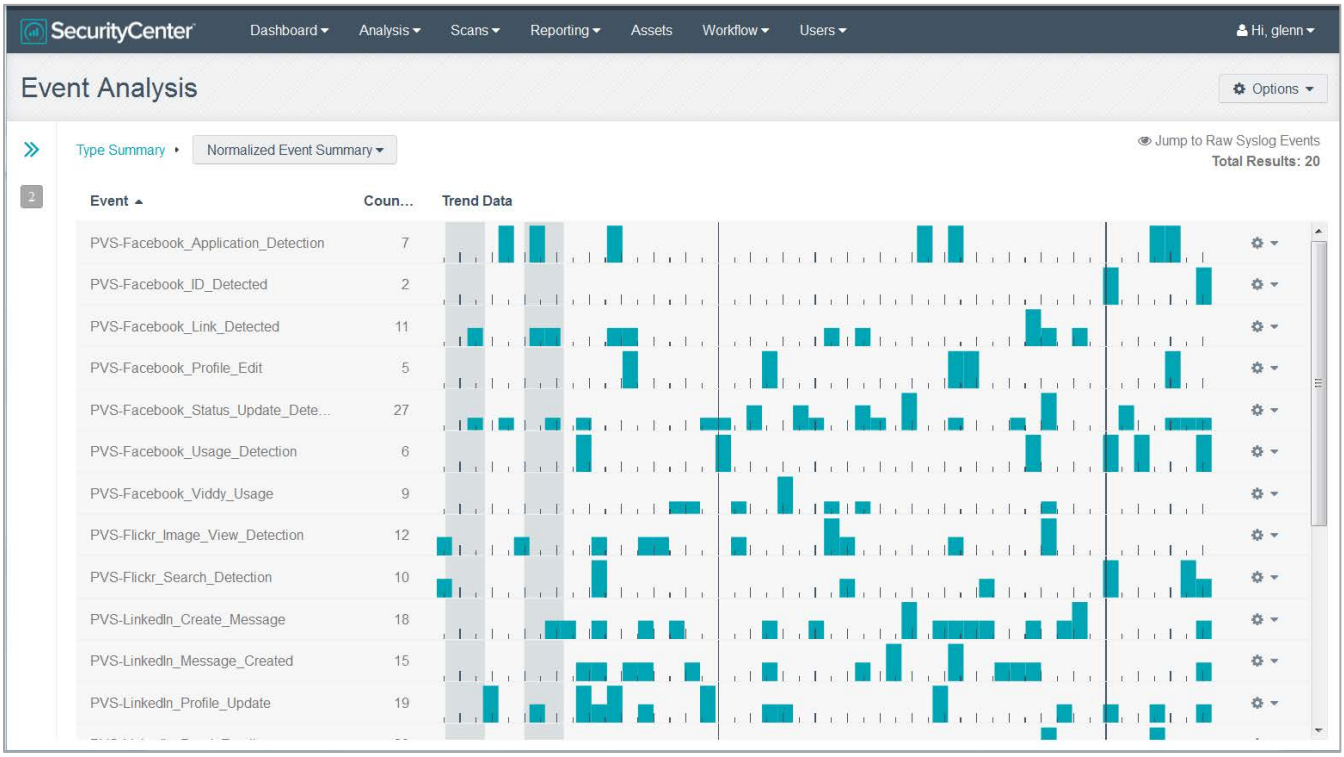
social_networks

The PVS detects a wide variety of social network activity such as Bing searches, logins to Gmail, Facebook, Wikipedia searches, Twitter and generic passively discovered IMAP and POP access. These are logged to the *social_networks* LCE event type.

Examples of some event types:

Event	Description
PVS-Web_Query_Yahoo_Search	PVS has logged a query to the Yahoo search engine.
PVS-FTP_UserID_Enumeration	PVS has logged an FTP access attempt and saved the account name.
PVS-MSN_Messenger_Login_Detection	PVS has observed a login to the Microsoft Messenger service and logged the username.
PVS-Facebook_Usage_Detection	PVS has observed a login to Facebook.

Selecting “social_networks” from the type summary page will show a list of *social_network* related events, as shown below:



spam

Logs from email servers, antivirus email tools, SPAM appliances, firewalls and other sources that indicate spam activity are normalized to the LCE *spam* event type.

Examples of some event types:

Event	Description
Postfix-SPF_Mail_Rejected	The Postfix email server rejected an email
StealthWatch-High_Volume_Email	The Lancope StealthWatch NBAD product detected a high volume of SMTP messages
Amavis-Blocked_Spam	The Amavis spam tool detected a spam message and dropped it

stats

For every unique type of event, the LCE will profile the frequency of events and alert when there is a statistical deviation for any event on a given system. Statistical anomaly detection is a core form of correlation for the LCE and is covered in depth in the Statistical Anomalies section. These events are normalized to the *stats* (statistics) LCE event type.

For example if the LCE detected a large spike in the frequency of the PVS-Web_Query_Yahoo_Search events from a system it was monitoring, it would issue a Statistics-Web_Access_Large_Anomaly event.

Examples of some event types:

Event	Description
Statistics-USB_Large_Anomaly	The LCE detected a large anomaly in USB (event type *usb*) insert or removal events
Statistics-Web_Access_Medium_Anomaly	The LCE detected a medium anomaly in the amount of *web-access* event types for a given host

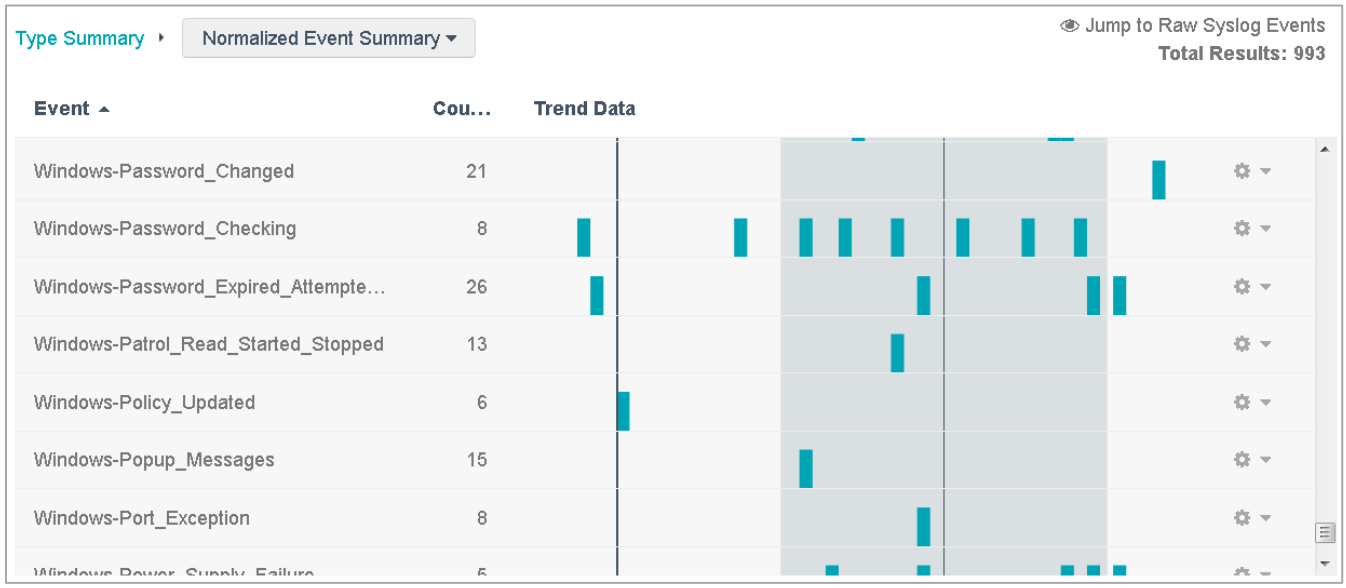
system

The LCE will normalize operating system, router, switch or device logs of significance to the event type of the system. Login failures, errors and application events are logged to other event types.

Examples of some event types:

Event	Description
Fortinet-Firewall_Added_Local_User	A new user was added to a Fortinet firewall
Promiscuous_Mode_Enabled	Sniffing was enabled on a Unix system
Linux-Group_Removed	A Linux system had a group removed
Windows-Directory_Service_Created	A directory service was created on a Windows server
Windows-Registry_Changed	A registry value was changed on a Windows system

Below is a partial screen capture of *system* events:



Some of the events in the *system* category are mapped to the LCE's *detect-change* correlation engine. Many are simply status messages that indicate significant operation of the operating system, application, router, server, etc.

threatlist

The LCE maintains a list of hostile IPv4 addresses and domains that are known to be participating in botnets. This is a core type of correlation performed by the LCE and is covered in-depth in the IP & DNS Reputation section.

The LCE considers connection, DNS lookup, web access, web error, intrusion and network events to detect when a hostile IP address connects inbound to your network as well as when a host on your network connect outbound.

Examples of some event types:

Event	Description
Inbound_Threatlist_Connection	A known hostile host has connected to a system on your network
Outbound_HTTPS_Threatlist_Connection	A local host has connected on port 443 to a known hostile IP address
Outbound_High_Port_Threatlist_Connection	A local host has connected on a port larger than 1024 to a known hostile IP address

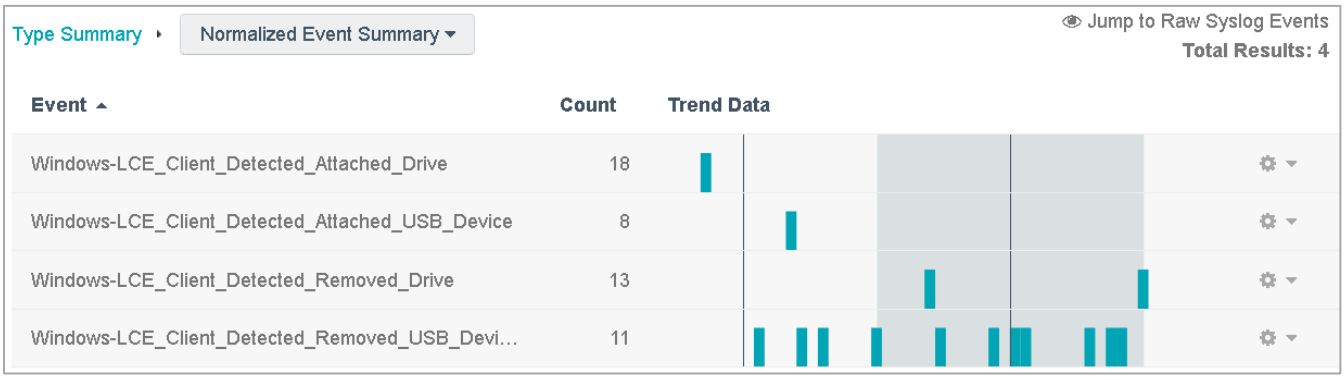
usb

The LCE Windows Client can detect USB and CD-ROM insertions and removals. The logs generated by these events are normalized to the *usb* event type.

Examples of some event types:

Event	Description
Windows-LCE_Client_Detected_Attached_Drive	A USB or CD-ROM drive was attached
Windows-LCE_Client_Detected_Removed_Drive	A USB or CD-ROM drive was removed

Below is a screen capture of all USB device connections and disconnections for a period of 24 hours:



virus

Logs that indicate the presence of a virus in email, a virus found on a system by an anti-virus agent and virus logs found by network IDS events and firewalls are normalized to the LCE event type of *virus*.

Information about the status of an anti-virus agent, such as the agent reporting a successful update of anti-virus signatures, is sent to the event type of *application*. Similarly, activity from a virus infection can be normalized to *intrusion* events, *threatlist* connections and *scanning* events.

Examples of some event types:

Event	Description
Symantec-Virus_Warning	A Symantec anti-virus agent found a virus
Sophos-Email_Quarantined	A Sophos email processing component found a virus present in an email and quarantined it
McAfee-Warn_Mode_Would_Be_Blocked	A McAfee anti-virus agent found a virus and could not block activity associated with it but the agent is in warning mode
SnortET-Malware_Activity	A Snort IDS sensor running the Emerging Threats ruleset has detected malware activity

The LCE allows virus detection from system, network and email sources of anti-virus protection and detection to be centralized into one event type.

vulnerability

As security issues and new information about systems and networks are reported as part of the vulnerability monitoring process, the LCE normalizes these event types to the *vulnerability* event type.

Examples of some event types:

Event	Description
Intrushield-Unwanted_Software	A McAfee IntruShield IPS has found software that is likely unwanted
RNA-OS_Confidence_Update	A SourceFire RNA sensor has observed enough traffic to update its guess for the operating system of a given node
PVS-High_Vulnerability	A high severity vulnerability was passively discovered on a given node

web-access

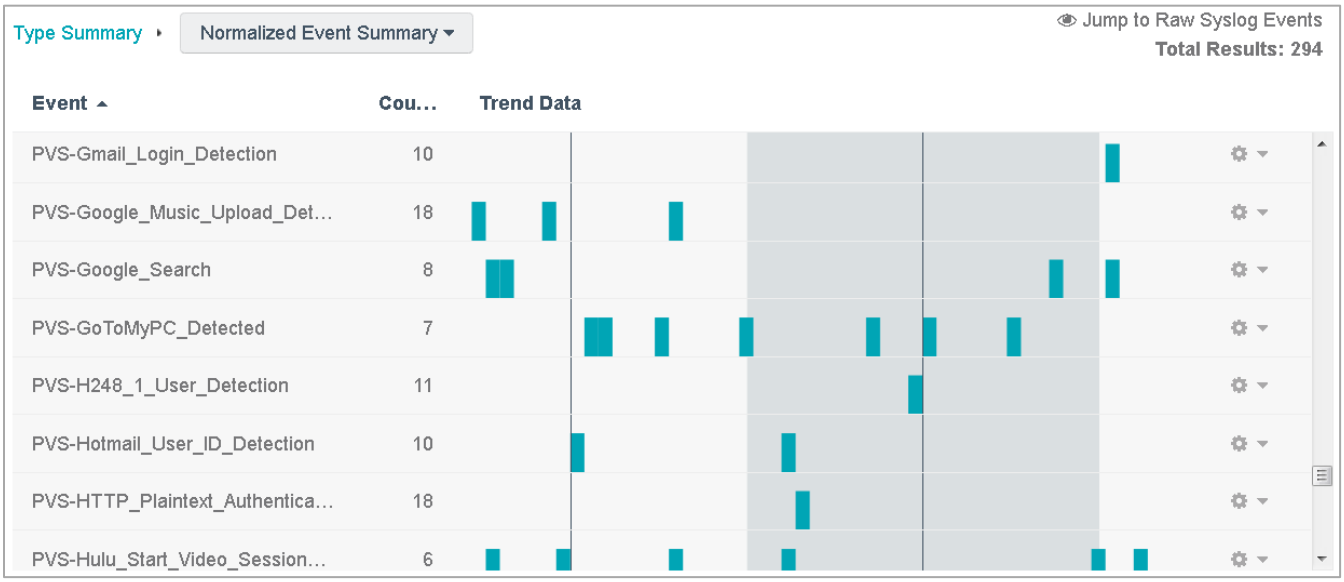
Any type of log that indicates a successful connection to a web resource is normalized as a *web-access* LCE event type.

Logs gathered by web servers, web proxies, firewalls and load balancers that indicate connections to web services are logged here. Note that *web-access* events can refer to a host on the Internet connecting to a public web server or internal users accessing the Internet through a web proxy.

Examples of some event types:

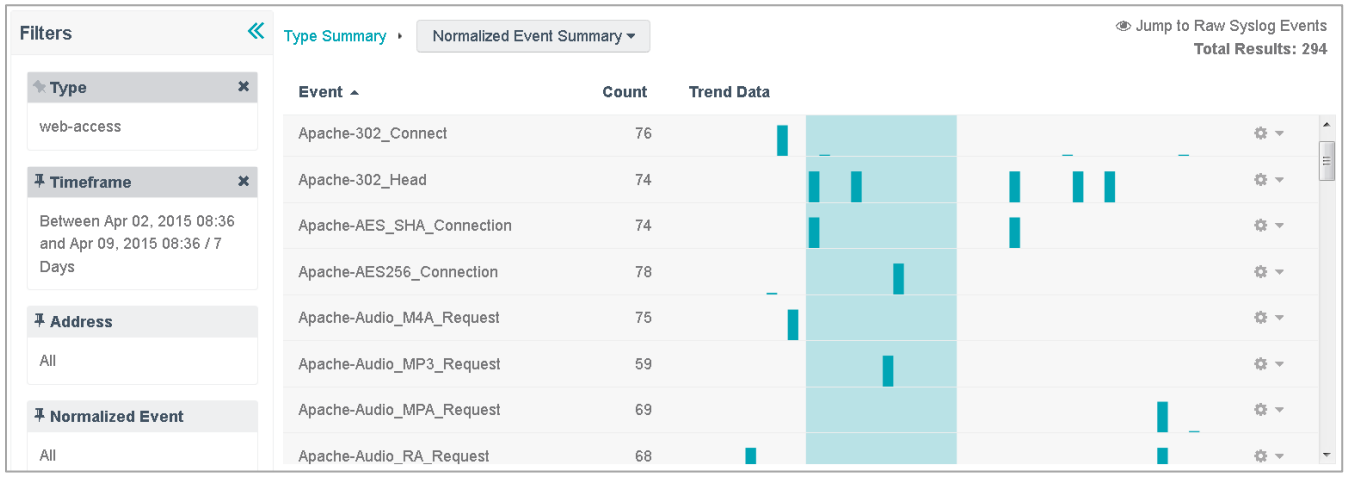
Event	Description
Apache-Valid_Web_GET_Request	An Apache web server logged a valid GET request to a resource hosted by it
CiscoASA-Accessed_URL	A Cisco ASA firewall logged the specific URL on a connection to a web site
IIS-Move_Request	An IIS web server encountered a request to move a resource
PVS-Web_Query_Google_Search	The PVS observed a user performing a Google search
PVS-NetFlix_User_Detected	The PVS observed a user log in to a Netflix account
Squid-TCP_Miss	A Squid web proxy encountered a valid request for a website, but the website was not present in the existing cache

Below is a partial screen capture of web-access events for a 24 hour period from a network monitored by the PVS:



These were all obtained by network analysis via the PVS and sent to the LCE which, in turn, normalized the logs with the names seen in the screen capture above.

Below is another example with logs from Apache web servers for seven days:



web-error

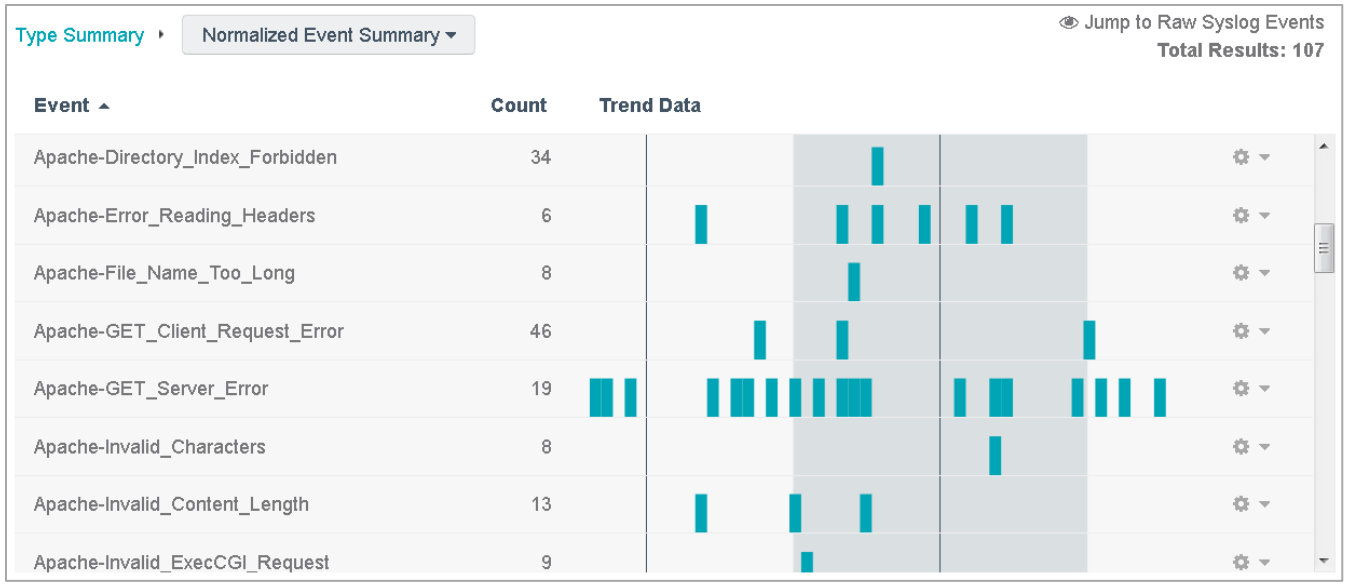
This event type denotes any type of web access event that is denied because the file does not exist, the server responded with an error or a firewall or web application firewall blocked the access. These logs are generated every day by users who reference incorrect URLs, but are also generated during web application probes.

The LCE processes error logs from web servers to look for attacks from IP addresses that are known to be botnets as well as low-and-slow web application attacks. The detection of low-and-slow and determined web application scans is covered in the Determined Scan and Attack Detection section.

Examples of some event types:

Event	Description
Apache-Invalid_Method	An Apache web server encountered a request that referenced an invalid HTTP method
IIS-Bad_Get	An IIS web server logged a GET request that returned an error
Web_GET_Forbidden	A generic HTTP GET request was deemed forbidden and logged as an error
Squid-Proxy_Denied	The Squid web proxy log denied a web proxy request
Web_POST_ServerError	A generic attempt to perform an HTTP POST to a web server encountered an error
Web_Error_From_Threatlist_Address	A web server logged a web request that returned an error code and the source of the request was an IP address that is known to be a botnet

Below is an example screen capture of *web-error* event types for a seven day period from a cluster of Apache web servers:



IV. Core Event Correlation

First Time Seen Events

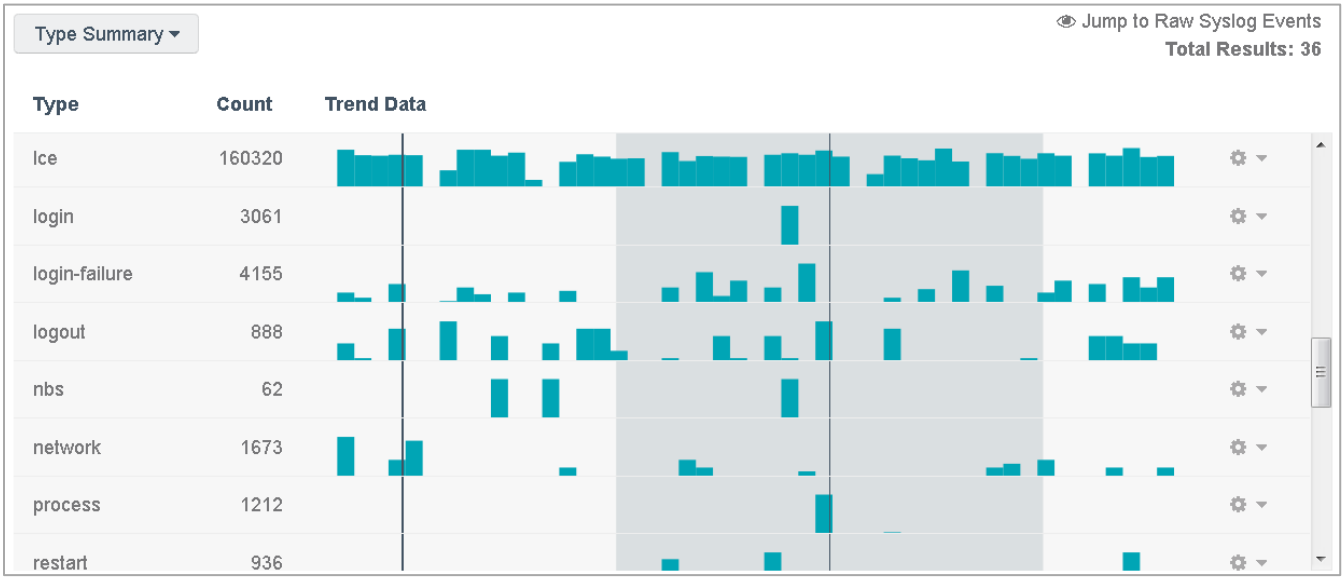
What is it?	Identifies logs that are new, unique and have never been seen previously on a particular host or associated with that host.
What does it do?	As logs are normalized, the event and IP address are associated with the log and, if that particular type of event has never been seen previously for that host, a new alert is logged.
Why does it matter?	The presence of new types of IDS events, proxy logs, login failures, errors and many other types of sources can indicate attackers, compromised systems, new systems, major application issues and changes.
LCE Event Type	All correlated events are assigned to the *nbs* event type, which stands for "Never Before Seen".
Statistical Events	If a host encounters a new type of statistical anomaly, the LCE will create a new event with the name of Never_Before_Seen-Statistical_Event. Similarly, if there is a statistical spike in *nbs* events for a given host, events Statistics-NeverBeforeSeen_Minor_Anomaly, Statistics-NeverBeforeSeen_Anomaly, Statistics-NeverBeforeSeen_Medium_Anomaly and Statistics-NeverBeforeSeen_Large_Anomaly will be generated depending on the size of the anomaly.
First Time Seen Events	First time seen events are generated for every type of event *except* those from the never before seen correlation engine.
Continuous Events	The LCE does not process *nbs* for continuous activity.

Administrators who view server, application and security logs are usually sensitive to deviations in log data that conveys information not previously known. Such data could indicate a new type of status, security state or error that is of interest.

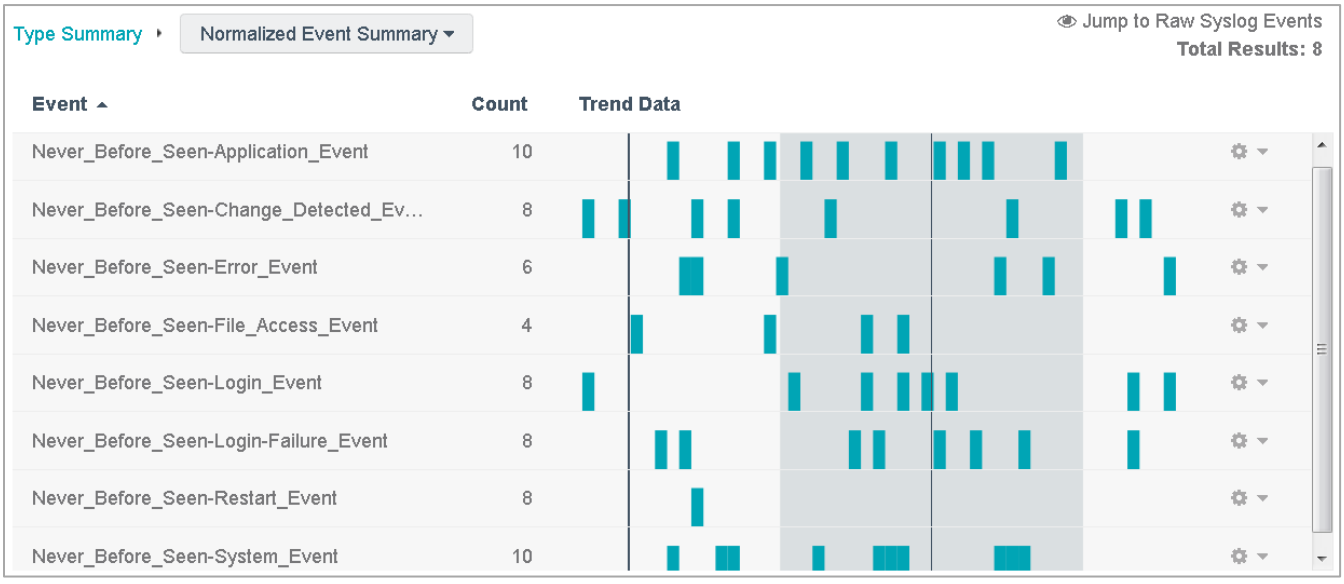
The LCE performs “Never Before Seen” analysis on every normalized log for each host on the local network. Normalized events are placed in the *nbs* event type category.

The relationship between log and host is based on the normalized IP address. This means the LCE will identify new events for a host regardless if the log came from the host or not. For example, a new type of error from a Windows 2008 domain controller could be recognized as occurring on 172.20.20.56. If that same host had a new type of intrusion detection event detected by a TippingPoint sensor, the LCE would also report this as a “new” event from 172.20.20.56.

Below is a screen capture of a type summary for a small lab network:



There were 62 *nbs* events. Drilling into them produced the following screen:



There were several statistical events that had never occurred previously. Displaying the logs from some these events showed the following:

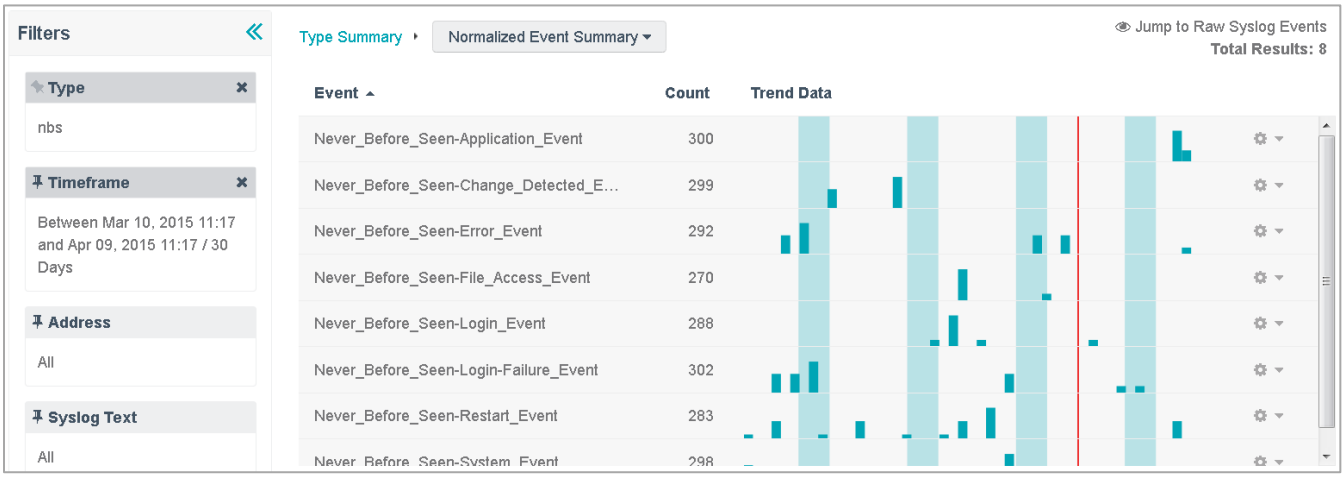
```
Never_Before_Seen-stats event, source IP address 192.168.1.55 (Work22511.lab)
      has never seen event Statistics-DNS_Minor_Anomaly in the past. This event
      was towards host 192.168.1.55. If this event is unusual for this IP
      address, please investigate this event further.
Never_Before_Seen-stats event, source IP address 192.168.1.6 (lab23-
      computer.lab) has never seen event Statistics-Application_Minor_Anomaly
      in the past. This event was towards host 192.168.1.6. If this event is
      unusual for this IP address, please investigate this event further.
Never_Before_Seen-stats event, source IP address 192.168.1.61 (Work24511.lab)
      has never seen event Statistics-PVS-Network_Minor_Anomaly in the past.
      This event was towards host 192.168.1.61. If this event is unusual for
      this IP address, please investigate this event further.
```

Taking the first log, we have a statistical DNS minor anomaly that occurred on 192.168.1.55 for the first time. If there were one hundred more logs of this type the next day (or the next minute), we'd never see another *nbs* alert for this event.

Never before seen event names are based on the type of the underlying event. If a given event occurring for the first time is from the USB category, the alert generated by the LCE will be Never_Before_Seen-USB_Event.

There are several different ways to leverage *nbs* event logs.

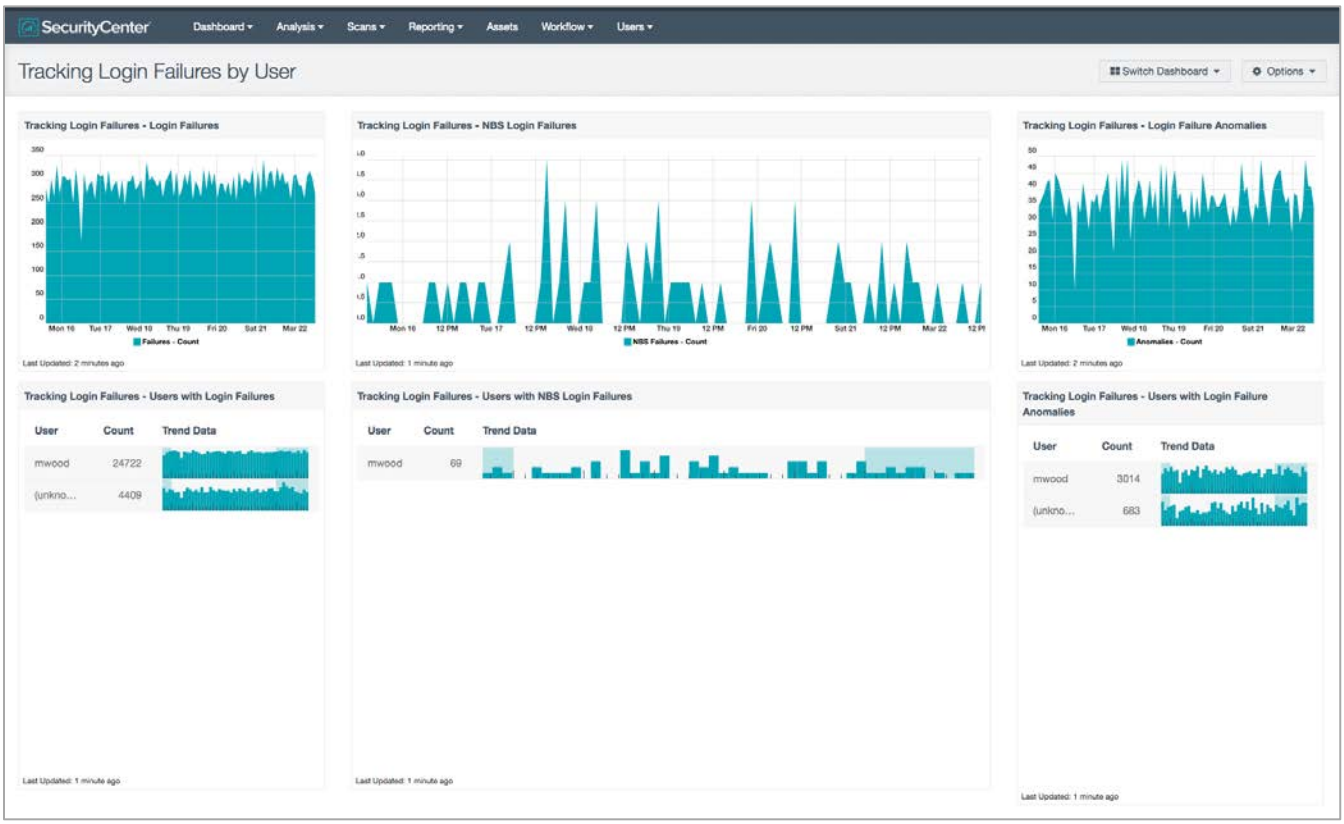
Strategically, attempting to trend all *nbs* events for large networks is an excellent form of event log reduction. Instead of attempting to understand all logs occurring on the network, identifying new types of logs that have occurred provides insight to changes in function and behavior. For example, consider the following screen capture of all *nbs* events for the past thirty days in a small lab:



Near the midpoint of this time frame, there is a spike of certain events occurring at the same time. Such a spike could indicate rapid change, such as the addition of a new host generating logs as well as a major type of attack or compromise.

Never_Before_Seen-Intrusion_Event logs may also be seen under the *nbs* event type. As a security practitioner, it is worth investigating if you had IDS events on your network over a long time that you had not seen before. It shows that attacks or normal network browsing is causing your IDS technology to log security events that have not been logged previously.

Events from the *nbs* type can also be combined with other event types, both discretely or generically in dashboards. For example, in the screen capture below a dashboard is used that displays login failure information per user, including login failure events that have never been seen before:



To learn more about this dashboard, please read the associated blog entry on Tenable's dashboard web site.

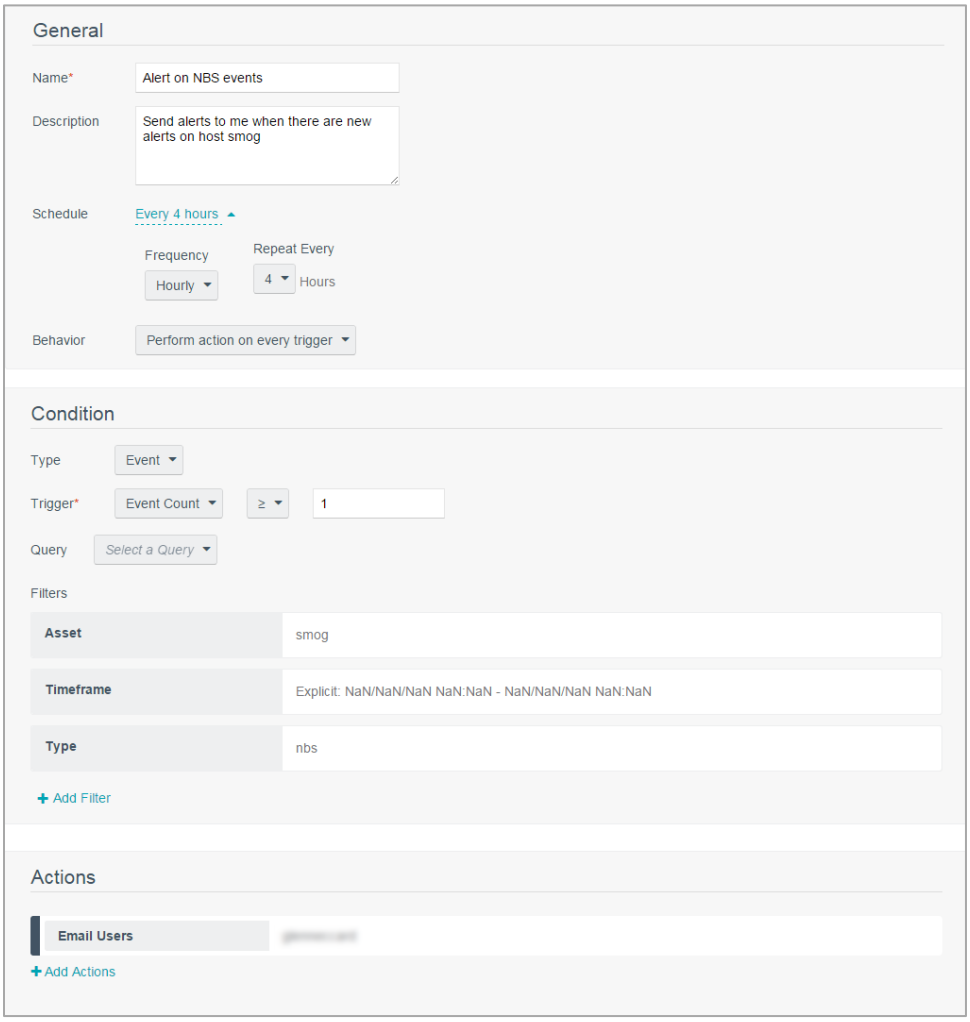
For reporting, *nbs* events can be used as a source of iteration. The use case would be to quickly create a report that listed the discrete events for each host or asset that had a never before seen event. Report iteration can be used to do this.

An example use case would be to have a daily report created that identified each host that had an *nbs* event associated with it, and then for each of those hosts, display first one hundred *nbs* event logs and then an event summary to provide context.

Below is an example screen capture of what a custom report template that performed this type of reporting would look like:

Reports like this could be further customized to only report *nbs* events for certain assets, to only iterate on certain *nbs* event types or combinations of both. For example, new errors detected on laptops might not be as interesting as new errors obtained from core network switches, routers or DNS servers.

Finally, *nbs* events can be used to create useful alerts. Below is an example alert that has been configured to fire if an event count of *nbs* events for an asset named "smog" is one or higher for the past four hours.



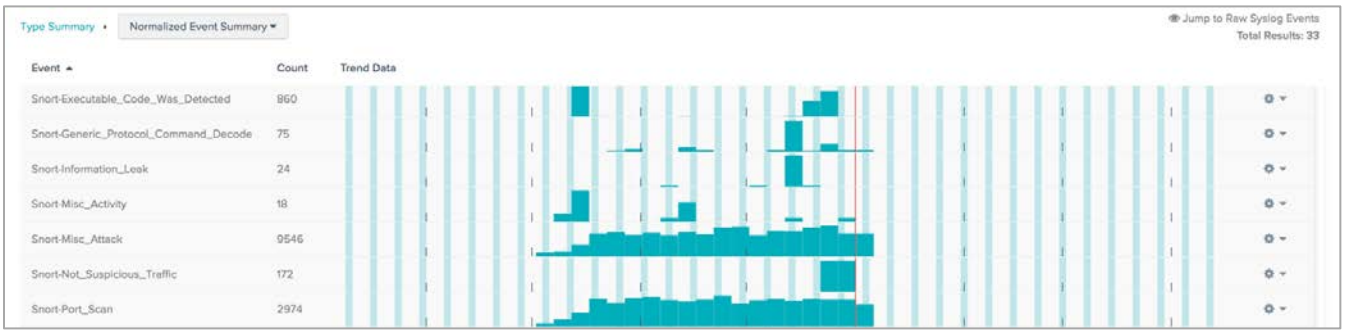
The alert sends me an email. For alerts, it's very useful to consider combinations of lengths of time for both the LCE query (the past four hours in this case) and the alert frequency (also four hours in this case). For this alert, I want to get an email once every four hours if there are *nbs* events on "smog". Since the behavior of the alert is set to "Perform Action on Every Trigger", I'll get an email every four hours as long as there are new *nbs* events on smog. Sometimes, advanced LCE users decouple the query time length with the frequency of the alert checking and also change the behavior to "Perform Actions Only on First Trigger". This provides a higher frequency of checking across longer periods of time. For example, perhaps an LCE user wants to know if there have been more than one hundred *nbs* events in the past ten hours, but they want this checked every two hours. If the behavior was not set to perform actions on just the first hit, an LCE may get multiple emails from this alert.

In conclusion, the *nbs* correlation of the LCE identifies classes of logs that occur on systems that have not been seen before. These new logs can be leveraged for forensics analysis, alerting, trending and understanding the change on your network.

Continuous Activity Detection

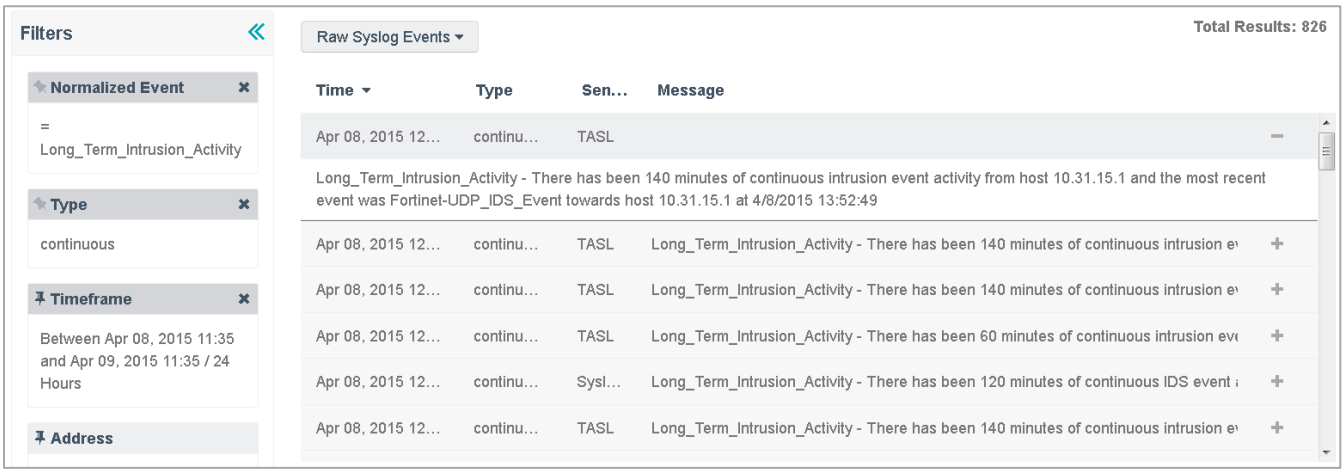
What is it?	Identifies ongoing activities and events when previously there were none.
What does it do?	Tracks *intrusion*, *error*, *threatlist* and many other types of events to look for determined attackers, brute force password guessing, spam activity, virus outbreaks and much more.
Why does it matter?	It adds significance to events that would otherwise not be identified by the LCE's first time seen or statistical correlation engines. Most event patterns are "bursty" and do not occur continuously. Those that occur repeatedly might occur at a low event rate and would otherwise go unnoticed.
LCE Event Type	All correlated events are assigned to the *continuous* event type.
Statistical Events	If a host generates enough *continuous* events to warrant a statistical deviation, the LCE will create a Statistics-Continuous_Minor_Anomaly, Statistics-Continuous_Anomaly, Statistics-Continuous_Medium_Anomaly and Statistics-Continuous_Large_Anomaly event depending on the size of the anomaly.
First Time Seen Events	For any *continuous* event occurring on a host for the very first time, the LCE will generate a Never_Before_Seen-Continuous_Event log.
Continuous Events	The LCE does not process *continuous* events for further consideration of *continuous* events.

Detecting events that were not occurring and then begin to occur without stopping for long periods of time is a core form of correlation performed by the LCE. It attempts to replicate detection of step functions such as this event trace below:



The LCE assigns events of the continuous category names that start with “Long_Term”. In this case, the event was continuous occurrences of an *intrusion* event type. The activity had been occurring for roughly forty minutes. The host in this case most recently generated a Snort-Generic_Protocol_Command_Decode event towards one of our IPs.

The LCE’s engine will issue updates to the continuous activity every 20 minutes if the activity is still ongoing. Below is an example screen capture of various types of continuous events taken from a live network:



Many of the events reported at one point are updated roughly twenty minutes later with a new log stating that the period of activity has been active for the new time.

The LCE will look for continuous events for specific forms of events, entire classes of event types and also discrete lists of pre-programmed events. Each type of *continuous* activity event will be briefly explained.

Long_Term_DNS_Failures

Most users type incorrect website names and email addresses every day. However, we likely don’t do this continuously and for long periods of times. Hosts that perform DNS queries that fail for long periods of time typically are:

- Attempting to send emails to domains that do not exist
- Systems with misconfigured DNS information
- Systems performing vulnerability scanning of the network that attempt to resolve IP addresses not in DNS
- Infected systems with spyware or botnet programs performing Internet scanning or attempting to connect to predefined DNS names that are botnets that do not exist
- Systems that attempt to resolve IP addresses from visitors such as email and web sites

The LCE can process DNS query logs from servers running BIND and it can also see them based on traffic analysis from the PVS, which logs both successful and unsuccessful lookups.

To leverage this correlation effectively, alerts, reports and dashboard can be applied to asset lists that do not normally create these logs. You may not know that your Exchange server fails to resolve thousands of DNS names every day if not more through “normal” email operations. However, your laptop computers, routers and many other devices likely do not perform DNS lookups incorrectly unless there is an issue.

Long_Term_DOS_Activity

A type of attack normalized by the LCE is the *dos* (Denial of Service) type. The majority of these logs are from network IDS devices such as Snort or TippingPoint. An example sanitized log of this type would be:

```
Long_Term_DOS_Activity - There has been 80 minutes of continuous DOS activity
      from host 382.316.332.199 (a-sanitized-customer-
      from.cable.virginmedia.com) and the most recent event was Snort-
      Denial_Of_Service towards host 172.25.210.95
      (an.internet.facing.web.server.com) at 1/7/2012 20:56:51
```

Denial of service attacks are generally classified into instant, degrading or bandwidth. An instant attack will cause a system to reboot, go to 100% CPU utilization, crash or otherwise instantly become unavailable. A degrading attack reduces the amount of capacity a system may be able to handle and finally, a bandwidth attack will simply clog the network connections to a target.

The ability to trigger on *continuous* denial of service events is useful for several reasons:

- For degrading attacks, an attacker may have to resend attacks multiple times.
- For attacks against entire networks (such as sending the infamous “ping of death”) it may take time for a remote attacker to sweep the targets.
- The directionality of the correlation is inbound and outbound – so it will see if you have a system on your network attempting to launch denial of service attacks against various targets.

Long_Term_Error_Activity

Systems generate errors through normal courses of action. However, when a system generates error messages over and over in vain trying to tell a user that something is wrong, these messages can go unnoticed if they are buried in a sea of other status and logging messages. The LCE attempts to solve the identification of this by applying the continuous event filter to all normalized *error* messages.

Errors can result from incorrect configurations, hardware failures, local and network attacks, incorrect product usage, and actual bugs in code. Identifying when a system log has been created for an error that repeats again and again is useful to identify major errors.

Long_Term_HighCPU_Activity

This activity looks for hosts that have reported a CPU spike continuously. CPU utilization information is returned by both LCE Windows and LCE Unix clients. The LCE correlation specifically subscribes to the LCE-High_Load and LCE-High_CPU_Usage events.

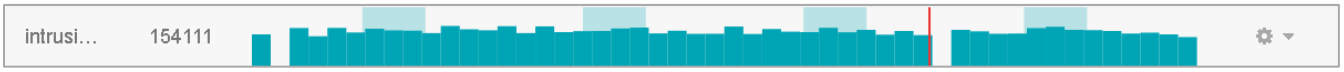
A system with long term CPU usage may be experiencing several scenarios including:

- Underpowered hardware
- Under a high external load
- Under a denial of service attack
- Operating system and application errors
- Some types of viruses and malware

Some LCE customers have leveraged the iterator report to create lists of systems that have had spiked CPUs, and then created individual chapters for each that list all *system*, *error* and *process* event types for the past twenty four hours.

Long_Term_Intrusion_Activity

To detect long term scanning, probing and attacks, long term *intrusion* events of this type can be leveraged for alerting, reporting and dashboards. As a form of event reduction, consider the following type summary of *intrusion* events:



In comparison, for that same time period, there were only 25683 continuous events of IDS activity:

For this particular monitored network, most of the intrusion events were Snort-Information_Leak. Below is an example of such an event for a single IP address that the LCE had created a Long_Term_Intrusion_Activity event:

In environments where there is a high false positive rate, a high attack rate or simply a lot of activity, detecting events that are continuous can help prioritize large attacks.

When creating dashboards, iteration reports and alerts, keep in mind that the LCE considers the source IP address of the *intrusion* events. This means that the directionality filters within the LCE can help separate inbound versus outbound correlated events.

Long_Term_Network_Scanning

The LCE considers port scanning events from the *scanning* event type category for long term scanning activity. The intent is to find internal and external IP addresses that are performing many different types of port scanning events in a determined effort to find targets.

Sources of port scan alerts include:

- Legitimate and unauthorized vulnerability scans
- Legitimate and unauthorized port scanning
- Network probes from network management tools (i.e., sweeping for SNMP stacks)
- Worm propagation probes
- Incorrect application analysis such as Skype and BitTorrent

Here is an example sanitized log of such a correlation:

```
Long_Term_Network_Scanning - There has been 40 minutes of continuous scanning
      activity from host 358.310.398.326 (someplace.optusnet.com.au) and the
      most recent event was Snort-Decoy_Portscan towards host 128.133.323.318
      (someotherplace.here.org) at 12/25/2011 03:50:16
```

As with the *intrusion* events, the LCE will consider the source IP address of the port scan events when looking for continuous activity. This means that directional filtering with either inbound or outbound directionality filters or source asset filtering could be used to create alerts, dashboards or reports.

Long_Term_RDP_Client_Activity

The PVS produces real-time logs of many different types of network sessions including a log of the start of a Windows Remote Desktop session. Below is an example sanitized log:

```
<36>Jan 06 19:15:50 pvs:
      349.834.210.110:3389|439.348.939.338:39044|6|5935|Windows RDP / Terminal
      Services
      Detection|{03}{00}{00}{0b}{06}{d0}{00}{00}{12}4{00}|{03}{00}{00}#{1e}{e0}
      {00}{00}{00}{00}{00}Cookie:{20}mstshash=admin{0d}{0a}|NONE
```

By looking for continuous RDP sessions from one host, the LCE can find evidence of brute force password guess between two hosts, as well as sweeps when one host attempts to connect via RDP to multiple targets.

If you aren't in a position to collect logs from Windows systems, the ability to passively monitor network traffic with the PVS to create these logs is very useful since without them, you would have no local knowledge of system, application and security events occurring. Even if you were collecting system logs from each Windows system, the event log does not log outbound requests for RDP.

For networks that expose port 445 to the Internet, it is quite common to see event patterns such as this:

For the past 30 days, this network has had 571 RDP sessions. If we performed a manual analysis on this, we'd see that most were inbound from many different parts of the world. Even worse, many of the systems that were targeted were eventually connecting out to other Windows computers.

To help make sense of this, the LCE's correlation engine has the PVS-Windows_RDP_Session_Started event added to its continuous event engine. For this particular network, the same period of time produced 174 *continuous* events as shown below:

The LCE considers the source IP address of the PVS-Windows_RDP_Session_Started event when performing this type of correlation. Directionality and source event filtering can be used to look for internal versus external long term activity.

Long_Term_Social_Network_Activity

The LCE can use *continuous* event correlation to identify when network users engage in Twitter, Facebook and other types of social media for a long time. When combining these events with the user ID to IP address tracking (see the User IP Address Correlation section) users who have been performing social networking for a long time can be identified.

The majority of the events that indicate social network activity come from the PVS's ability to analyze network web traffic to classify YouTube, Facebook and other forms of social networking sites. Below is an example sanitized log from a network that has detected a system visiting the LinkedIn website:

```
Long_Term_Social-Network_Activity - There has been 40 minutes of continuous
      social network activity from host 216.52.242.80 (linkedin-ela4.com) and
      the most recent event was PVS-LinkedIn_Read_Email towards host
      39.34.139.38 at 1/4/2012 16:02:01
```

Long_Term_SSH_Client_Activity

Secure Shell (SSH) is another form of network application logging performed by the PVS that the LCE can then consume to look for brute force SSH scanning. Below is an example sanitized log of an SSH session observed and logged by the PVS:

```
<36>Jan 03 08:33:15 pvs: 192.168.1.24:22|192.168.1.55:56630|6|5936|PVS-SSH-
      Server-Session_Start|SSH-2.0-OpenSSH_4.3{0a}||NONE
```

When the LCE receives these logs, it looks for SSH sessions being started by a potential client for long periods of time. A variety of automation software for Unix platforms may leverage SSH for communications. Unix scripting, SecurityCenter communication with the LCE and tools such as puppet leverage SSH for their underlying authentication and function. Below is an example alert created by the LCE after observing SecurityCenter log into an LCE several times continuously while building reports and dashboards:

```
Long_Term_SSH_Client_Activity - There has been 140 minutes of continuous SSH
      Client activity from host 192.168.41.229 and the most recent event was
      PVS-SSH_Server_Session_Start towards host 192.168.152.16
      (example.system.edu) at 1/6/2012 20:21:42
```

It is useful to leverage asset or directional filtering when creating alerts, dashboards and reports. As with other types of logs, the LCE considers the source IP address of the SSH client when performing this correlation.

Logging SSH connectivity passively has many advantages from pure system logging including:

- It logs both inbound and outbound connections, including those made from Unix and Windows servers.
- There is no need to place agents or collect logs from the monitored hosts, just packets.
- The connectivity is port and application independent. If the SSH server gets moved to a new port or a new program leverages the SSH protocol, the PVS will record it.

Long_Term_Statistical_Anomalies

The intent of this correlation rule is to leverage feedback between the LCE's continuous engine and the statistical event engine. A series of events could be anomalous, and the LCE will create alerts for those anomalies as such (see the next section which is dedicated to Statistical Anomalies). However, the anomaly engine works in chunks of 60 minutes. For any 60 minute period, it will tell you if activities are anomalistic or not. While useful, there was nothing that could look for multiple hours of anomalies occurring from a host.

To compensate for this, the LCE's continuous engine accepts any type of anomaly (Low, Anomaly, Medium or Large) to track anomalistic behavior for long periods of time. Below is an example log:

```
Statistics-Long_Term_Abuse - 1/8/2012 03:45:41 - There have been 3 or more hours
      of continuous statistical anomalies to 192.168.1.61 (Server2511.lab)
      which may indicate a suspicious increase in usage or worm activity
```

Long term statistical anomalies can indicate many types of change. For example, the addition of a new website to an existing web server would not only create new types of logs and usage patterns as new users connected to the website, there may be different types of anomalies throughout the day.

Long_Term_Threatlist_Activity

The LCE's ability to correlate normalized logs with IP address and domain names with reputations associated with botnets is extensively covered in the IP & DNS Reputation section. For continuous correlation, the LCE can find when there have been ongoing connections to or from known botnet addresses. Consider the following example sanitized log:

```
Long_Term_Threatlist_Activity - There has been 40 minutes of continuous
      threatlist activity from host 49.384.99.16 and the most recent event was
      Outbound_High_Port_Threatlist_Connection towards host 12.22.137.332
      (some.server.at.bigpond.net.au) at 1/6/2012 19:15:11
```

In this case, we have a host creating Outbound_High_Port_Threatlist_Connection events for long periods of time. This means that a host on our network is reaching out to a known botnet on a port larger than 1024 for at least 40 minutes. This type of determined and persistent connection can indicate a botnet.

Long_Term_Virus_Or_Malware_Activity

All events from the *virus* event category are used by the LCE to look for *continuous* activity associated with viruses.

If you have mail servers, you may see logs such as this:

```
Long_Term_Virus_Or_Malware_Activity - There has been 140 minutes of continuous
      virus or malware event activity from host 42.66.3.43 (bellatrix.mylab.ru)
      and the most recent event was ClamAV-Virus_Detected towards host
      42.66.3.43 (bellatrix.mylab.ru) at 1/7/2012 18:42:25
```

Servers that are exposed to malicious content with some form of anti-virus software will surely generate continuous logs that indicate viruses have been found and removed. Email servers with spam or anti-virus software fall into this category, as do web application gateways that scrub chat and web browsing traffic.

This type of correlation also pulls in events from host based anti-virus software as well as network IDS technologies such as Snort or TippingPoint.

Because of this, when setting up alerting and monitoring for this type of event, be sure to use directionality and source asset filtering so that you can alert on virus outbreaks on systems that are likely infected.

Long_Term_VNC_Client_Activity

As with the SSH and RDP protocols, the PVS can sniff the beginning of VNC remote desktop sessions and identify continuous activity, which may indicate brute force password guessing or network sweeps of VNC servers.

Long_Term_Web_Error_Activity

To help identify remote web site attackers, the LCE's continuous event filter is subscribed to *web-error* events. These error logs contain evidence of common web errors that we all see (such as the common 404 error) as well as errors specific to PHP, Apache, IIS and many other web technologies.

By looking for these types of errors on a continuous basis, the LCE can identify long term web application scanning and web probes. While performing a vulnerability scan or web application brute force testing, the majority of the logs aren't for valid web requests. Very often they make references to objects, URLs, web sites, forms, form variables, etc. that don't exist. The LCE can use continuous occurrences of hosts creating a long footprint of web errors to identify web application scans.

Below is a sanitized example log:

```
Long_Term_Web_Error_Activity - There has been 140 minutes of continuous web
      errors from host 42.36.27.10 (source.web.edu.ca) and the most recent
      event was Apache-Invalid_Method towards host 42.36.27.12
      (target.web.edu.ca) at 1/7/2012 20:57:57
```

The ability to discriminate long term errors being generated by potential hackers is important because Internet facing servers are constantly exposed to indexers and crawlers from a variety of sites.

Long_Term_Windows_App_Errors

Our last type of correlation for continuous events considers Windows event logs for hung or faulting applications. Below is an example log:

```
Long_Term_Application-Fault_Activity - There has been 40 minutes of continuous
      application-fault activity from host 172.20.100.76 and the most recent
      event was faulting-application towards host 172.20.100.76 at 1/7/2012
      22:53:02
```

Detecting faulting or hung Windows applications is a slightly different technique than looking for generic errors as with the Long_Term_Error_Activity event, but it's the same concept. If a host has either the same or many applications that keep hanging or faulting, the behavior can be identified with this correlation.

Faulting or hanging applications are not only an IT concern for the users, but they can also indicate the presence of malware, viruses and potentially hostile software.

In conclusion for this entire section, one more example is provided of how the continuous event type can be used to gain knowledge and situational awareness about the network. Below is a SecurityCenter CV dashboard component that highlights *continuous* activity.

This matrix component is included on the Indicators dashboard. Cells in this matrix component are highlighted if the specified long-term activity was detected in the last 48 hours. In SecurityCenter CV, clicking on a highlighted cell in this component will bring up the analysis screen to display details on the events. In the analysis screen, setting the tool to IP Summary will display the systems on which the long-term activity was discovered. Other tools such as Class C Summary and Asset Summary can also be used to determine where long-term activity is being detected on the network.

Statistical Anomalies

What is it?	Automatically identifies anomalies in network traffic, error logs, authentication sources, web browsing and many other types of sources.
What does it do?	For each host on the network, it compares the amount of every unique normalized event and client or server connection for that hour to that same hour on previous days and identifies statistical deviations for the lifetime of behavior for that host.
Why does it matter?	Network usage and behavior often changes based on traffic from the Internet, usage changes, new applications, new servers, attackers, compromised systems and many other reasons. Applying statistical analysis to the high volumes of traffic allows LCE users to understand and have situational awareness of how their logs indicate significant behavioral changes.
LCE Event Type	All correlated events are assigned to the *stats* (statistics) event type.
Statistical Events	The LCE does not perform statistical analysis of statistical events.
First Time Seen Events	First time seen events are generated for all statistical events. The event name is Never_Before_Seen-Statistical_Event and this will be generated each time a new event from the *stats* event family occurs for each host.
Continuous Events	The LCE searches for *continuous* events from the *stats* event types and produces a Long_Term_Statistical_Anomalies event if more than three hours of anomalies occur from a given host.

Statistical event and connectivity anomaly detection is a core form of correlation for the LCE. The intent of the LCE's statistical engine is to automatically define thresholds for any event or connection on any host and then identify when the threshold is crossed in a statistically significant manner.

What makes things more interesting (and more accurate) is that the LCE will automatically learn the thresholds for each host for each hour of the day. This ensures high accuracy in detecting subtle differences in behavior without having a user's morning habits bias the analysis in the afternoon.

Thresholds come in two types: event spikes and client or server connections. An event spike occurs when a host has had more than its expected amount of events in a given one hour period. Below is an example log indicating an increase in events for a given host:

```
stats: Jan 09 06:45:30 - 192.168.1.61 0.0.0.0 Statistics-web-access_Spike
       sip/dip SrcIp event PVS-Web_Query_Request window Jan 09 06:00:00 07:00:00
       average 0.46 stddev 3.12 nhits 84 stddev_units 26.74 freq 0.00
```

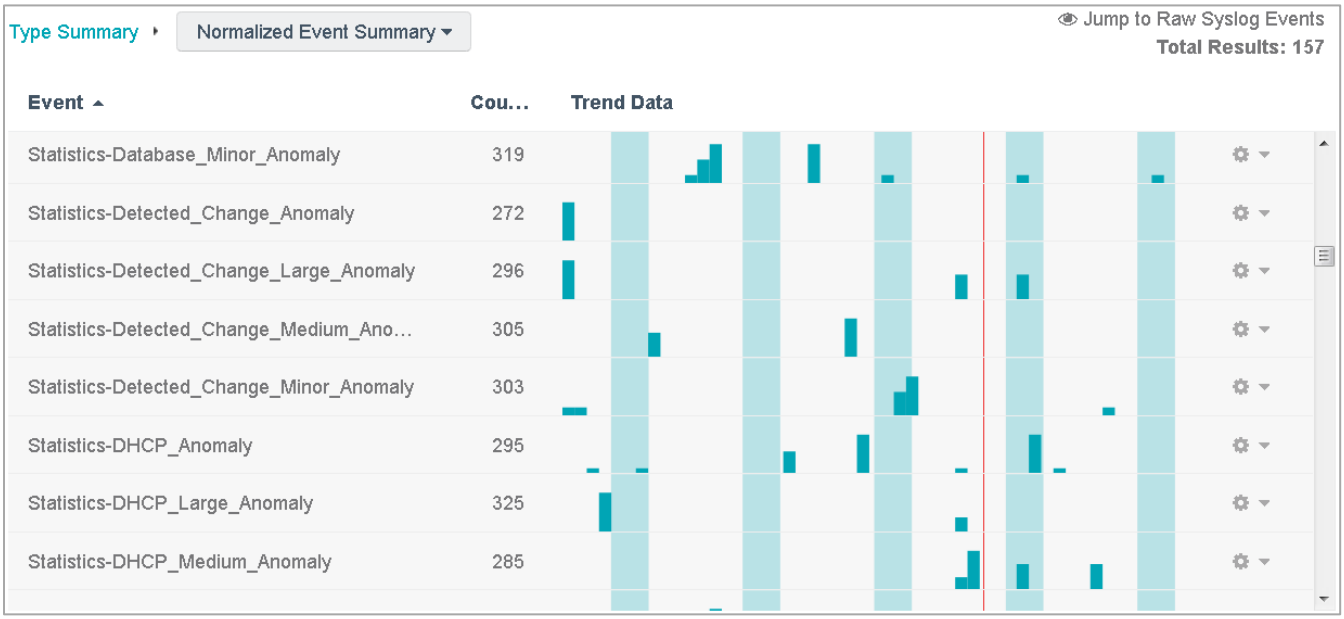
The IP address of the anomaly is in green and is host 192.168.1.62. The underlying event generated by the LCE is in blue and was named Statistics-web-access_Spike. The LCE's statistics engine generates normalized alerts based on the type of the event of the anomaly. This particular event anomaly was from a statistically significant number of PVS-Web_Query_Request events, which is of the *web-access* event type, hence the Statistics-web-access_Spike event name. This log was based on an event spike and has the "event" keyword in the log. We will see a log next that is purely based on connectivity analysis and this keyword will not be present. Finally, we see in red the actual number of standard deviation units – 26.74 in this case. Those 26.74 standard deviation units were the result of seeing 84 actual events (the "nhits" value) when on average this host see 0.46 events of these types per hour with a standard deviation of 3.12. This log was also performed by creating a statistical analysis of other events of the PVS-Web_Query_Request for host 192.168.1.61 during the times of 06:00 AM though 07:00 AM.

Working with event spikes is very useful because they generally tell you more information about an event you already know about, but didn't know there was an anomaly. The LCE can also perform connectivity analysis on any type of log that has a source and destination IP address. These work with firewall logs, NetFlow, PVS network traffic, logs from the Tenable Network Monitor, email logs and much more. Below is an example log for connection analysis:

```
stats: Jan 09 11:45:30 - 192.168.1.61 0.0.0.0 Statistics-
       Connection_Initiation_Spike window Jan 09 11:00:00 12:00:00 average 1.32
       stddev 14.65 nhits 177 stddev_units 11.99 freq 0.00
```

This log doesn't say which events contributed to the correlation, just that there was an increase in connection initiations. According to the log, host 192.168.1.61 normally averages 1.32 connections with a standard deviation of 14.65, but today's 177 connections places it 11.99 standard deviations from normal, creating the anomaly.

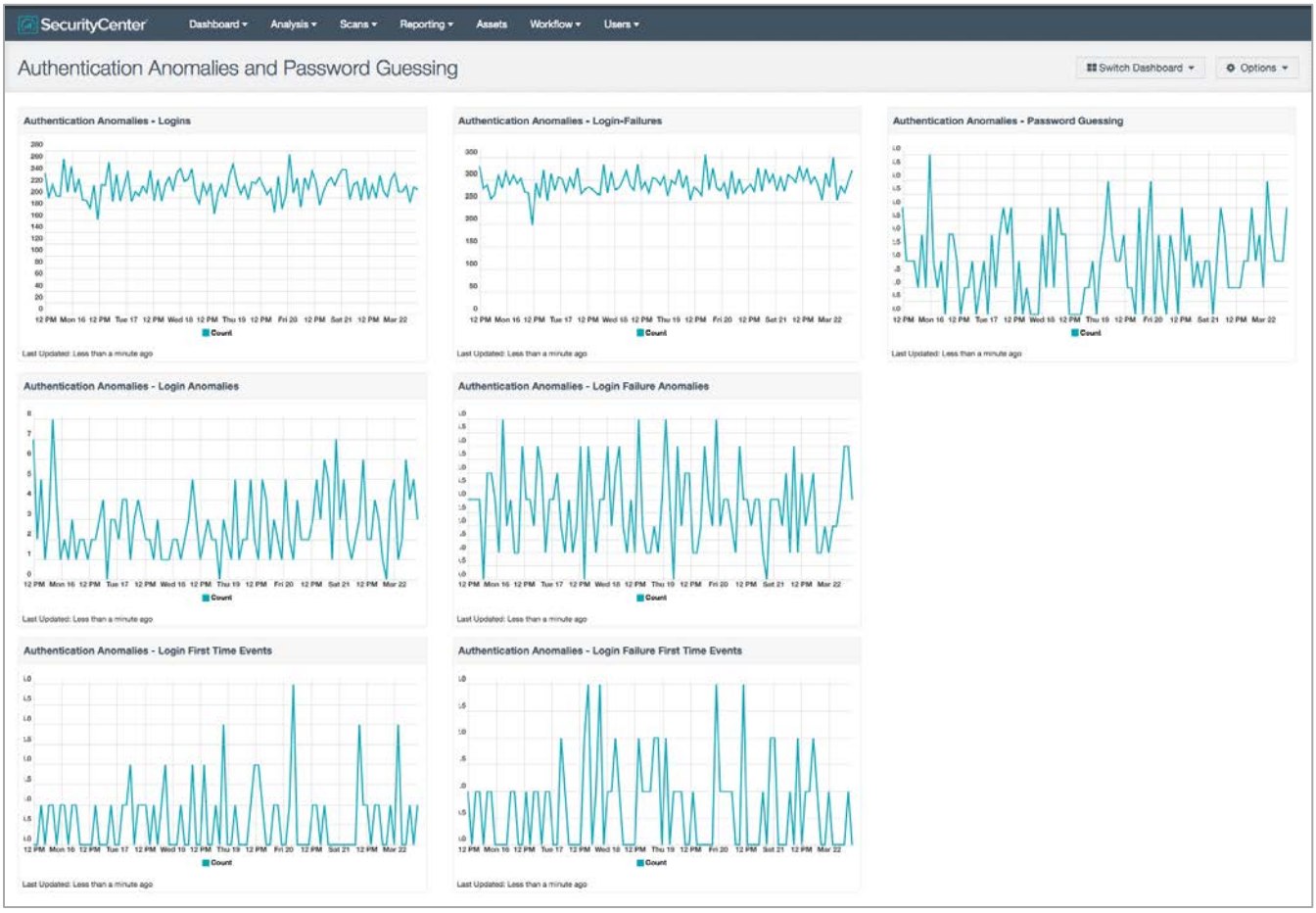
The LCE assigns names to the *stats* events based on the magnitude of the anomaly and the underlying connectivity direction or event type. Below is a screen capture of some example statistical anomalies from small lab network:



The magnitude of the event name (Low, Medium, etc.) is based on the numeric value of the standard deviation units as follows:

- 1-5 Minor Anomaly
- 6-9 Anomaly
- 10-100 Medium Anomaly
- 100-1000 Large Anomaly

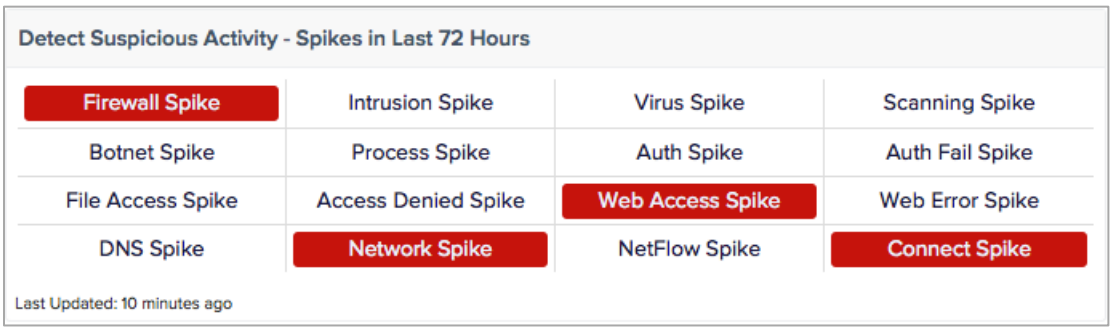
A technique to filter alerts, reports and dashboards to only highlight certain types of anomalies is to leverage SecurityCenter CV's ability to perform partial string matches on event name. For example, in the screen capture below, a dashboard is shown (from the Tenable dashboard repository) that has two components (in the middle) for *login* and *login-failure* event type anomalies:



This was created by leveraging a trend filter that set the *stats* event type and also used the string "*Login_Failure*" as shown below:

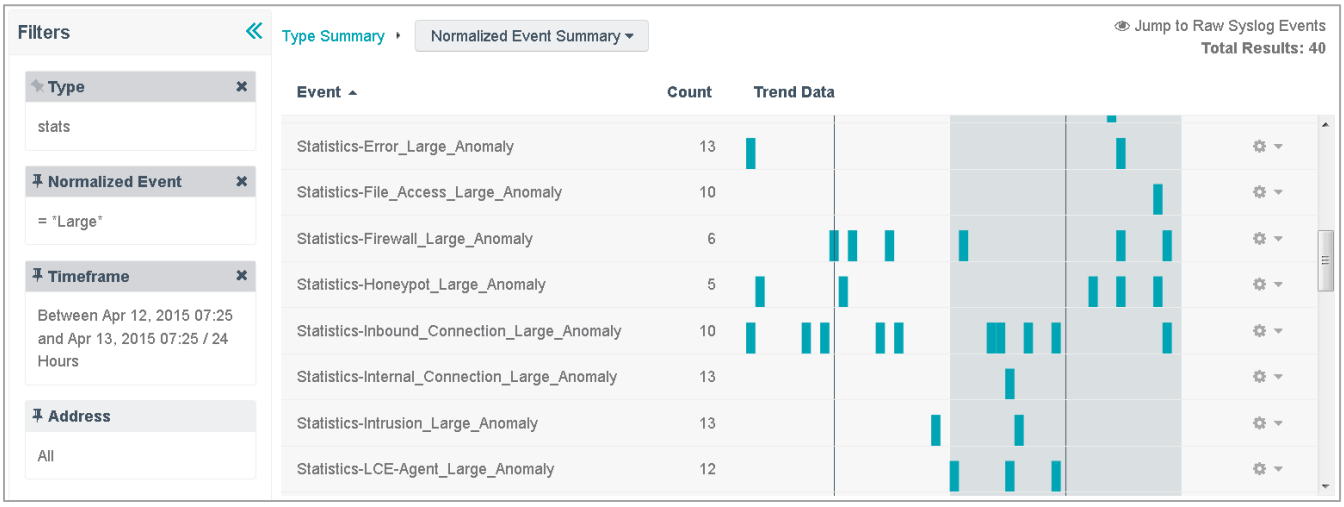
This would cause the query to match any of the Statistics-Login_Failure_Minor_Anomaly, Statistics-Login_Failure_Anomaly, Statistics-Login_Failure_Medium_Anomaly or Statistics-Login_Failure_Large_Anomaly event names.

The SecurityCenter CV dashboard component below highlights large event spikes in several areas. Note that LCE reports additional event spikes that are not included in this component.

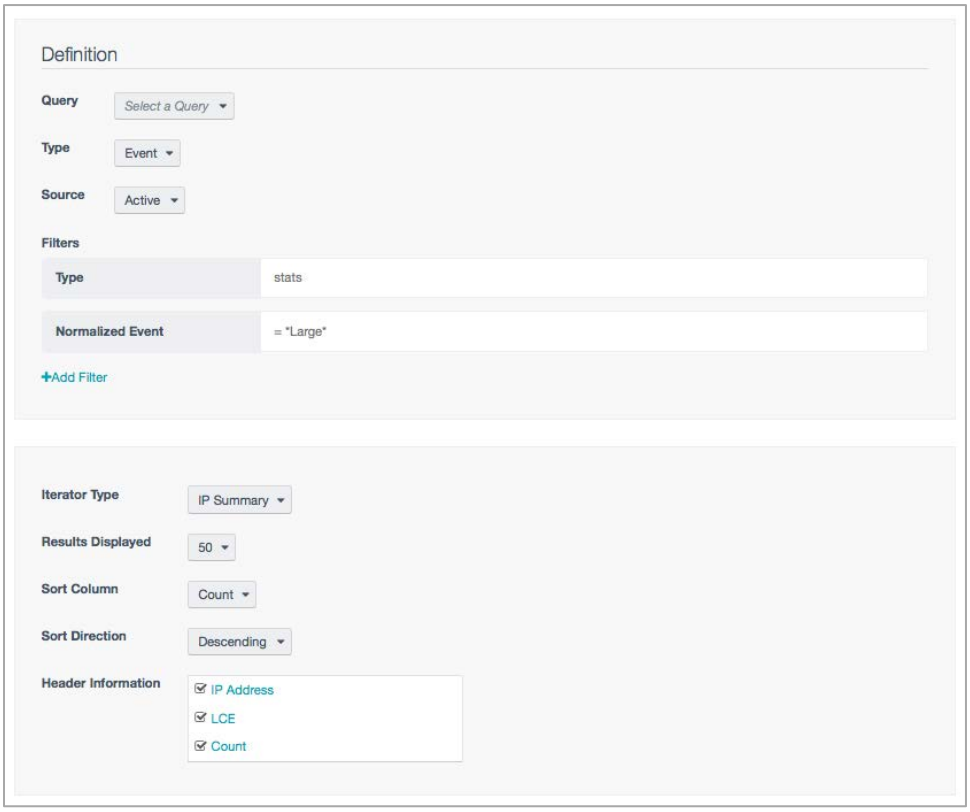


This matrix component is included on the Detect Suspicious Activity dashboard. Cells in this matrix component are highlighted if the specified spike in events was detected in the last 72 hours. In SecurityCenter CV, clicking on a highlighted cell in this component will bring up the analysis screen to display details on the events. In the analysis screen, setting the tool to Raw Syslog Events will display information on the events including the time, IP address and statistical information detailed earlier.

Queries to look for *any* large anomalies can be created by leveraging a filter on the *stats* event type and a string of "*Large*". Below is a screen capture from a network that implemented a similar type of filter:



Having the LCE leverage its iterator function can save you a lot of time when performing manual analysis for reporting. Below is a screen capture of a typical iterator function used to set up a report based on “Large” anomalies when they occur:



The use case is to take all of the IP addresses that have been reported to have had a large anomaly (see the *stats* event filter and the “*Large*” text filter) and then iterate through each of those IP addresses and produce individual chapters. With the iterator used here, a full syslog chapter was set up to read the large anomaly log that occurred, and then a full event summary was used to visually look at the event trace and see what types of events were occurring that may have caused the anomaly.

From a forensics and situational awareness point of view, anomalies that occur for various event types deserve some suggested causes for what might be occurring. Below is a chart that shows the root event type without the severity of the anomaly:

Root Event Type	Potential Causes for Anomalies
access-denied	New systems and applications that create *access-denied* events may also cause anomalies. New forms of internal scanning, worm outbreaks, unauthorized user access attempts and even new security policies can also create new types of *access-denied* events that can lead to detection of anomalies.
application	Any change in the usage of an application such as new software, using new modules of an application or simply more user requests can create new *application* logs that in turn create anomalies.

connection	The LCE also considers *connection* event types as discrete events and processes these like any other event type. It is very common to see statistical anomalies in both *connection* events such as a VPN network session being established and the LCE's generic connection initiation and reception anomaly detection. Changes in connections can be produced by new systems, new applications, changes in the network firewall policy, addition of new types of logging (such as adding a NetFlow monitor) and more nefarious reasons such as compromised systems performing scanning, systems dramatically increasing the amount of clients connecting to them, compromised systems sending volumes of new SPAM emails and botnet command and control sessions.
continuous	Correlation *continuous* events are also fed into the LCE's statistics engine.
data-leak	Spikes in *data-leak* events could indicate that someone is indeed sending data that contains sensitive information at a rate that generates an anomaly.
database	Changes in usage patterns to databases (such as setting up a new table or a new web application) can often cause changes and anomalies in the database alerts. If a website's database queries are driven by internet usage, and there is an increase, this may correspond to an increase in *database* events.
detected-change	Large software installations, bulk changes to servers and patch updates can all make changes to a system. A compromised system that installs new users, creates file integrity alerts or installs new servers will also create alerts. All change should be audited, but in reality, an attacker will likely make many changes in a manner not normally performed by your IT staff and will create anomalies.
dhcp	Spikes in DHCP leases and renewals can also generate anomalies. If there are more laptops in the office this week because of visiting employees, you may see more DHCP logs and therefore more *dhcp* events that may cause anomalies. A malicious attacker that exploits a desktop or virtual server may attempt to renew DHCP leases in order to hijack IP addresses or gain access to other VLANs.
dns	The LCE processes spikes in DNS queries and lookup failures in the same manner. A spike in DNS queries could mean that a user is browsing many more new sites than normal. It could also mean that an email server is attempting to resolve many remote DNS names to send a new volume of email. Spikes in DNS resolution errors often mean misconfigured DNS information or a compromised system participating in a botnet or sending spam email.
dos	Increases in the expected amount of denial of service events could indicate a determined attack. Typically, if you have any *dos* events at all, they are likely "false positive" alerts. Seeing a spike in traffic from what is expected can help identify determined attacks.
error	There are many types of errors and alerts that indicate problems. Using the LCE to observe spikes in *error* events helps identify major problems. Problems can be created by attacks, hardware failures, misconfigurations, user error and many other factors.
file-access	The LCE normalizes file sharing events such as the PVS's ability to sniff PDF files shared over Windows file sharing. So when there is an anomaly that needs to be investigated – what type of event was it? Which system(s) were involved with the sharing? Is this a user collecting PDFs and other documents before leaving the company? The LCE can also detect legitimate spikes in access to files, such as when the HR department posts a new spreadsheet for an expense form and it is downloaded by the entire sales force.

firewall	Spikes in *firewall* event traffic often are the result of new deny rules or traffic patterns, which are normal but cause denied firewall logs to be created. Large spikes in firewall traffic can also be the result of a compromised host attempting to communicate outbound when there are specific outbound rules in place. Internal probing, even from an employee, will also cause denied firewall events to occur at times they normally didn't occur.
honeypot	A honeypot, by its very nature, logs traffic that should not be occurring. Increased traffic to a honeypot could indicate new probes, internal scanning and worm outbreaks.
intrusion	When attackers launch scans and attacks against internal or external targets, the footprint of *intrusion* events will create an anomaly if there is a significant enough spike in the traffic. These events are directional in nature, so an LCE may generate an alert when there is a spike if a system is attacked as well as one if the system is attacking other systems. The automatic identification of *intrusion* events that is outside of the "normal" false positive landscape is very useful if you are faced with intrusion detection technologies that are hard to tune or have high false positive rates.
lce	The *lce* event category is diagnostic information about LCE clients and LCE system events. Any anomalies in these events could indicate configuration changes that cause issues with LCE client connection to the LCE as well as LCE operation.
login-failure	Anomalies in *login-failure* events indicate brute force password guessing. They can also indicate when a host has been automated to access many other systems with credentials and the credentials have changed.
login	Anomalies in *login* events indicate an over-usage of an account. For example, a hacker could compromise someone's server account and then there could be a spike in access to that account with hackers share that information and it is used by many others.
logout	Spikes in *logout* messages could indicate system-wide issues that cause all users to be logged out at the same time.
nbs	Anomalies in *nbs* events are most often associated with new systems coming online and the LCE learning all of these events about that system. For a new server, system or application, everything it does for the first time will be treated as an *nbs* event by the LCE. During this time, those events will likely be recognized as an anomaly. Once a system has been running for some time, any spike in *nbs* events should be investigated as it will indicate a large change in how the system is being used. As *nbs* event can come from any type of normalized log, seeing an anomaly in them indicates a major change in behavior, errors, usage or activity.
network	Detecting anomalies with network traffic means that there has been a change in the network connections, session length or session bandwidth detected by the Tenable Network Monitor, the Tenable NetFlow Monitor or the Passive Vulnerability Scanner. Anomalies are created for spikes in network lengths of a certain type, such as detecting an increase of 10 minute sessions. Anomalies are also created for session bandwidth (i.e., 100 MB sessions) as well as discrete network classifications of sessions, such as when the PVS identifies an Xbox Live login starting. By identifying changes in network usage, many types of changes can be detected and the size of the anomaly prioritized for investigation. Big changes in network usage can come from new user behavior, compromised systems, sending spam, launching BitTorrent servers and many other factors.

process	The LCE logs which executable files, services, daemons and programs have been run and if they have crashed or exited. Detecting spikes in these types of events can indicate that a system is being used in a different manner than previously. If enough *process* events occur to create an anomaly, this may indicate a new type of program, a new type of error, a user running a program in a new way or even malicious software.
scanning	Spikes in port scanning events are obviously associated with an increase in port scanning in general. Worm outbreaks, botnet infections and unauthorized vulnerability scanning are all examples that can create spikes in port scan events. If you have an intrusion detection system that isn't tuned, it may also detect port scan events from various protocols such as Voice over IP, Skype and network management systems.
spam	The LCE creates normalized *spam* logs from email server logs and specific spam filter applications. Tracking the amount of spam your network has received and anomalies in the blocked spam is useful for gauging how well your anti-spam defenses are. However, a spike in *spam* events can also indicate that there is a spike in normal levels. Since most spam systems are only so efficient, this can corresponding indicate that more spam messages got through during this time. If your email server processes local logs that are flagged as being spam related, then it may also indicate that one of your local systems has been compromised and is sending spam email.
system	Analysis of *system* logs for anomalies can lead to interesting results, but it indicates a change in how the system was used. This is often associated with hardware failures, configuration changes, new applications and sometimes infected malicious software.
threatlist	Detecting a spike in *threatlist* events means that there was an anomaly in the number of connections that originated to or from a potential botnet IP address. For outbound *threatlist* connections, this could indicate that a host is heavily communicating with a known hostile IP address. For inbound *threatlist* events, it could indicate that one of your Internet facing web servers has been scanned at a rate much higher than previously recorded. If you have many *threatlist* events both inbound and outbound from your network, analyzing anomalies in this traffic can identify your top potential sources of infection and attack.
usb	The *usb* event type indicates an insertion or removal of a USB device into a computer. A high number of these could indicate that a user is manually connecting and reconnecting devices.
virus	The *virus* event type includes events from anti-virus agents, email anti-virus technologies and from network IDS devices. As such, any spike in these events likely indicates that a virus has propagated or that there has been an increase in the amount of malicious software that has been received and processed.
vulnerability	The *vulnerability* event consists of logs that indicate the presence of newly discovered vulnerabilities, primarily from the Passive Vulnerability Scanner. Spikes in this type of traffic are associated when a new host has been found on the network and the PVS reports everything it knows about it all at once. These alerts can also occur if a previously known host launches an old and out-of-date web browser or if a previously vulnerable server that was disabled instead of patched was re-launched.

web-access	Web browsing, web server logs and sniffed web sessions are all candidates for anomaly detection by the LCE. Spikes in *web-access* events can indicate a busier web server, more Internet browsing and even more trips to Facebook. Anomalies from this type should be analyzed based on the source of the network. For example, you may have a DNS server that reaches out to Microsoft over the web for patch updates and then see a spike one day if the administrator of that server logs onto a Facebook account from that system. The spikes in web traffic on the surface always indicate an increased amount of connections to the web. With some further analysis, they can also identify many types of abuse.
web-error	Spikes in *web-error* traffic indicate that a web server is generating error logs based on queries to it. These can occur when an attacker is probing for potentially vulnerable web pages that don't exist, when a web application is referencing web sites and objects that don't exist and even when websites outside of your network incorrectly link to website URLs that have been removed.

Working with anomalies within the LCE can lead to a much deeper understanding of your network situational awareness and can also help identify a wide variety of abuses that would otherwise go undetected.

Anomalies can be used to look for a wide variety of behaviors and understand how your network is performing compared to previous days. Changes in activity don't necessarily mean that your network has been attacked, or that there have been major changes in how your users are behaving, but they do provide a great deal of understanding as to what is occurring on the network.

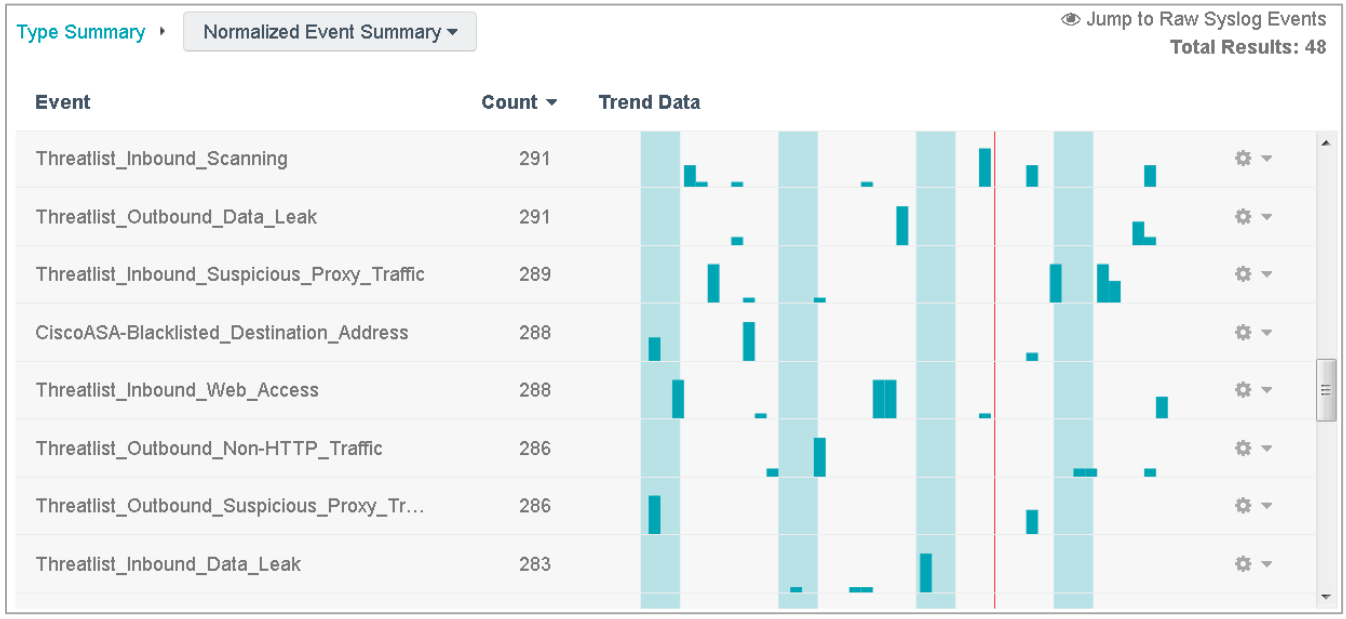
IP & DNS Reputation (Botnet Detection)

What is it?	Identifies when systems on your network connect to hostile botnets and when hostile botnets connect to systems on the network.
What does it do?	IP addresses and DNS names extracted from many different types of normalized logs are compared against a daily list of highly accurate known botnet sources.
Why does it matter?	Identifies compromised systems that are under control of a known botnet.
LCE Event Type	All correlated events are assigned to the *threatlist* event type.
Statistical Events	For each monitored host, the LCE will track the statistical occurrences of all unique *threatlist* events and generate one of the following events, based on the size of the anomaly: Statistics-Threatlist_Minor_Anomaly, Statistics-Threatlist_Anomaly, Statistics-Threatlist_Medium_Anomaly orStatistics-Threatlist_Large_Anomaly
First Time Seen Events	For each monitored host, the LCE will generate a never before seen event of the *nbs* event type named Never_Before_Seen-Threatlist.
Continuous Events	The LCE will recognize ongoing *threatlist* communications and generate a Long_Term_Threatlist_Activity event under the *continuous* event type.

Each LCE receives an updated daily list of IP addresses and DNS names that are participating in known botnets. This list is highly accurate and is modified very often. A system that has been made part of botnet today might remain on a list for just a few days or for months.

As the LCE normalizes connections, web logs, intrusion events, network sessions logged by the PVS, logins, login failures and NetFlow records, it checks the IP address of each log to look for any connections to or from your network. Any event that has a source or destination IP address matching on this list has a new event generated for the LCE event type category of *threatlist*.

Below is an example screen capture of *threatlist* events:

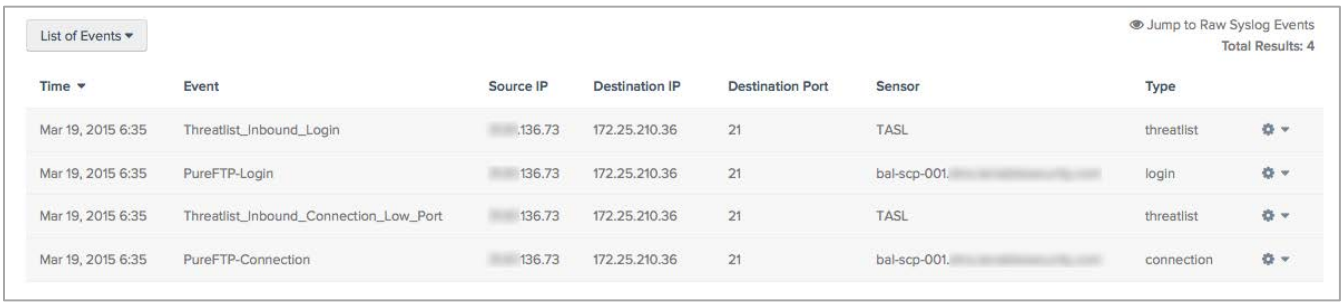


In this particular view, each row represents one type of activity on the threatlist for the past 30 days. The very first row indicates that "Scanning" (such as port scanning) has occurred 291 times in the time frame and that the source address of these scans was from an IP address on the list. It's possible that this LCE had many other scanning events, but none of the IP addresses were on the threatlist.

The *threatlist* event type names are loosely based on the specific LCE event type that has occurred. In some cases, generic events, such as network connections, are further named for common ports and directionality. Ports such as HTTP, HTTPS and SMTP have distinct names and other ports are normalized to Low_Port or High_Port depending if they targeted ports above or below 1024. Drilling into an Inbound_Threatlist_Connection_Low_Port event and then performing a port summary on a live LCE, we see that the targeted ports are indeed below 1024:

It's important to understand that threatlist events are created by the LCE. A log may originate from a source such as a Snort sensor, but when the normalized IP addresses are compared against the list of IP addresses that are known to be botnets, a new log is created.

For example, in the sanitized screen capture below, an FTP connection from an IP address known to be part of a botnet was captured:



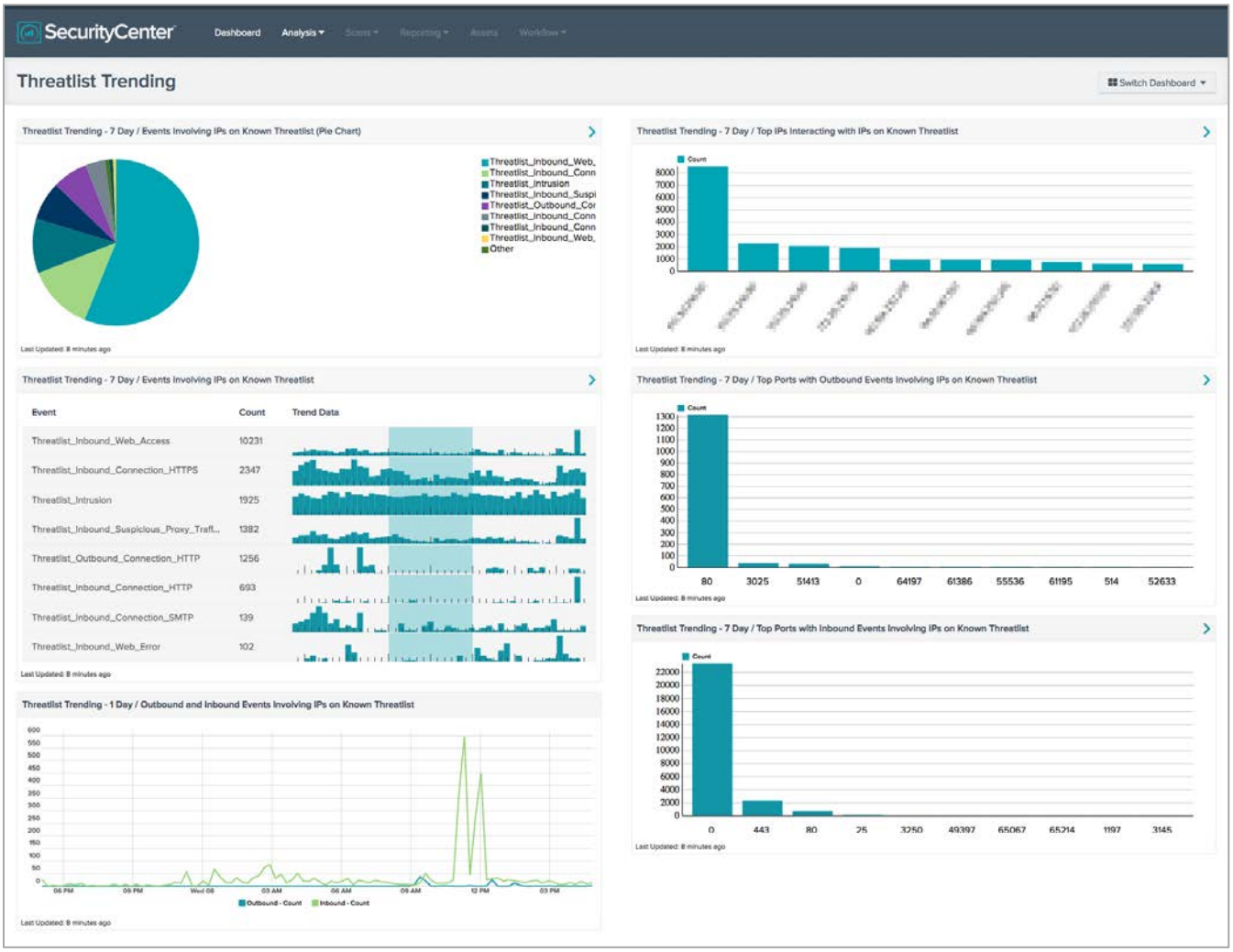
Each of the *threatlist* events were generated based on a previous log entry. The login event "PureFTP-Login" was created when this log was received (sanitized):

```
Mar 19 06:35:39 bal-scp-001 pure-ftpd: (?@31.61.136.73) [INFO] cust44333 is now
       logged in
```

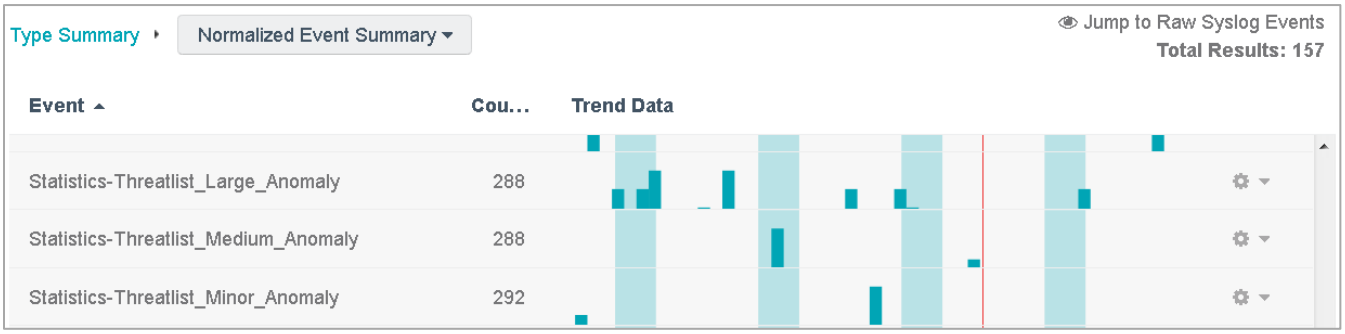
And this in turn produced the following log that was normalized to the event "Threatlist_Inbound_Login":

```
Threatlist_Inbound_Login 31.61.136.73:0 to 172.25.210.36:21 identified as
      Spam_Bot , the event was triggered by a PureFTP-Login event
```

Creating reports, alerts or dashboards based on *threatlist* events is very useful. The Threatlist Trending SecurityCenter CV dashboard shown below presents threatlist events over a 7-day period and displays the top IP addresses and ports associated with this network activity. In addition, a 1-day trend of outbound and inbound threatlist activity is presented.



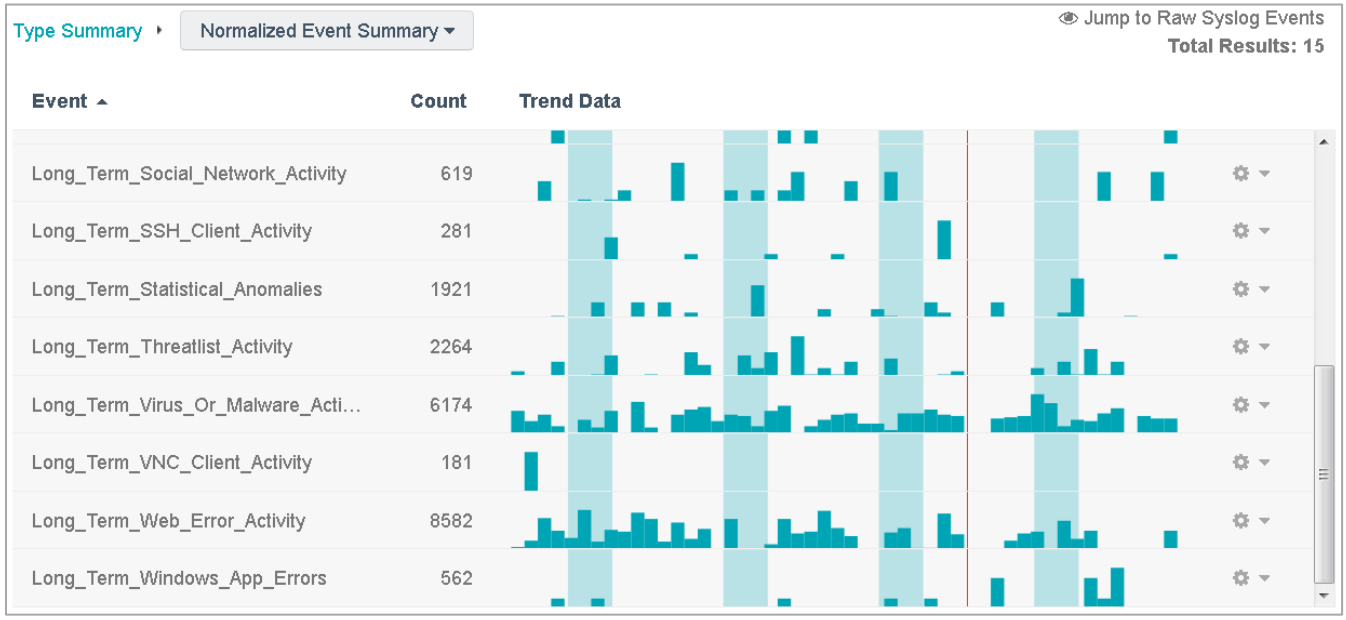
The LCE's statistical anomaly engine will also trigger on *threatlist* event types. Below is a screen capture of a minor Threatlist anomaly:



This event was generated because host 172.69.32.118 had a *threatlist* event named Inbound_Threatlist_Connection_Low_Port occur during an hour of the day that typically has far fewer occurrences and was flagged as a minor anomaly. The actual log associated with that event occurred is shown below:

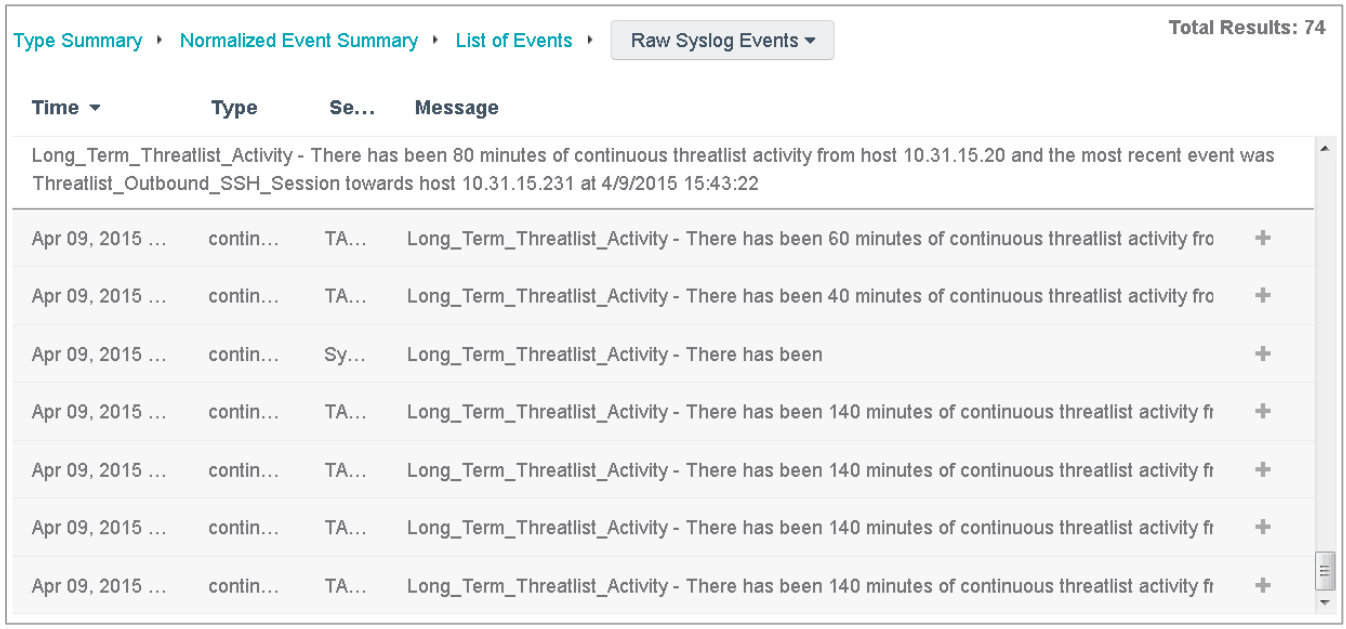
```
stats: Jan 03 22:47:52 - 0.0.0.0 172.69.32.118 Statistics-threatlist_Spike
       sip/dip DstIp event Inbound_Threatlist_Connection_Low_Port  window Jan 03
       22:00:00 23:00:00 average 5.67 stddev 13.47 nhits 47 stddev_units 3.07
       freq 0.02
```

Finally, the LCE's continuous activity detection engine also tracks hosts that are experiencing *threatlist* events for extended periods of time (longer than 20 minutes). In this screen capture below, there is a variety of *continuous* events occurring during a 24 hour period:



The *continuous* event names are all given the phrase "Long_Term" in their event name. The event associated with continuous threatlist events is known as Long_Term_Threatlist_Activity and there were 2264 of these events in the above screen capture.

Below is an example sanitized screen capture of example logs associated with *continuous* events of the *threatlist* type:



Detecting Valid Attacks in Intrusion Detection Logs

What is it?	Identification of attacks in intrusion detection logs that indicate a high chance of compromise.
What does it do?	Correlates attack logs from network intrusion detection systems with vulnerability data from Nessus scans, Nessus patch audits and real-time sniffed client and server vulnerabilities from the Passive Vulnerability Scanner to identify attacks that will likely succeed against their targets.
Why does it matter?	Many network intrusion detection products have high false positive rates and auditing all of these logs can be very tedious.
LCE Event Type	No event type is created. However the “Targeted IDS Event” filter can be used to select IDS events that correlate with known attacks.
Statistical Events	There are no statistical events generated for correlated IDS events.
First Time Seen Events	There are no first-time-seen events generated for correlated IDS events.
Continuous Events	There are no continuous events generated for correlated IDS events.

Network intrusion detection devices are designed to detect attacks. Unfortunately, they do not always consider that the target system may not be vulnerable to the attack or the detection could be poorly written. Both of these facts create events that don’t affect the actual security of the target network and create work for security analysts to process.

However, the LCE, through SecurityCenter Continuous View, can perform correlation between the latest detected vulnerabilities on the network with meta-data about IDS events. This allows the LCE to log which IDS events likely would work (or may have worked) against their targets while ignoring attacks that would never have worked.

This type of correlation is very important because all of the underlying intrusion events are still available for first time seen, continuous and statistical analysis. These types of correlations allow better understanding of when you are attacked and who is attacking you but they aren't accurate enough to say that a given attack worked against a given target.

For server attacks, correlation occurs on the port level. For example, if a Windows computer is seen to have a vulnerable IIS Web server on port 80 but not on port 443, the LCE will only alert if a relevant attack against port 80 is logged. For client attacks against email, browser, chat and other non-server applications, vulnerabilities from Nessus patch audits and client-side detected vulnerabilities with the PVS are correlated independent of the port logged by the IDS sensor.

The chart below details the types of vulnerabilities detected by Nessus and the Passive Vulnerability Scanner as they are relevant accurate forms of VA/IDS correlation:

Desired VA/IDS Correlation	Recommended Source of Vulnerability Data
Attacks against servers	• Recent un-credentialed Nessus scans targeting all ports • Passive Vulnerability Scanner data • Recent credentialed Nessus patch audits • Recent Nessus scans that reference patch management systems
Attacks against clients	• Passive Vulnerability Scanner data • Recent credentialed Nessus patch audits • Recent Nessus scans that reference patch management systems

A common misconception with VA/IDS correlation is that un-credentialed scans will detect client-side vulnerabilities. Some clients and some operating system security issues are indeed detected by un-credentialed scans, but issues with missing email and web browser patches are commonly not part of un-credentialed audits.

Adding credentials to Nessus scans allows for the identification of issues with Thunderbird, Internet Explorer, Outlook, Chrome, Adobe PDF and Flash, Java and much more. Nessus scans can also be augmented with access to patch management systems such as Red Hat Satellite or Microsoft SCCM. These types of scans will identify client vulnerabilities in the underlying technologies, such as pulling in Outlook patch information from SCCM, but won't identify other client-side issues such as missing patches in an anti-virus agent or a third-party web browser.

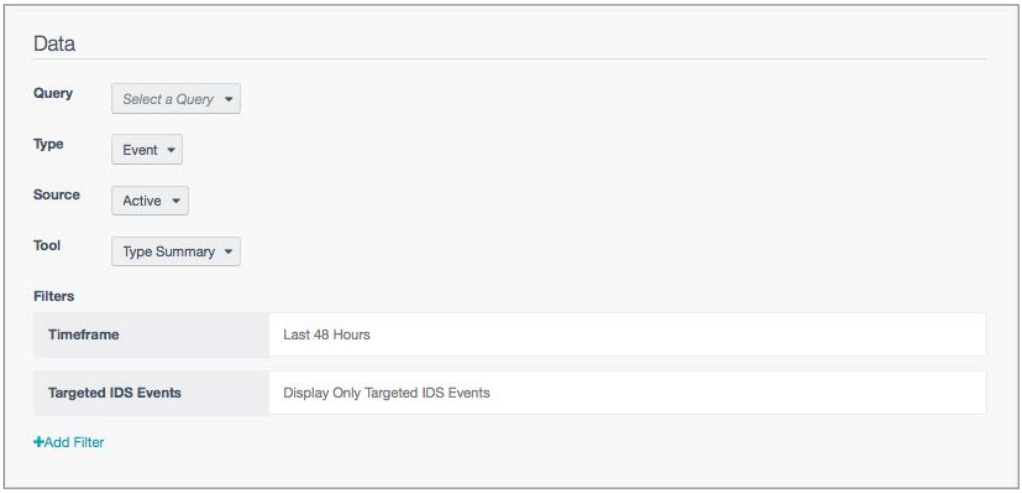
Server audits also benefit from increased correlation accuracy because of "back porting". While scanning a host, if Nessus identifies a system running "Apache 2.0" in the banner, it may indeed be version 2.0. However, it may also really be version 2.0.14, which includes 14 different patch releases. Because of "back porting", un-credentialed scans can't differentiate between a patched and unpatched system. Adding in credentials for patch auditing for services of both the operating systems and patch management systems increases the results and accuracy of the Nessus report and therefore the VA/IDS correlation value.

By passively monitoring continuous network traffic, the Passive Vulnerability Scanner will identify a variety of client and server issues. It has the same back porting detection issues as previously discussed, but it also has an advantage of seeing any network traffic to any port, such as web server running on port 40000.

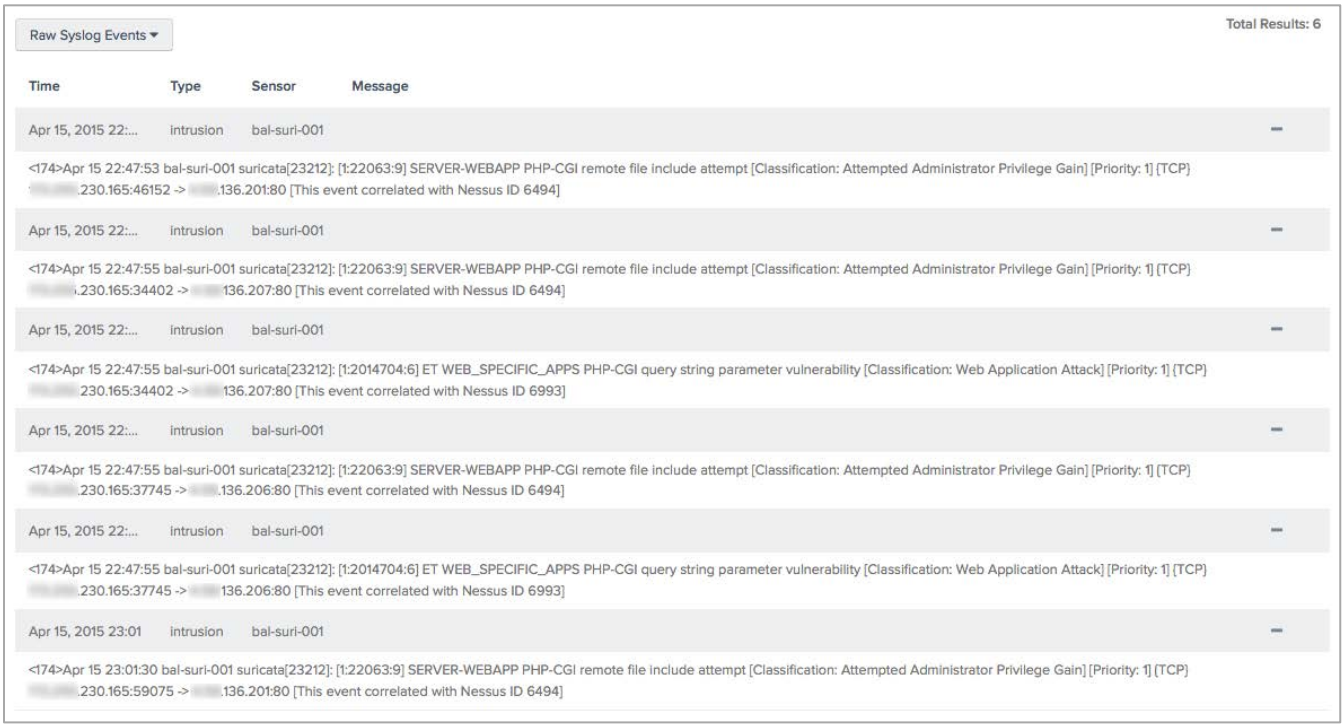
The following network IDS technologies are supported for full VA/IDS correlation:

- Cisco IDS/IPS
- Open Source Snort instances running Sourcefire VRT rules
- Logs from Sourcefire sensors and management consoles
- Juniper IPS
- TippingPoint IPS

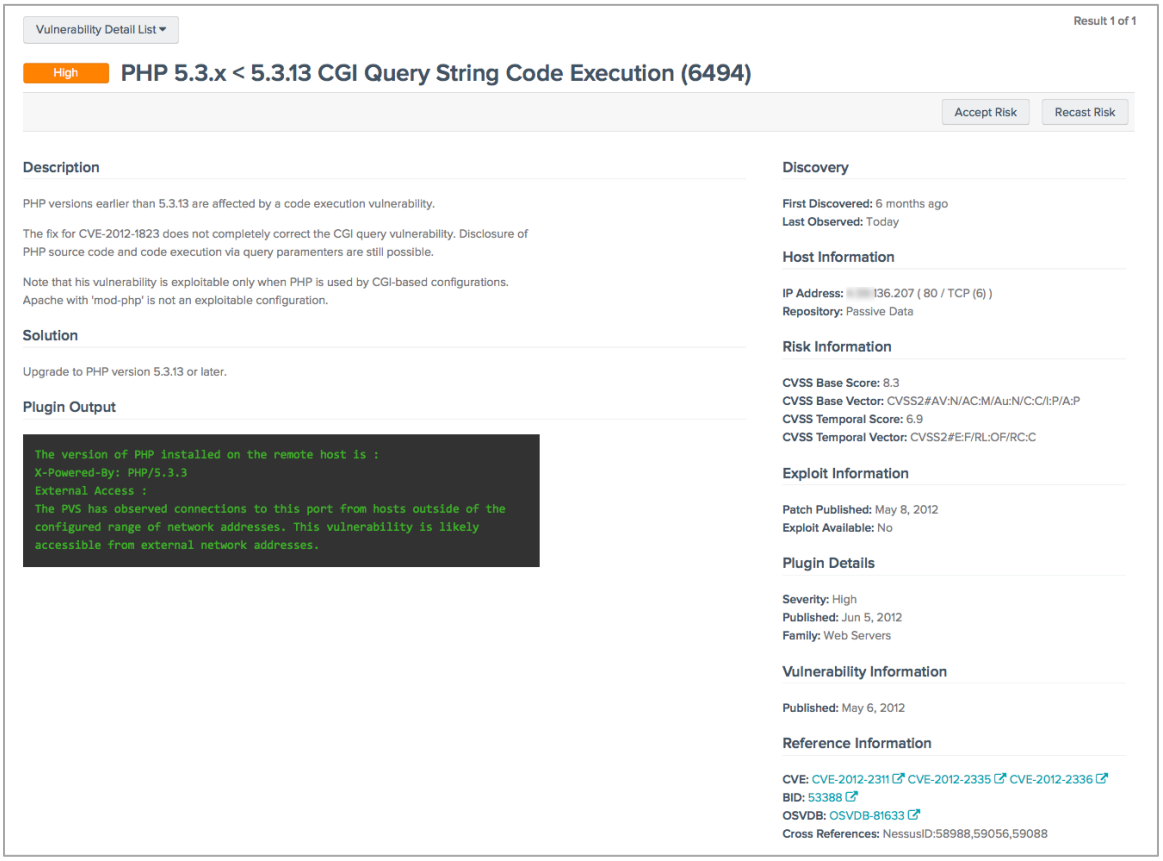
Targeted IDS events can be used to create dashboards, reports, analyze logs and create alerts. Targeted IDS events record *intrusion* or *network* type activity against a system that can take advantage of the vulnerabilities present on that system. The “Event Filters” tab has an option to specify filtering of “Targeted IDS Events”. Below is a screen capture of the dialog:



Events of this type consist of the original IDS message and an additional message about the correlated vulnerability. Below is an example screen capture:



In this case, the host with the IP address ending in 230.165 launched an attack against the IP addresses endng with 136.207 that was decoded by Snort as a “SERVER-WEBAPP PHP-CGI remote file include attempt”. The LCE correlated this alert with a relevant vulnerability and appended the message to the original syslog. In this case, the correlated Nessus plugin ID was 6494. A vulnerability screen capture for this record against the IP address ending in 136.207 is shown below:



There are a variety of use cases for tracking these VA/IDS correlated alerts discussed in the following table. Combinations of alerting, dashboard trending and reports can help drive understanding and detection of attackers.

Use Case	VA/IDS Benefit
100% passive network monitoring of large Internet facing network that is mostly unmanaged	Running an IDS in front of an unknown and unmanaged network will generate many alerts. Adding in passive discovery of the network and VA/IDS correlation will help identify real attacks and reduce the amount of work load associated with intrusion analysis.
Internal unpatched systems and internal vulnerability scans	Running an IDS or IPS on the inside of a network may be a compensating control for also not patching servers as often as policy mandates. Vulnerability scans of the internal infrastructure allow VA/IDS correlation to be used to identify real attacks.
Client side exploit detection	Recent credentialed Nessus scans or Passive Vulnerability Scan data identifies client vulnerabilities that can then be correlated with IDS events to detect attacks against web browsers and other types of clients.

Penetration tester detection	Sanctioned penetration testing teams will likely launch exploits for which the targets are indeed vulnerable. The detection of this type of event with VA/IDS correlation on a corporate network may be a good indicator of a penetration testing team's efforts when they start to target their attacks.
Vulnerability and IDS event coverage	If a sanctioned penetration testing team does indeed compromise a target and there is not an intrusion detection log or a correlated VA/IDS event, there may be issues with the coverage of your IDS monitoring or vulnerability scanning.

Change Detection

What is it?	Automatically identifies changes to the network, operating systems, applications, invoked executable programs, file modifications and user settings.
What does it do?	The LCE tracks normalized events that indicate change and alerts accordingly. It also keeps tables of gathered information, such as observed user logins, and alerts when a "new" user has been identified.
Why does it matter?	Manually tracking all of the different types of changes that can occur is humanly impossible. Placing these events in a single LCE type category makes it easy to create dashboard, reports, alerts, etc. to gain a deeper understanding of change detection.
LCE Event Type	The LCE type of *detected-change* is used for the majority of changes detected. New recognized programs are sent to the *process* event type.
Statistical Events	If a host generates a series of detected-change events that are an anomaly, the LCE will create a Statistics-Detected_Change_Minor_Anomaly, Statistics-Detected_Change_Anomaly, Statistics-Detected_Change_Medium_Anomaly or a Statistics-Detected_Change_Large_Anomaly event depending on the size of the anomaly.
First Time Seen Events	If a type of change is detected from a system for the very first time, an *nbs* event named Never_Before_Seen-Change_Detected_Event will be generated by the LCE.
Continuous Events	There are no statistical events generated for continuous *detected-change* events.

Detecting changes in complex networks, servers, applications and user profiles is very difficult because there are many types of logs that indicate different types of change. There are also many, many logs that occur on a daily basis. The LCE creates new *detect-change* events when it observes all types of change.

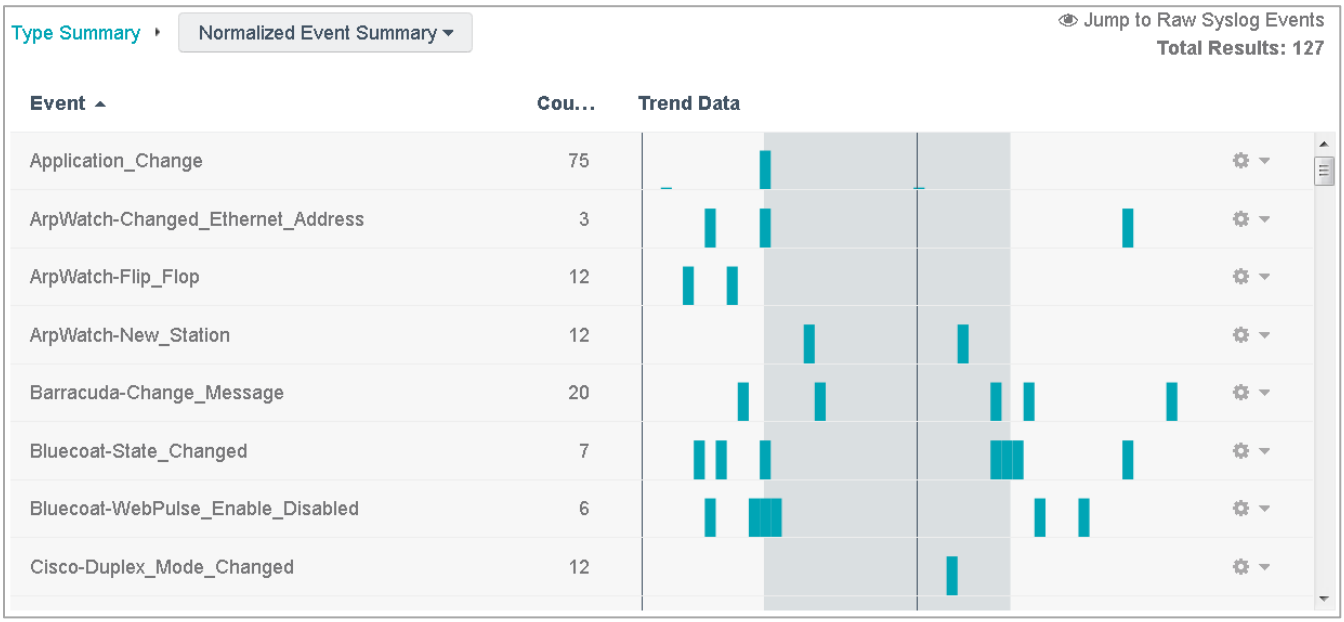
This is a core form of correlation for the LCE because detecting change is very important for the following reasons:

- New changes can introduce security risks
- Changes may be unauthorized
- Changes may violate your compliance profile
- Changes made have been made by hackers or malicious insiders
- Legitimate software may make unknown changes

In all cases, the earlier a change is detected, the more quickly it can be validated.

Below is an example Type Summary screen capture from a network that has had more than 4100 *detected-change* events in the past 24 hours:

Those particular *detected-change* events were comprised of several dozen different types of specific events. A screen capture of the first page is below:



The LCE processes logs for general indications of change. These events have discrete event names generated by the LCE specifically to report these general event types. Below is a table of the event names and a short description of each.

Static Change Detection	Description
Application_Change	Highlights when configuration changes are made to applications such as SecurityCenter
Database_Change	Indicates the detection of new database schemas
Device_Change	Changes to running configurations of devices such as wireless access points, VPNs, load balancers, etc.
Firewall_Change	Changes to firewalls including configuration changes and modified firewall rules
Network_Change	Detections of new hosts, new open ports, new browsed ports, new trust relationships, new routes, etc.
Router_Change	Indicates when any type of configuration change in a router log has been detected

Server_Change	Identifies general changes to running processes, setting promiscuous mode, writing to the Windows registry, etc.
Software_Installed	Any type of software installation log, including kernel patches and adding web browser plugins
Software_Removed	Any type of software removal log
Switch_Change	Indicates when any type of configuration change in a switch log has been detected
Time_Update	Creates an event any time the system time has been modified, including slight changes by the network time protocol
User_Added	Identifies any time a new user has been detected through some sort of system or application log
User_Changed	Identifies any time an existing user has had a modification to their access, security policy, etc.
User_Removed	Identifies any time an existing user has been removed from a system

The LCE also performs a variety of in-depth log processing to extract new types of information. Unlike the above table's events that corresponds specifically to certain types of single logs, these events below dynamically learn data from previous logs to identify "new" types of change.

Dynamic Change Detection	Description
New_Command	Learns commands, processes and executable files that run on each host and creates a new event when a new command is encountered
New_MAC	Learns Ethernet addresses from PVS, wireless access points, switches and DHCP logs. It creates a new event when a new Ethernet address is encountered
New_User	Dynamically learns users that exist on servers and applications and alerts when it recognizes a valid login from a new user

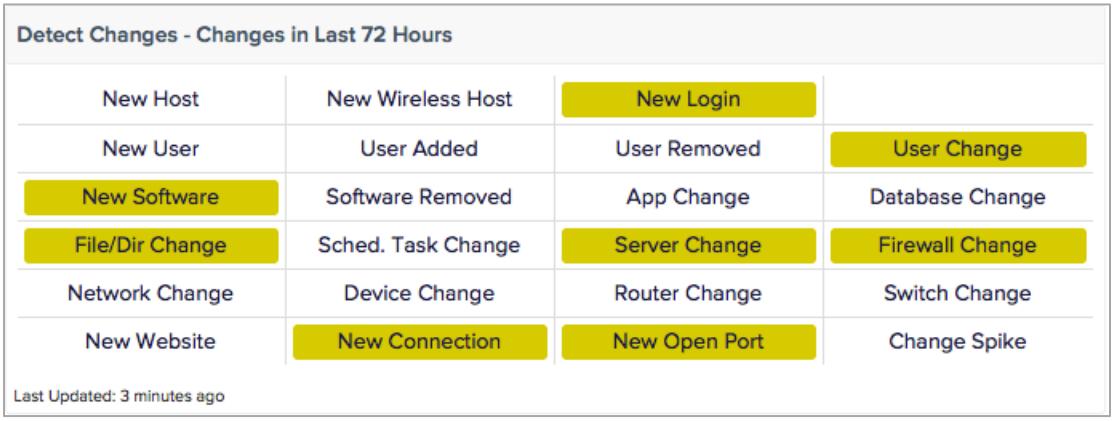
The LCE can also leverage LCE Clients for a variety of Unix and Windows operating systems to monitor systems for changes. Below is a table of supported change detections that occur only with local client monitoring:

LCE Client Change Detection	Description
File Integrity Detection	Each LCE Client for Unix and Windows operating systems can be configured to report file creation, removal and changes for specific files or directory structures. These events are normalized to the *detected-change* event type and start with the "LCE-" prefix. An example event name is "LCE-Windows_Executable_File_Modified", which indicates a Windows **`.exe`** or **`.dll`** file was modified.
USB Insertions and Removals	For Windows operating systems, the LCE Client will report when a USB device is attached and when it is removed. These particular events are sent to the *usb* LCE event type.

Detecting change can be used for a variety of use cases.

When combined with asset filtering, change for specific types of servers, desktops and other technologies can be identified. These can be used to drive reports, alerts and dashboards.

Detecting change can be used for a variety of use cases. The SecurityCenter CV dashboard component below highlights *detected-change* events that occurred in the last 72 hours.



This matrix component is included in several dashboards. Cells in this matrix component are highlighted if the specified change was detected in the last 72 hours. For example, user account creation and changes will be reflected in the New User and User Added/Removed/Change cells, and software inventory and file integrity could be monitored using the New Software, Software Removed and File/Dir Change cells. In SecurityCenter CV, clicking on a highlighted cell in this component will bring up the analysis screen to display details on the events. In the analysis screen, setting the tool to Raw Syslog Events will display information on the events including change details, time and IP address.

Finally, when an anomaly occurs based on detected-change event spikes, or when a detect-change event occurs for the first time, these can also indicate changes that are out of the ordinary and could be investigated. The events to consider in this case are as follows:

- Statistics-Detected_Change_Minor_Anomaly
- Statistics-Detected_Change_Anomaly
- Statistics-Detected_Change_Medium_Anomaly
- Statistics-Detected_Change_Large_Anomaly
- Never_Before_Seen-Change_Detected_Event

These events are ideal to create alerts and reports against key assets as they indicate that new forms of change have been detected.

V. Tactical Event Correlation

The LCE includes a variety of event correlation technologies that are specific to an area. For example, in the core sections, the correlation applied generically to many types of normalized events, but in this section, these tactical correlation functions typically apply to just one type of discipline. For each of these sections, a summary chart is displayed, and example logs and screen captures are given.

Determined Scan and Attack Detection

What is it?	Identifies attempts at multiple different attack types against one system or sweeps against your network of different host targets with one attack type.
What does it do?	The LCE monitors *intrusion*, *login-failure* and *web-error* logs to identify determined attackers.
Why does it matter?	Most networks have a high occurrence of *intrusion*, *login-failure* and *web-error* logs making sweeps and determined scans hard to find.
LCE Event Type	Specific events are associated with these correlations including: Intrusion_Host_Scan, Intrusion_Network_Scan, Web_Server_Scan, Web_Servers_Scanned, Network_Login_Sweep and Password_Guessing.
Statistical Events	Each of the above events will be considered for statistical anomaly correlation. The event type is *intrusion*. Any anomalies would result in a *stats* event such as a Statistics-Intrusion_Medium_Anomaly, depending in the magnitude of the statistical deviation.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Intrusion_Event log would be issued.
Continuous Events	The LCE will only consider continuous *intrusion* events. Any of the events for this type of correlation will be considered for continuous *intrusion* event detection.

Each type of correlation for attacks, web errors and login failures will be discussed separately.

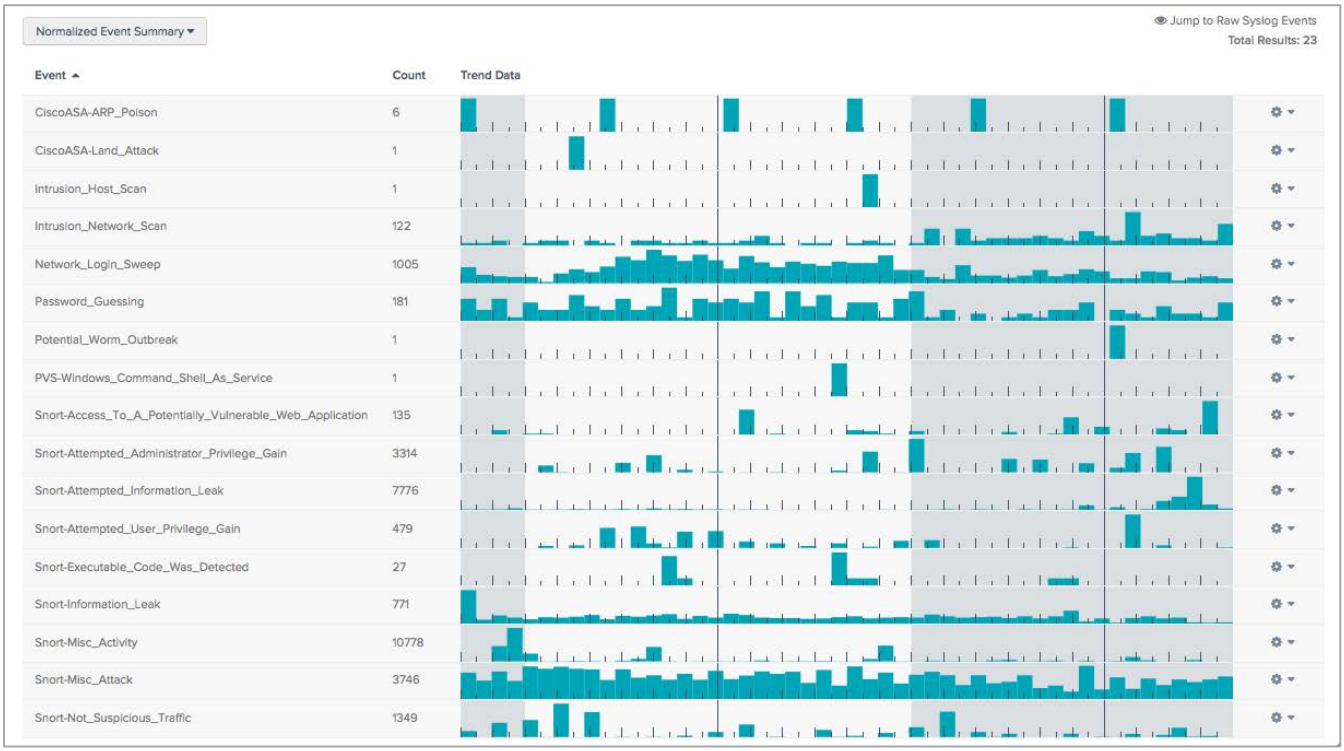
Intrusion Logs

There are two types of correlated events generated by the LCE for processing *intrusion* events:

- **Intrusion_Host_Scan** – The LCE detected a series of *intrusion* events that indicated that one host was trying different attacks against another.
- **Intrusion_Network_Scan** – The LCE detected a series of *intrusion* events that indicated that one host was trying the same attack against many target systems.

If you have performed penetration testing or have looked at logs during an incident response process, you will know that these two patterns occur quite often. Attackers may try different types of attacks to compromise a host. These are reflected by different types of intrusion detection alerts. Attackers may also expect that a certain percentage of hosts on a target network are vulnerable to an exploit and “sweep” the network in an automated fashion.

Below is screen capture of an LCE monitoring a network that is collecting logs from a Snort IDS:

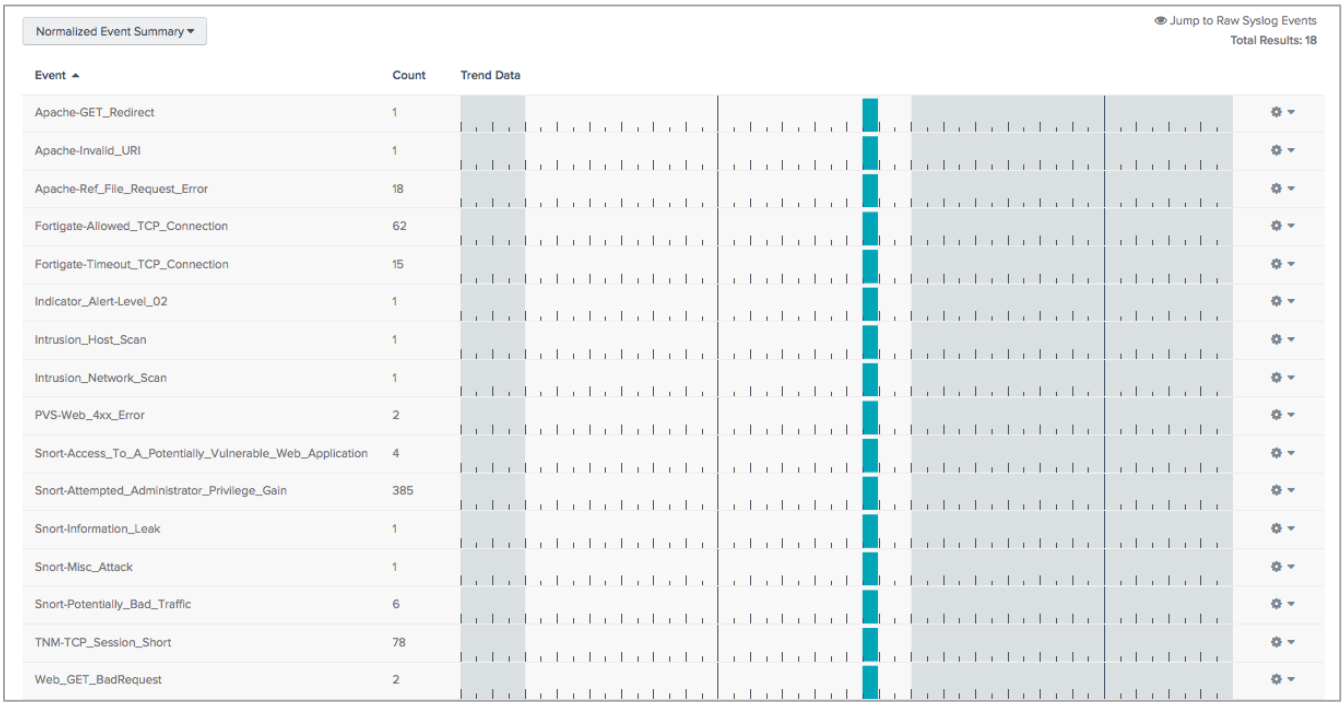


The two correlated events can be seen near the top of the screen.

Below is an example log of an Intrusion_Host_Scan:

```
4/8/2015 16:41:16 Intrusion_Host_Scan ###.###.121.11 attacked or probed
      172.30.210.60 (chi.example.com) and generated these 4 types of intrusion
      events: Snort-Misc_Attack, Snort-Attempted_Administrator_Privilege_Gain,
      Snort-Information_Leak, Snort-
      Access_To_A_Potentially_Vulnerable_Web_Application
```

The log indicates that a host (IP address ending with 121.11) attacked another host and tried a variety of different attack techniques that manifested themselves into the listed Snort IDS events. A common form of investigation is to take the source IP address and see its footprint of events in the LCE. An example screen capture is shown below:



Event	Count	Trend Data
Apache-GET_Redirect	1	
Apache-Invalid_URI	1	
Apache-Ref_File_Request_Error	18	
Fortigate-Allowed_TCP_Connection	62	
Fortigate-Timeout_TCP_Connection	15	
Indicator_Alert-Level_02	1	
Intrusion_Host_Scan	1	
Intrusion_Network_Scan	1	
PVS-Web_4xx_Error	2	
Snort-Access_To_A_Potentially_Vulnerable_Web_Application	4	
Snort-Attempted_Administrator_Privilege_Gain	385	
Snort-Information_Leak	1	
Snort-Misc_Attack	1	
Snort-Potentially_Bad_Traffic	6	
TNM-TCP_Session_Short	78	
Web_GET_BadRequest	2	

The vertical sequence of events occurring at the same time can also be seen in the previous original summary of *intrusion* events. This type of correlation is a great event for alerting, dashboards and reporting.

Below is an example log of the Intrusion_Network_Scan:

```
4/8/2015 16:41:06 Intrusion_Network_Scan ###.###.121.11 has performed a network
      probe or scan which generated intrusion events across the following
      systems: ##.##.149.100 172.30.210.40 ##.##.149.120 172.30.210.60
```

This log indicated that the host (IP address ending with 121.11) also attacked multiple targets on the local network.

Both of these logs are directional in nature. They will accurately identify internal scanning, inbound scanning, as well as outbound scanning. The correlation is also best-effort and is dependent on the uniqueness of the received *intrusion* events, the total number of *intrusion* events and the distribution of scan targets.

Web Error Logs

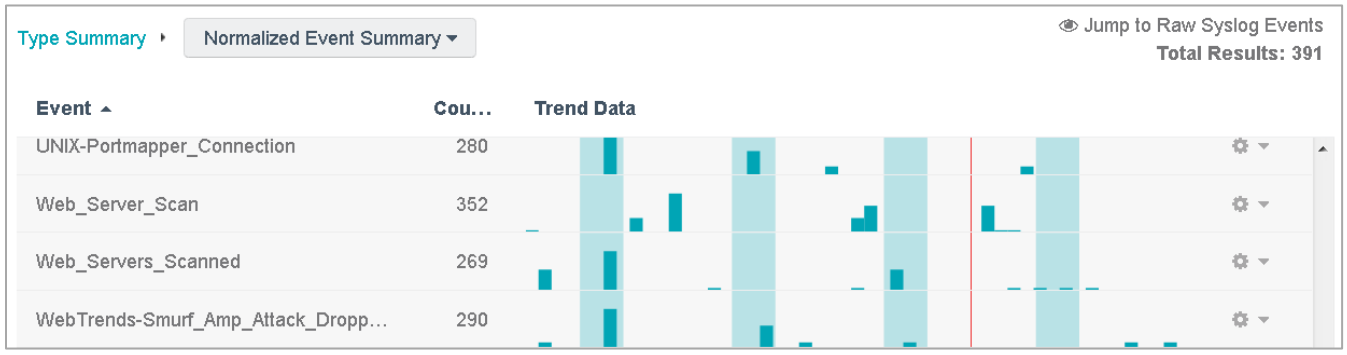
Web application attacks, web application penetration, web-borne malware looking for new targets to infect, and vulnerability scans create a footprint of web error logs. Known web attacks are indeed detected by network intrusion detection devices if the attack is not sufficiently complex or hidden by encryption over an SSL connection.

The LCE processes web logs to look for errors generated by web requests that do not result in a typical “200” web code. The LCE recognizes logs that indicate traditional web errors in Apache, IIS and other web server logs such as a “404” web page not found event. The LCE also recognizes logs from PHP and various web servers that indicate errors in rendering objects and web resources.

The LCE recognizes when a single IP address creates a unique footprint of different *web-error* events on one host or a single *web-error* event on multiple web servers. The events generated by the LCE are:

- **Web_Server_Scan** – A single IP address created multiple unique *web-error* events on one web server.
- **Web_Servers_Scanned** – A single IP address created multiple *web-error* events on multiple target web servers.

Below is a screen capture of a live media network's intrusion events for a five day period:



The 352 Web_Server_Scan events had logs that indicated a local web server was indeed scanned. An example log format for this event is as follows:

```
1/11/2012 05:12:20 Web_Server_Scan ##.###.##.## (adk-khmer-win7.cia-example.org)
      has performed a web app scan of 192.168.1.106 (web01.my.org) and
      generated these 4 unique types of web errors: Web_GET_Forbidden,
      Web_GET_ServerError, Web_GET_UnauthorizedRequest, Web_GET_PageNotFound
```

In this case, the redacted IP address above had four unique types of *web-error* events often associated with web vulnerability scanning occur while visiting 192.168.1.106. This likely indicates a probe of some sort. Using the redacted IP address as a pivot, the following screen capture of a query for that IP's *web-error* event activity for the same period of time was performed and is shown below:

It looks like this particular remote IP address has visited our target system more than once and the behavior of the types of *web-error* events is different.

An example log for the Web_Servers_Scanned event is as follows:

```
1/14/2012 01:02:50 Web_Servers_Scanned ###.###.##.### (s2b.lab.someplace.com)
      has performed web app scans of these servers: ###.###.10.11
      ###.###.14.217 ###.###.14.219
```

The log indicates that one particular system had a footprint of *web-errors* on three visited web servers.

Both of these event types are very useful to observe the security of web sites. They have many advantages in use including:

- Since these are based on logs, targeted IP address are known to be some sort of web server that creates web logs.
- Normal web browsing does not involve the creation of web errors. Unless a website is broken, having different types of web-error events is a strong indicator that a site is being probed or under heavy attack.
- Unless your NIDS can decrypt web SSL queries, watching web logs from a secure web server may be your only option to really understand what was asked and how the web server responded.
- With just a few queries, any of these events can be used to identify what the potential attacker was using to communicate to the web server. This can identify the actual query that was performed that resulted in an error, and also identify potentially benign web traffic that had created errors on your web server.

Login Failure Sweeps

As the LCE processes *login-failure* event types, it keeps track of the host initiating the login. If a single hosts attempts to log into a large number of target hosts and fails, the LCE will issue a Network_Login_Sweep alert. Below is an example log of this type:

```
Network_Login_Sweep - There have been 7 login failures in the last hour against
      multiple target hosts. The most recent login failure was from
      ##.##.##.###.example.net to 49.84.247.117 my-fusion.school.edu with a
      login failure event of Windows-Account_Expired
```

Here is a slightly different log based on a Snort login-failure detection:

```
Network_Login_Sweep - There have been 5 login failures in the last 24 hours
      against multiple target hosts. The most recent login failure was from
      49.284.22.108  to ###.##.###.##  with a login failure event of Snort-
      An_Attempted_Login_Using_A_Suspicious_Username_Was_Detected
```

The LCE will display the last system for which the login was attempted to and the login type. This correlation does not consider brute force password guessing (see the Successful and Unsuccessful Password Guessing section later).

Login failure sweeps can occur for many legitimate reasons including:

- Software that performs discovery, such as network management and software inventory systems
- Un-credentialed vulnerability scanning
- Credentialed vulnerability scanning that has an incorrect password

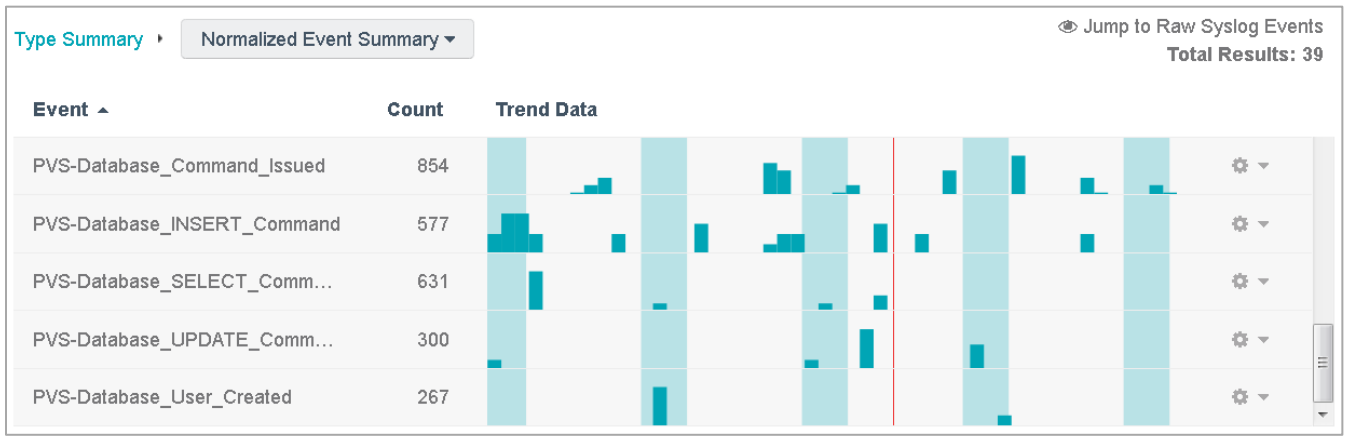
The Network_Login_Sweep is useful for many reasons including:

- It is independent of the *login-failure* event type. A hacker who has failed to log into Windows server and has moved onto a Linux server will still have the Windows login failure count against them.
- It is directional in nature, so the inbound, internal and outbound filtering can be leveraged.
- For hosts that are truly noisy, perhaps infected with a virus, the LCE's statistics, continuous and first time seen events will likely generate correlations based on this event.

SQL Injection Detection

What is it?	Identifies potential SQL injection attacks.
What does it do?	The LCE performs a variety of checks on observed SQL queries to identify malicious attempts to run commands and extract data from a SQL site.
Why does it matter?	SQL injection is a common attack leveraged against custom web applications powered by databases.
LCE Event Type	The LCE produces the Suspicious_SQL_Query_Detected event and several others (detailed below) when it detects evidence of SQL injection or suspicious SQL queries.
Statistical Events	Since the Suspicious_SQL_Query_Detected event is an *intrusion* event, any anomalies associated with it will be correlated by the LCE's statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Intrusion_Event log would be issued.
Continuous Events	The LCE would consider these events for continuous activity since it is part of the *intrusion* family.

The primary source of SQL activity logs for consumption by the LCE is the Passive Vulnerability Scanner. It can log unencrypted SQL traffic and log it to the database LCE event type as shown below in this screen capture:



When a SQL query log arrives at the LCE, it will be analyzed with a variety of pattern matching, regular expressions and analytics that will result in one or more of the following events:

- Suspicious_SQL_Query_Detected
- Suspicious_SQL-Command_Execution
- Suspicious_SQL-Benchmark_Delay
- Suspicious_SQL-Meta_Characters_Seen
- Suspicious_SQL-CONCAT_Command_Seen
- Suspicious_SQL-Write_Output_to_File
- Suspicious_SQL-User_Database_Dump
- Suspicious_SQL-Injection_Attack_Detected

These events are all contained in the LCE *intrusion* event type category and will be used for correlation with first time seen, continuous and statistical event spikes. Below is an example LCE screen capture of an *intrusion* normalized event type summary that contains a series of Suspicious_SQL-Meta_Characters_Seen events:

Dashboards, alerts and reports, including the report iterator can leverage these types of events by employing event filters for the string “Suspicious_SQL*” or by filtering on specific event names.

Network Outage and Crash Detection

What is it?	Detects when many of the local systems all reboot, crash or have a major error all in common.
What does it do?	The LCE tracks logs that indicate reboots, system crashes, application faults and other types of critical errors across the network occurring in the last five minutes. When many systems all report similar errors, an alert is issued indicating a network-wide issue may be occurring.
Why does it matter?	Network wide issues are hard to spot if there are many different types of errors occurring on a daily basis.
LCE Event Type	The LCE produces the Multiple_System_Crashes event and this is normalized as a *process* event type.
Statistical Events	Since the produced event is a *process* event, any anomalies associated with it will be correlated by the LCE's statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Process_Event log would be issued.
Continuous Events	The LCE does not consider *process* events for continuous activity.

Below is an example log produced by this form of correlation:

```
Multiple_System_Crashes - There have been 5 hosts that have generated error
      events in the last 300 seconds.  This could indicate a failed attack,
      virus outbreak or crashing application. The affected hosts are:
      172.20.101.158 172.20.101.207 172.20.100.128 172.20.130.149 172.20.101.41
```

As the log indicates, crashes like this could indicate a variety of issues including virus outbreaks, bad software installations, flawed Windows group policy object (GPO) configuration changes and many more. Analysis of these types of logs can be assisted by the LCE's ability to summarize process execution as well as hung process reports on a daily and hourly basis. This is covered in the Process Executable Summary Reporting section. Analysis can also be aided by checking to see if there had been any continuous events associated with crashing Windows applications or high CPU loads. This was previously covered in the Continuous Activity Detection section.

Below is an example screen capture of an alert occurring on a network that gathers process execution logs from Unix and Windows hosts:

This particular alert normalization takes the source IP address of the most recent alert. For detailed analysis, the list of IPs could be cut and pasted into a static asset list for bulk analysis of all *process*, *system* and *error* event types associated with the alert.

New Hosts Port Scanning

What is it?	Detects when a system that was recently added to the network initiates port scans.
What does it do?	The LCE correlates the reception of a PVS-New_Host_Alert followed by any port scanning events from that host.
Why does it matter?	Mobile systems that are taken home, to conferences, to coffee shops or other potentially hostile places can become infected. Upon reconnection, the host may start to probe the local network.
LCE Event Type	The LCE produces the PVS-New_Host_Portscanning event and this is normalized as a *scanning* event type.
Statistical Events	Since the produced event is a *scanning* event, any anomalies associated with it will be correlated by the LCE's statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Scanning_Event log would be issued.
Continuous Events	The LCE would consider these events for continuous activity since it is part of the *scanning* family.

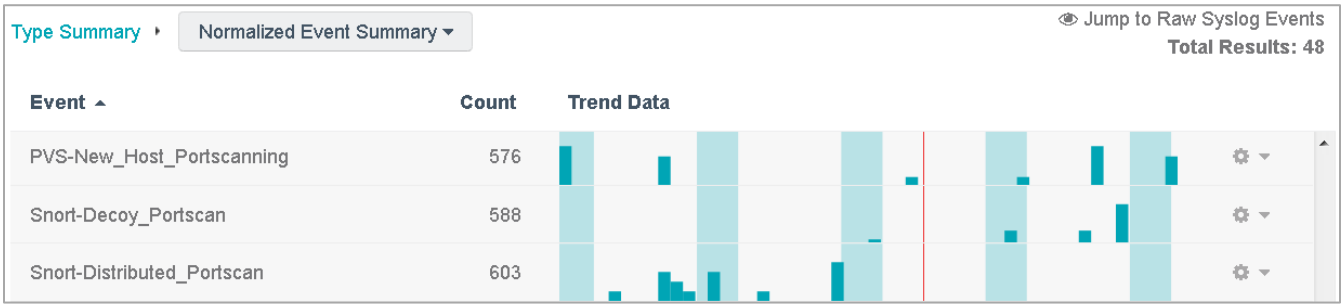
The Passive Vulnerability Scanner has an excellent detection engine to identify new systems that have suddenly become active on the network. The LCE has a built in correlation rule that detects any PVS-New_Host_Alert events followed by port scanning events associated with that new IP address.

Below is an example log:

```
PVS-New_Host_Portscanning - host ##.###.##.### (website.test.org) was recently
      discovered and is conducting portscans and most recently had a Snort-
      TCP_Portscan portscan event at 1/13/2012 14:38:18
```

The type of normalized port scan alert is indicated in the log.

Below is a screen capture of such a correlated event among other *scanning* events:



New hosts are hosts that the PVS was previously unaware of. They may have been on and off the network previously, but the PVS recognized that they had been removed. Hosts that are connected to the network but have been hibernating, disabled or otherwise not making any network traffic, will also become “discovered” by the PVS.

Worm Outbreaks

What is it?	Detects worm outbreaks through analysis of port scan traces.
What does it do?	The LCE tracks port scan event source addresses to see if they were previously scanned in another log.
Why does it matter?	The ability to detect a reflected port scan event is an accurate way to identify hosts on your network that have been compromised.
LCE Event Type	The LCE produces the Potential_Worm_Outbreak event and this is normalized as an *intrusion* event type.
Statistical Events	Since the produced event is an *intrusion* event, any anomalies associated with it will be correlated by the LCE’s statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Intrusion_Event log would be issued.
Continuous Events	The LCE would consider these events for continuous activity since it is part of the *intrusion* family.

A common virus and malware spreading pattern is to scan for new targets, infect them and then repeat the process from the new target and attempt to infect others. Purely considering port scan events, this type of infection can be detected anytime we see host A scan host B, and then at a later time, observe host B scan a new system, such as host C.

Below is an example log generated by the LCE:

```
Potential_Worm_Outbreak - host ###.###.###.### (sg-cogent.someplace.org) was
      recently scanned 1 time in the last hour and then scanned host
      ##.###.###.# (a88-114-249-2.elisa-laajakaista.fi) which could indicate a
      worm outbreak. The last detected scanning event was Snort-TCP_Portscan
      and occurred at 11/10/2011 16:5:5
```

This log entry indicates that the host at “someplace.org” was previously scanned, but now recently scanned another host with the detection of Snort TCP scan event. Within an LCE, a likely course of investigation would be to pick the source IP in this case and see which types of events are associated with it. It’s possible that all we have are the actual initial port scan event and then the outbound event. However, if you have access to system logs, firewall, intrusion and other types of events, you may be able to diagnose if this is a real attack or a potential false positive.

Since there are potential false positives with any type of network IDS port scan engine, there may be false positives with this type of correlation performed by the LCE. However, on large government and university networks we have monitored which have millions of port scan alerts on a given day, this type of correlation only occurred a few times each day.

When diagnosing potential infected hosts, consider the following types of patterns to look for:

- In system logs, any new processes, hung applications, crashes, errors, etc.
- Potential creation of new users, modified users, new services, new open ports or anything that indicates a change.
- Accompanying inbound and outbound intrusion events.
- Connections to IP addresses on the threatlist.
- Any type of *nbs*, *continuous* or *stats* events associated with scanning, network traffic, intrusions, etc.

Analysis of the events can be assisted by placing both the source and destination IP addresses of the worm outbreak event into a static asset list.

Successful and Unsuccessful Password Guessing

What is it?	Detects attempts to repeatedly guess a password and also identifies successful password guesses.
What does it do?	For any normalized *login-failure* event, the LCE tracks which IP addresses have acted as a client and induced multiple login failure records or logs for a given target system. The LCE also tracks if the IP address performing the password guessing then has a valid *login* event that could indicate a successful password guess.
Why does it matter?	Weak passwords may be easily guessable and the act of manually or automatically trying passwords several times can be easily detected.
LCE Event Type	The LCE produces the events Password_Guessing and Successful_Password_Guess, both of which are normalized to the *intrusion* event category.
Statistical Events	Since the produced event is an *intrusion* event, any anomalies associated with it will be correlated by the LCE's statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Intrusion_Event log would be issued.
Continuous Events	The LCE would consider these events for continuous activity since it is part of the *intrusion* family.

Password guessing is a very common form of security testing. Manually, a user can attempt to guess multiple passwords as quick as they can type. These passwords may come from a list of common passwords or even cracked passwords from other systems. There are also a variety of tools that can automate this form of guessing with a variety of parameters that dictate which characters to choose, how often to try a password, which root words should be used and so on.

Below is an example log that indicates brute force password guessing:

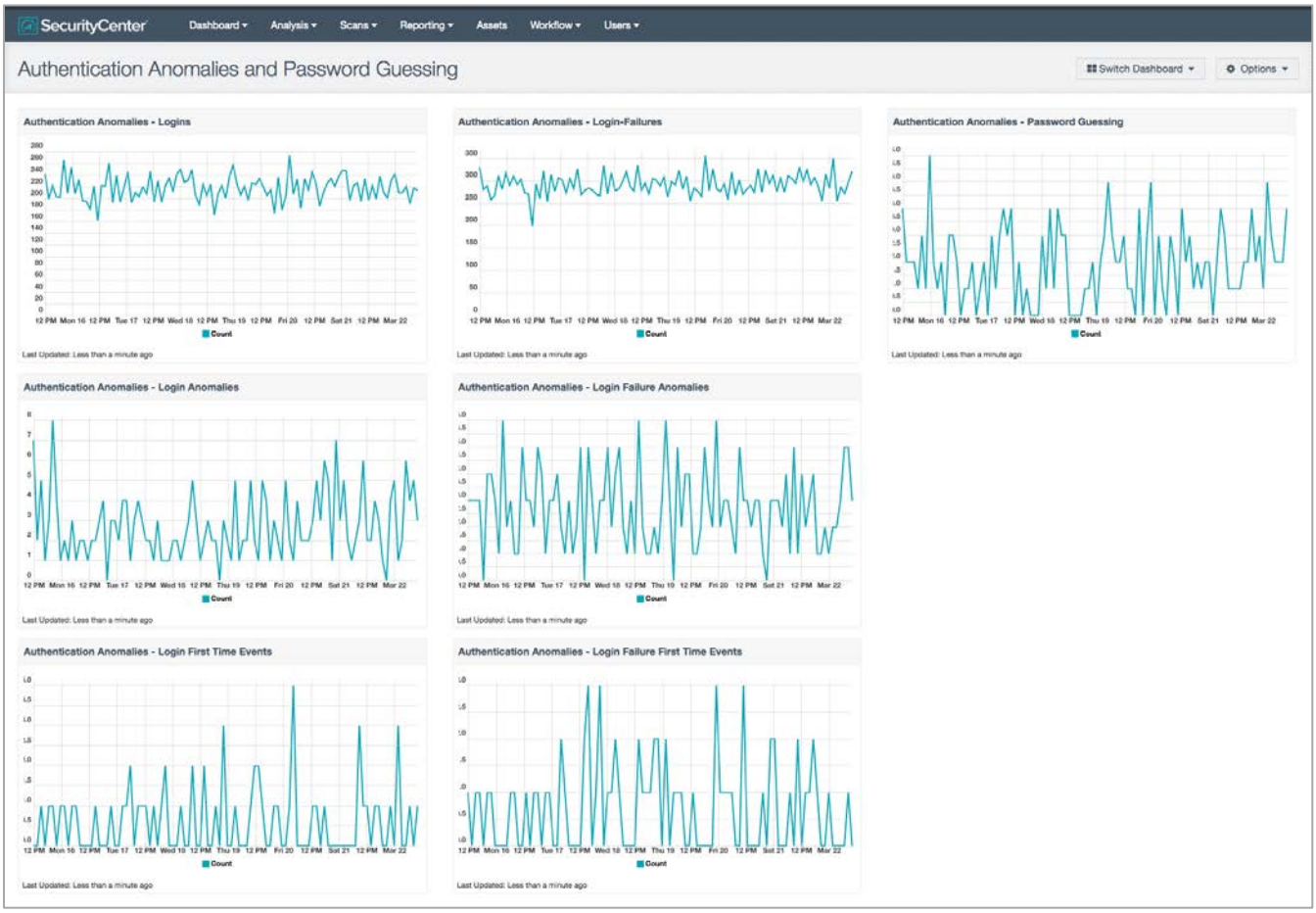
```
Password_Guessing - There have been 30 login failures in the last hour. The most
      recent login failure was from 172.20.100.100 (corpsc4.corp.company.com)
      to 172.20.100.101 (corplce.corp.company.com) with a login failure event
      of Linux-Audit_User_Login_Failed
```

The LCE is not concerned with the specific type of authentication log type. It is purely focused on counting the number of *login-failure* events between one IP address and its potential target. Since the LCE normalized hundreds of different types of authentication logs, any of these are candidates for analysis by the LCE's password guessing detection engine.

Legitimate forms of password guessing can come from many types of sources. These were previously discussed in the Determined Scan and Attack Detection section.

This normalized alert retains the source and destination IPs of the last login failure, which means the LCE's ability to filter, based on direction or asset, can be used to drive alerting, dashboards, reporting and analysis.

Tenable has published a dashboard that combines this form of correlation along with other types of login-failure events applied to the LCE's first time seen and statistical engines. The dashboard is named "Authentication Anomalies and Password Guessing" and is available here on the Tenable public dashboard site. A screen capture from the example dashboard is shown below:



An example successful password guess log is shown below:

```
Successful_Password_Guess - host 172.125.100.254 colo1.corp.company.com has
      logged onto 172.125.100.101 colo2.corp.company.com with a SSH-
      Accepted_Public_Key event. This event is suspicious, as the same source
      host failed to log into this destination host several times in the
      previous 15 minutes. This may indicated brute force password guessing.
```

This alert is also directional in nature, so filtering out events based on direction or asset is supported. If your network performs any type of automated discovery followed up by credentialed scanning, it is very likely that these logs will be generated for your vulnerability scanners or network management nodes. Most vulnerability scanners will cause some sort of authentication failure while probing services. If a credentialed patch audit is run immediately after a services scan, it is possible that the LCE will identify the scanner as a source of brute force password guessing and flag the valid logic as a successful password guess.

Suspicious Proxy Detection

What is it?	Detects “leapfrog” network connections that may have resulted from a compromised system being used to connect to other systems.
What does it do?	The LCE tracks the source and destination of all TCP connections as reported by the Tenable Network Monitor or Tenable NetFlow Monitor as well SSH, VNC and RDP sessions as reported by the Passive Vulnerability Scanner. The LCE identifies when a system receives a network connection from a client and then that system connects out to other systems.
Why does it matter?	It is common practice to compromise one host to achieve control of it and then use it as a beachhead to attack other hosts.
LCE Event Type	The LCE produces the Suspicious_Proxy, Suspicious_RDP_Proxy, Suspicious_SSH_Proxy or Suspicious_VNC_Proxy events that are normalized to the *network* event category.
Statistical Events	Since the produced events are of the *network* event type, any anomalies associated with it will be correlated by the LCE’s statistical event engine.
First Time Seen Events	If any of the above events had not previously been observed for a targeted host, an *nbs* Never_Before_Seen-Network_Event log would be issued.
Continuous Events	The LCE would not consider these events for continuous activity since it is part of the *network* family.
Threatlist	The LCE considers some *network* events for correlation with known IPs associated with botnets and will produce either an Outbound_Suspicious_Threatlist_Proxy or an Inbound_Suspicious_Threatlist_Proxy when a proxy to a hostile IP address occurs.

Many networks leverage proxy services to limit inbound, outbound and internal access to applications, networks and the Internet in general. A network proxy typically maintains a bridge between the client and the service it is connecting to and the bridge is composed of two separate TCP connections. Most proxies also attempt to understand the data and protocol being communicated and offer some type of security, performance enhancement, filtering or encryption services.

The concept of relaying network connections is also something leveraged indirectly by malicious software and attackers. Typically a system is compromised, the system is controlled, and then other systems are attacked and possibly compromised in the same manner. It is very likely that a control session, regardless of protocol, will stay open as an active TCP session while new outbound sessions are also in play.

The LCE’s correlation engine allows detection of these “unauthorized proxy” situations. Once they are detected, they can be reviewed along with other intrusion events or reported, alerted or “dashboarded” according to the assets being monitored.

There are two types of detected proxies. The first one is based purely on TCP connections sniffed by the Tenable Network Monitor (TNM) or gathered via netflow by the Tenable NetFlow Monitor (TFM). Below is an example log:

```
Suspicious_Proxy - host: 192.168.1.24 just completed a network session lasting
      multiple hours from client ##.##.##.## to port 22 and during that time
      completed the following TCP sessions (destIP:dport:time):
      192.168.1.21:22:many-hours
```

The LCE normalizes TNM and TFM logs based on TCP session length and bandwidth. The LCE's proxy detection logs track how many sessions may have come from an IP address that also had an active connection at the same time.

In the above log, the redacted IP address is the ultimate client. It connected to 192.168.1.24 on port 22 and during that time, 192.168.1.24 connected to 192.168.1.21 on port 22 and that session lasted for many hours. This is a classic example of an administrator "SSHing" into a server and then from there, "SSHing" to another system.

Servers that may make many types of TCP connections and also receive connections will likely cause alerts of this sort to be generated. For example, a Unix server that hosts FTP, IRC and email services and also supports SSH access will no doubt have many TCP connections active at any given time.

Below is an example log of this type that shows an Exchange server supporting many different SMTP, IMAP and POP connections to a secondary mail server:

```
Suspicious_Proxy - host: 172.20.210.11 just completed a network session lasting
      multiple hours from client 192.168.100.30 to port 25 and during that time
      completed the following TCP sessions (destIP:dport:time):
      192.168.99.133:110:many-days 192.168.99.133:143:many-days
      192.168.99.133:25:many-days 192.168.99.133:25:many-days
      192.168.99.133:25:many-hours 192.168.99.133:25:many-hours
      192.168.99.133:110:many-days 192.168.99.133:25:many-days
      192.168.99.133:25:many-hours
```

For each connection, the log attempts to summarize the length of the TCP session by giving it names such as many-hours, many-days, "15min", etc.

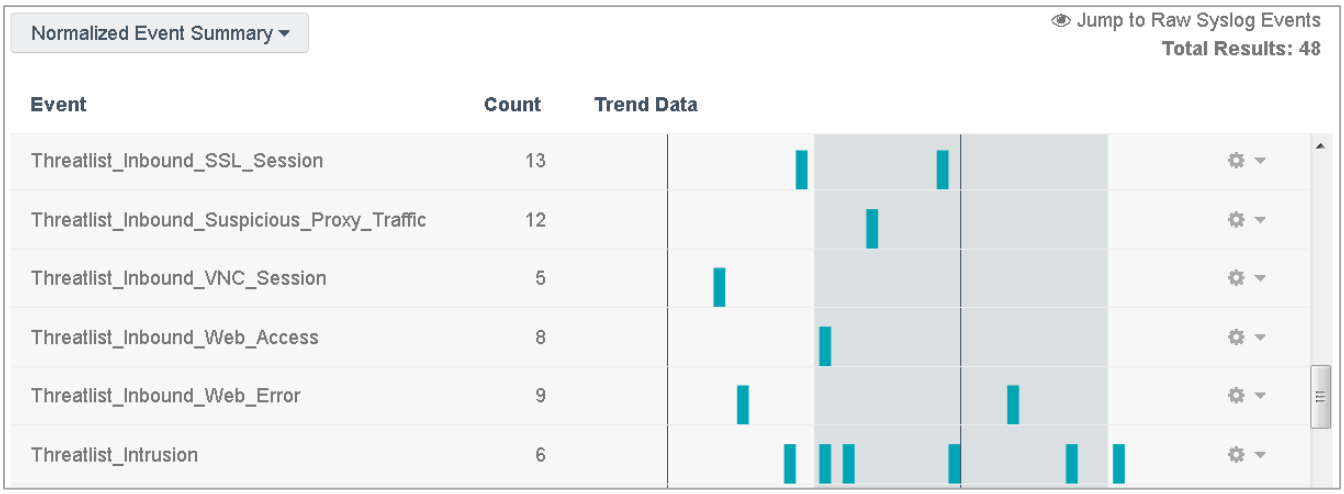
Below is another example log with sessions lasting fifteen minutes in length:

```
Suspicious_Proxy - host: ##.###.###.### just completed a network session lasting
      approximately 15 minutes from client ###.###.###.## to port 48595 and
      during that time completed the following TCP sessions
      (destIP:dport:time): ###.###.##.###:44532:15min ##.##.###.###:11173:15min
      ##.##.###.###:56094:15min ###.##.##.###:22309:45min
      ###.##.##.###:47201:15min ###.##.##.###:47201:15min
      ###.##.###.##:58266:15min ###.###.##.###:47201:15min
```

Many applications such a Voice over IP, Skype, online gaming, BitTorrent, video conferencing and some VPNs establish connections on many different ports. In the above log, both the inbound TCP port and all of the other TCP ports involved were on high "non-standard" ports.

The Suspicious_Proxy event is analyzed by the LCE's anomaly engines, so any spikes or deviations in network traffic will generate alerts. Additionally, the first time a server or IP address on the local network has one of these events, a *nbs* alert will also be generated.

In addition to this type of correlation, the LCE's *threatlist* correlation will also consider proxy events. Below is a screen capture of one day of *threatlist* events on a sample network. It has many inbound and outbound connections to potential botnets. The second row on this screen capture is a Threatlist_Inbound_Suspicious_Proxy_Traffic event.



This means that not only has a Suspicious_Proxy event occurred, one of the IP addresses involved with it is also on a list of known potential botnet servers.

Working with logs from the TNM and the TFM, it is tempting to call well known port traffic by their common name. However, not everything that travels on port 80 is HTTP, nor is everything running on port 22 necessarily a Secure Shell service.

To compensate for this, the real-time logs from the Passive Vulnerability Scanner are used to look for "instant" proxies based on detected SSH, VNC and RDP sessions. Below is an example log:

```
Suspicious_VNC_Proxy - host: 192.168.21.13 (godzilla.lab) has both received and
      initiated VNC sessions in the past 20 minutes the most recent towards
      192.168.21.16 (gigan.lab). The system could be a bastion host or being
      used to leapfrog and attack other systems.
```

In this case, the PVS detected a VNC session to a system and then a short time later, a new VNC session from that system.

The LCE supports detection of these instant proxies on the following protocols:

Protocol	Event Name	Port Independent	Description
RDP	Suspicious_RDP_Proxy	No – locked to RDP	Identifies a Windows "Terminal Services" session that launches a new session to a different computer.
SSH	Suspicious_SSH_Proxy	Any detected SSH session	Identifies the typical practice of SSHing into one host and then SSHing into another.
VNC	Suspicious_VNC_Proxy	Any detected VNC session	Identifies a VNC session that starts yet another VNC session to a new computer.

The LCE does not attempt to evaluate the potential maliciousness of the proxy. This is one of the reasons it is placed in the *network* event type category and not *intrusions*.

This type of network connection could be perpetrated by your IT staff right now and is normal practice on your network. However, this type of connection to servers may be a method you were unaware of. Since it can be detected, if malicious software attempts to connect through your network are made with these techniques, it's also something that can be found through alerting, reporting and dashboards.

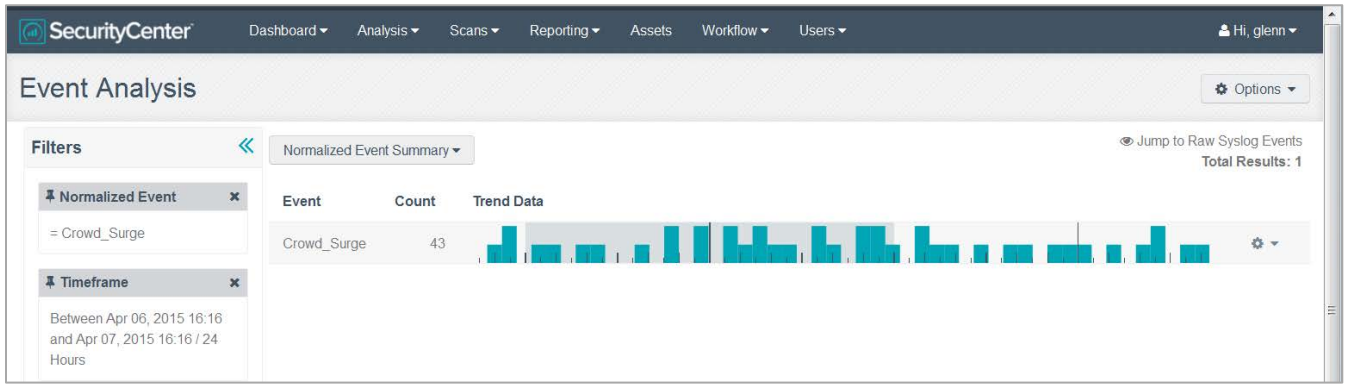
Questionable Outbound Connection Spikes

What is it?	The "Crowd_Surge" normalized event type will trigger when more than 20 outbound connection attempts to the same remote IP address occur.
What does it do?	The LCE uses "crowd_surge.tasl" to watch various connection types and network event logs. The Crowd Surge will generate a normalized event if a large number of hosts in a network connect to a single external IP address.
Why does it matter?	This could indicate spyware, malware, a worm or a botnet on a network reaching out to phone home to a command and control server.
LCE Event Type	Specific events that are associated with these correlations include any event types that start with TFM-TCP_Session, TNM-Long_TCP_Session, TNM-UDP_Activity network, TFM-UDP_Activity network, PVS-SSL and PVS_Web. Events are utilized by crowd_surge. tasl. PVS-Web_Query events are not considered.
Statistical Events	The LCE does not consider Statistical events for the "Crowd_Surge" normalized event type.
First Time Seen Events	The LCE does not consider never before seen events for the "Crowd_Surge" normalized event type.
Continuous Events	The LCE does not use continuous events types for the "Crowd_Surge" normalized event type.
Threatlist	The LCE does not use threatlist events types for the "Crowd_Surge" normalized event type.

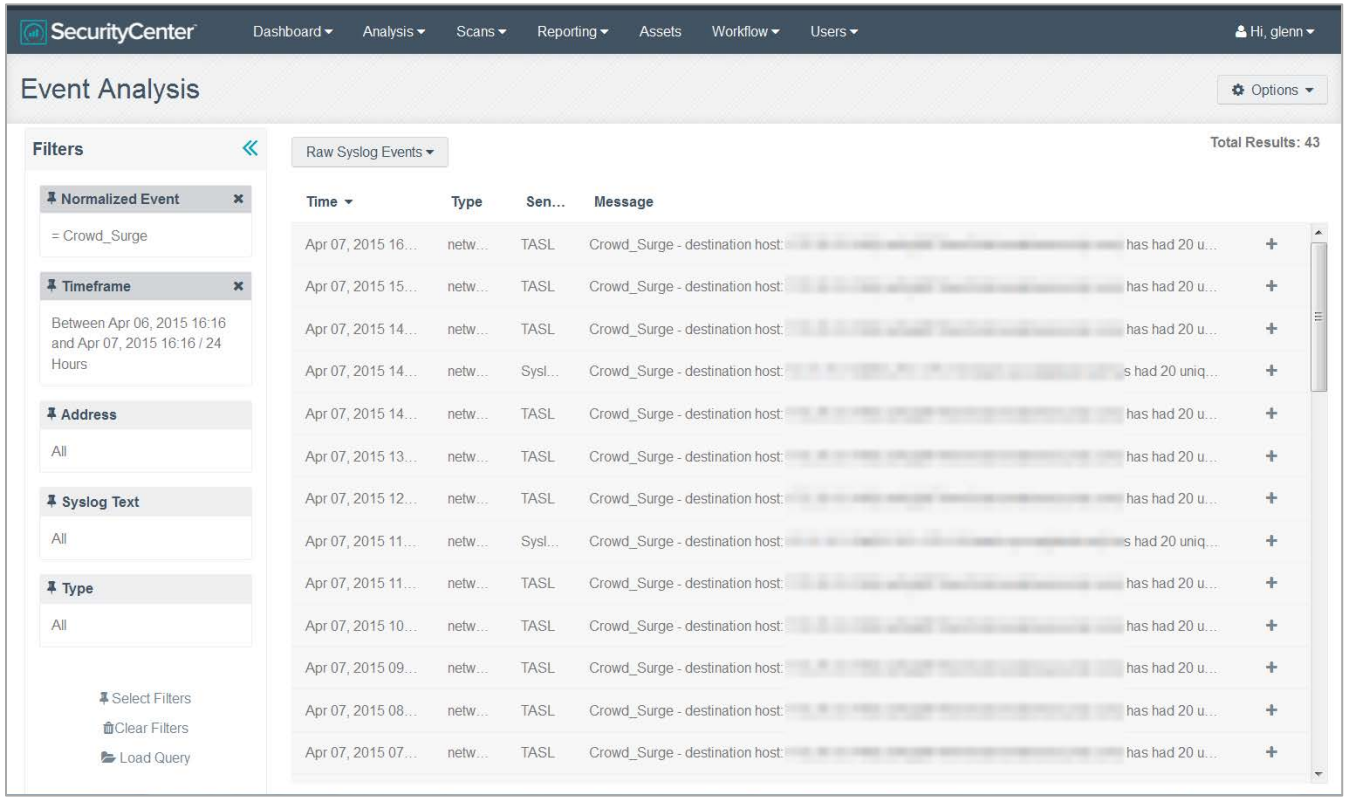
Crowd Surge watches various connection and network event logs and will alert if a large number of hosts in a network connect to a single external IP address. This kind of activity could indicate spyware, malware, a worm or a botnet on a network reaching out to phone home to a control server on a remote network.

Crowd Surge considers all events of the type "connection" sent by firewalls, switches, IDS/IPS and some OS types collected by syslog and LCE Clients. The event types collected by the Tenable Network Monitor and Tenable NetFlow Monitor clients are "TCP Session Network" and "UDP Activity Network". The difference between the clients is that the Tenable Network Monitor measures the length of time that sessions are active, while the Tenable NetFlow Monitor measures the amount of data gathered by the client.

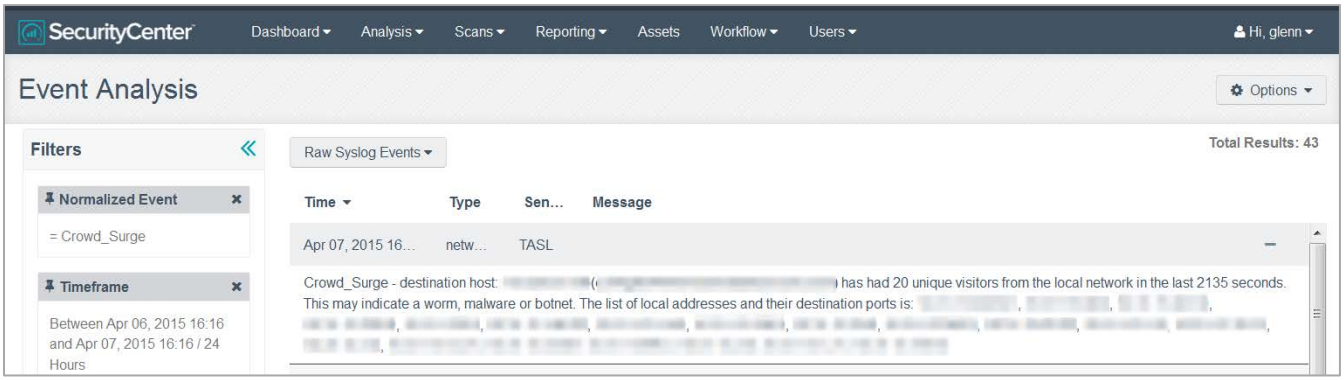
The Crowd Surge alert will triggers if 20 or more hosts in a network connect to uncommon ports on the same external destination, or 100 or more hosts connect to common ports on the same external destination address. The threshold is higher for common ports such as TCP 53 (DNS), 80 (http) and 443 (https). In the example below, it can be seen that 43 crowd surge events had occurred in the last 24 hour period where 20 or more hosts hit the same destination on the same port. The “Normalized Event” filter “=Crowd_Surge" is a quick way to find any of these in SecurityCenter CV.



Selecting “Raw Syslog Events” from the filter drop-down will show the raw syslog for each of the 43 events.



The full syslog can be reviewed by selecting the plus (+) symbol next to each event. The detail will include a list of local network addresses and the ports used to connect to the single remote IP address, which have been redacted in the example below.



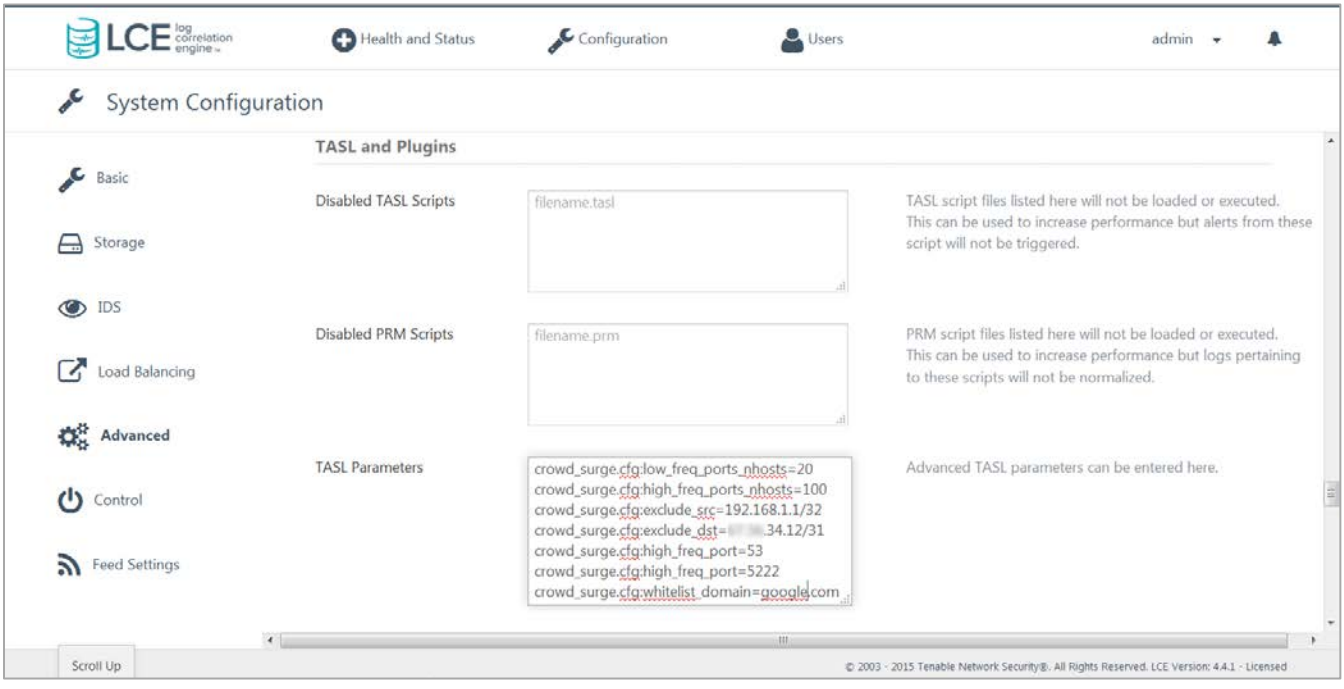
Crowd Surge currently operates on heuristics, mentioned above, but it can be tuned to an environment. The table below shows the options available to tune crowd_surge.tasl, which can be used to create a crowd surge configuration (crowdsurge.cfg):

Options	Description
low_freq_ports_nhosts	The minimum number of connections that need to occur on a port(s) where the outbound connection frequency to a single IP address would typically be minimal. After this threshold is met, a “Crowd_Surge” event will occur. The default value is 20 for this option. Example: low_freq_ports_nhosts=20
high_freq_ports_nhosts	The minimum number of connections that need to occur on a port(s) where outbound connections to a single IP address are typically frequent. After this threshold is met, a “Crowd_Surge” event will occur. The default value for this option is 100. Example: high_freq_ports_nhosts=100
exclude_src	A list of source IP addresses to be ignored, using an IP address (192.168.1.1), IP/CIDR (192.168.1.0/24), or IP/Netmask (192.168.1.0/255.255.255.0). Each “exclude_src” added requires a separate entry as shown in the example below. Example: exclude_src=192.168.1.0/24 exclude_src=192.168.10.0/24
exclude_dst	A list of destination IP addresses to be ignored using an IP address (192.168.1.1), IP/CIDR (192.168.1.0/24) or IP/Netmask (192.168.1.0/255.255.255.0). Each “exclude_dst” added requires a separate entry as shown in the example below. Example: exclude_dst= 203.0.113.0/24 exclude_dst= 198.51.100.0/24

high_freq_port	A list of ports that are utilized frequently. Each port added requires a separate entry as shown in the example below. Example: high_freq_port=53 high_freq_port=5222
whitelist_domain	A list of domains to be ignored due to frequent and expected connection to the same remote IP address. Each “white_list” domain requires a separate entry as shown in the example below. Example: whitelist_domain=google.com whitelist_domain=akamaitechnologies.com

To create a crowdsurge.cfg, the parameters can be entered into the LCE GUI. Log in to the LCE GUI by going to http://<ip address of the LCE server>:8836 in a web browser. Select “Configuration”, followed by “Advanced” and scroll down until the “TASL and Plugins” section is reached.

Each configuration value that is entered should be preceded with “crowd_surge.cfg:” as shown in the example below.



VI. Summary and Activity Reporting

The LCE performs a variety of log processing and summarizes and tags events in a manner that makes them easier to analyze for reporting, alerting and incident response. In many cases, the summarization of data provides additional situational awareness in a manner that is easy to comprehend.

User IP Address Correlation

The LCE includes the ability to keep track of which IP address a user ID is using and then apply this user ID to other events. For example, consider the following login event from an IMAP server:

```
Jul 30 11:31:01 mail1 imaps[19785]: login: cpe-###-##-###-###.example.com
       [###.##.###.###] joeuser plaintext+TLS User logged in
```

The user “joeuser” is coming from the redacted IP address above. Now consider this following SSH login failure event:

```
Jul 30 12:55:07 org1 sshd(pam_unix)[10129]: 1 more authentication failure;
       logname= uid=0 euid=0 tty=ssh ruser= rhost=###.##.###.### user=root
```

The login attempt was for the root user, but the source IP address above is the same IP address that “joeuser” just logged into the IMAP server with.

If you knew this relationship, an LCE user could manually type in the IP address of each user they wanted to keep track of, however, this doesn’t scale, especially with thousands of users. The good news is that the LCE can do this for you automatically across any normalized event.

To configure this in the LCE, you only need to do three things:

- Decide which login events you want to use for your IP addresses pivot.
- Find the LCE plugin ID for those events and place them into the *trusted_plugins.txt* file located in **/opt/lce/admin**.
- Restart the LCE.

All potential LCE plugin IDs that can be used for user ID tracking are listed at the end of the *prm_map.prm* file that is updated and stored in the **/opt/lce/daemons/plugins** directory.

There are many types of authentication logs that your organization may have in use. There could be independent audit trails for VPN users, domain authentication or email authentication. These resources could all point to the same LDAP authentication place.

Regardless of which authentication you are using, you should consider the IP addresses of what is involved and make sure there is relevant overlap. Many Tenable customers tend to use email authentication as central tracking log because it catches remote mobile users, users logging in from their home, VPN users and users working centrally in an office. Other Tenable customers make use of Windows domain logins that associates a user ID to each node participating in the network.

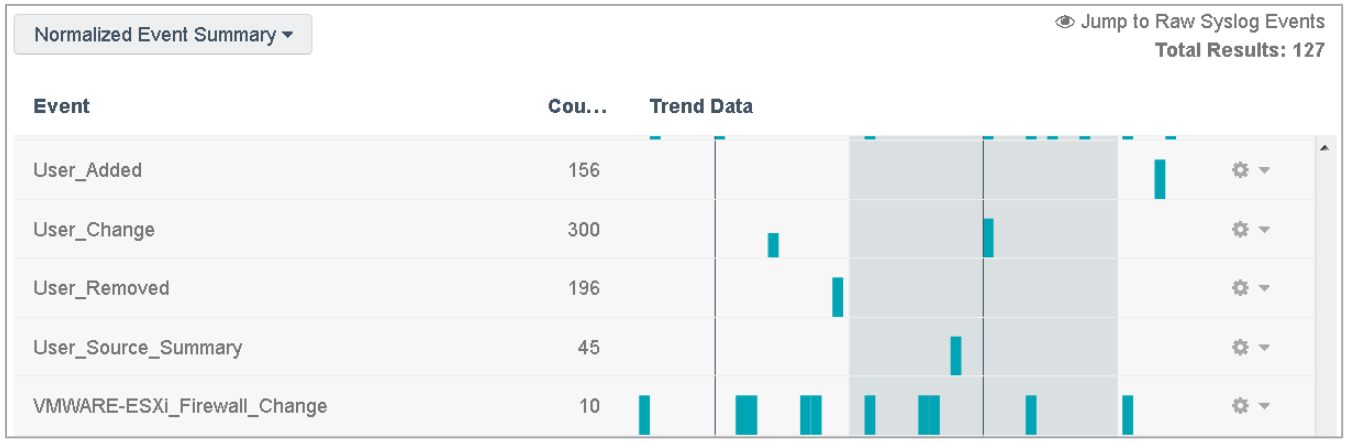
In some cases, log correlation does not make sense. For example, consider a network that has extensive NetFlow and network IDS coverage of their DMZ, but all user access originates from an RFC1918 network through a single Network Address Translation (NAT) firewall. In this case, all outbound network traffic from these users would have the external NAT IP address and this would not be relevant to the Internet IP address of the user.

The LCE will track users as they log in from new IP addresses. An event named “user-IP-change” is generated when a login occurs. Below is an example log from such a transition:

```
Network user IP address change: user joeuser became active at 172.20.101.203
       with event IMAP-User_Login2 (172.20.101.203:0 -> 172.20.210.11:0).
```

In this case, 172.20.210.11 was the IP address of the IMAP server. If your users have multiple devices that authenticate, it isn't unlikely that each IP address will be associated with the user. Also, if a user is migrating through your network, such as moving on a wireless device and getting new DHCP addresses, you may see many types of these logs for campus users.

Below is a screen capture of several *detected-change* events:

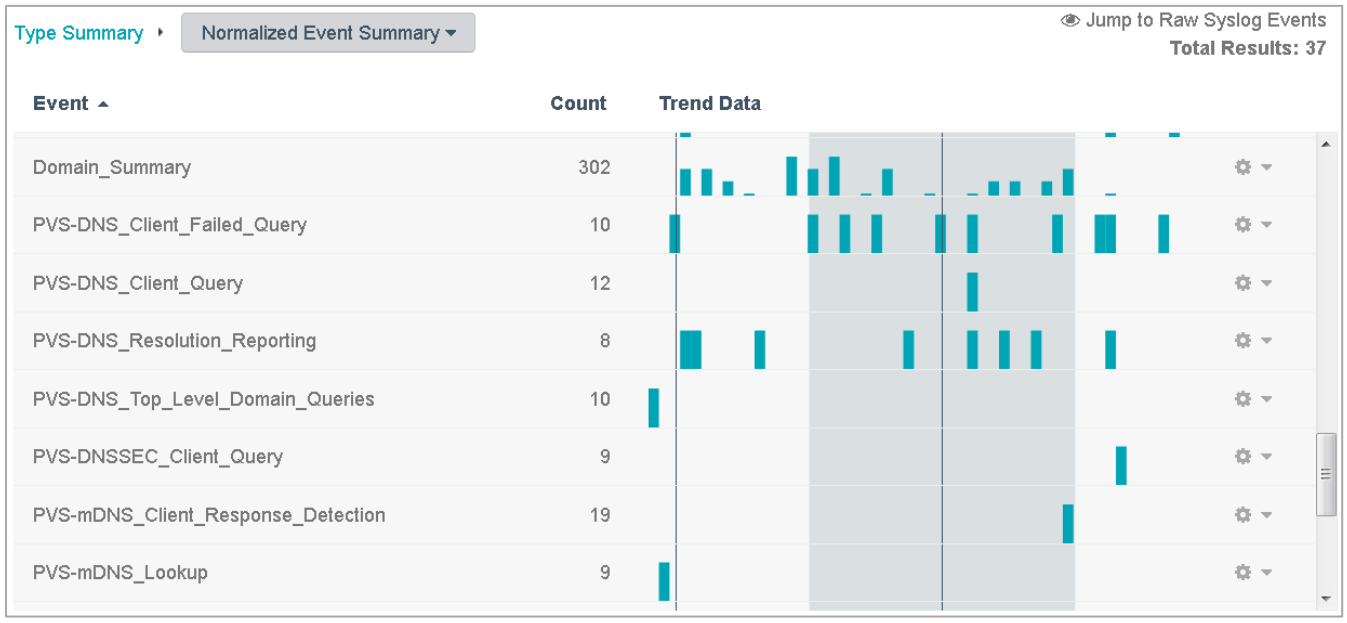


Domain Query and SSL Server Certificate Summary Reporting

The LCE attempts to summarize DNS queries and SSL target domain certificates for each IP address on the network. DNS query logs can be supplied by the PVS, BIND or domain controllers. Logs of SSL network sessions are provided by the PVS.

The LCE will produce a Domain_Summary event that contains a summary of all logs per hosts. It attempts to produce a report for each host every six hours, but will also produce a report if a host is fairly busy performing queries and will attempt to summarize domains queried in near real time if needed.

Below is a screen capture of *dns* event types from a small network monitored by the PVS:



The entirety of the 12 DNS client query events were summaries in 302 Domain_Summary events. An example log for such an event is shown below:

```
Domain_Summary since 1/18/2012 21:21:51 host 192.168.230.90
      (list.dmz.company.com) queried these domains:
      company.com.dob.sibl.support-intelligence.net
      company.com.dbl.spamhaus.org list.company.com.rhsbl.ahbl.org
      list.company.com 163mx00.mxmail.netease.com
      fidelity.co.in.s200a1.psmtp.com trentu.ca mx2.trentu.ca mx3.trentu.ca
      mx1.trentu.ca rps3624.ovh.net nate.com 113-197-50-93.reverse.ntc.net.pk
      113-197-50-93.reverse.ntc.net.pk.dmz.company.com kernel.org
      ks351762.kimsufi.com host-static-109-185-29-116.moldtelecom.md Static-
      52.147.195.14.tataidc.co.in Static-
      52.147.195.14.tataidc.co.in.dmz.company.com emailinator.com
      mailinator.com mail.digitalsanctuary.com customer-SLRC-113-
      162.megared.net.mx customer-SLRC-113-162.megared.net.mx.dmz.company.com
      mx2.comcast.net mx1.comcast.net mx2.mailhostbox.com qq.com
      ns6.company.com list.company.com.fulldom.rfc-ignorant.org
      novell.com.multi.surbl.org
```

These logs are very useful to help understand what type of DNS browsing has been performed by a given host.

In a similar manner, the LCE will attempt to summarize observed destination SSL certificate organizations logged by the PVS and summarize them. Tracking destination SSL certificates per host can provide more detail as to the type of secure web sites and applications that are leveraged by the IP address. Below is a screen capture of a *network* event type session summary.

There are 2705 SSL_Cert_Summary events that report logs such as the following:

```
SSL_Cert_Summary since 1/18/2012 18:17:38 host 192.168.1.59 (test-iPod.lab) had
      SSL sessions involving these server certs: Akamai Technologies Inc Google
      Inc. Thawte Consulting (Pty) Ltd GeoTrust
```

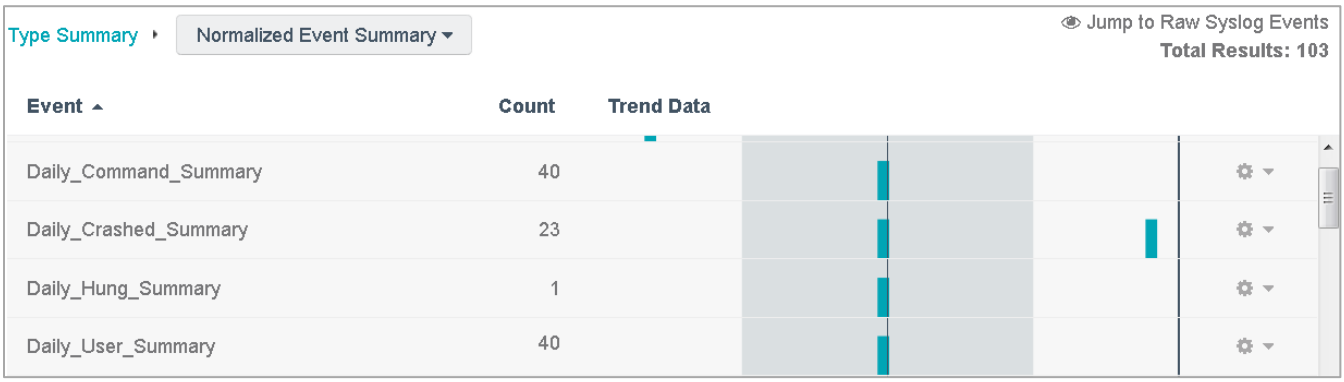
Knowledge of SSL certificates in use can also shed light on the use of a variety of web based services and Internet secure applications.

Process Executable Summary Reporting

The LCE tracks process execution via the LCE Clients on Unix systems, the Unix audit trail and Windows event logs. For each process that runs or crashes, the LCE will attempt to summarize processes names that have run hourly per system, daily per system and which users were active that day.

This level of reporting facilitates understanding which programs are running on a system and also supports the Execution Profiling and Change Detection methods of correlation.

Below is a screen capture of a variety of *process* event types:



There are 40 servers that are being monitored because there are 40 Daily_Command_Summary and Daily_User_Summary events within the 24 hour window of this chart. Below are example logs for these alerts, as well as the similar hourly alerts:

```
Daily_Command_Summary - host 172.30.2.66 (mini08-xp.someplace.org) issued these
      commands in the last day: javaw.exe, wmiprvse.exe, javaws.exe,
      searchfilterhost.exe (report generated at 1/19/2012 00:00:07)
Hourly_Command_Summary - host 172.30.2.66 (mini08-xp.someplace.org) issued these
      commands in the last hour: javaws.exe, Skype.exe, searchprotocolhost.exe,
      javaw.exe, searchfilterhost.exe, iexplore.exe, wmiprvse.exe, userinit.exe
      (report generated at 1/18/2012 15:00:05)
Hourly_Crash_Summary - host 172.30.2.6 (mini-01-xp.someplace.org) issued these
      commands in the last hour: skype.exe, audacity.exe (report generated at
      1/19/2012 08:40:01)
```

It is very convenient to have these summaries handy during an incident response situation or even when trying to determine what a system was running before a crash.

VII. About Tenable Network Security

Tenable Network Security transforms security technology for the business needs of tomorrow through comprehensive solutions that provide continuous visibility and critical context, enabling decisive actions to protect your organization. Tenable eliminates blind spots, prioritizes threats, and reduces exposure and loss. With more than one million users and more than 20,000 enterprise customers worldwide, organizations trust Tenable for proven security innovation. Tenable's customers range from Fortune Global 500 companies, to the global public sector, to mid-sized enterprises in all sectors, including finance, government, healthcare, higher education, retail and energy. Transform security with Tenable, the creators of Nessus and leaders in continuous monitoring, by visiting tenable.com.