Integration Guide

Micro Focus Common Event Format Integration Guide

Polyverse Corporation

ZeroTect

Date: December 14, 2020

ArcSight Tested

Contents

ArcSight Integration Guide

This document is provided for informational purposes only, and the information herein is subject to change without notice. Please report any errors herein to Micro Focus. Micro Focus does not provide any warranties covering this information and specifically disclaims any liability in connection with this document.

Certified Integration:

The integration complies with the requirements of the Micro Focus Technology Alliance Partner program. For inbound integrations, the Micro Focus ArcSight CEF connector will be able to process the events correctly and the events will be available for use within Micro Focus' ArcSight product. In addition, the event content has been deemed to be in accordance with standard SmartConnector requirements. For action and outbound integrations, the integration establishes outbound communications from Micro Focus ArcSight to a third party platform. The integration has been tested and demonstrated to Micro Focus by the third party.

Revision History

Date	Description
11/15/2020	First edition of this Configuration Guide.
12/14/2020	Version 0.4 Certified by Micro Focus

Polyverse Integration Guide

This guide provides information for configuring the Polyverse Zerotect integration for ArcSight ESM. This integration is supported on ESM versions [7.0] and later. Polyverse Zerotect version(s) 0.4.11 and above are supported.

Joint Solution Overview

Zerotect is an open-source agent that, integrated with Micro Focus ArcSight, detects attempted zero-day attacks with no prior knowledge of the vulnerability or attack being used.

Zerotect observes various system events (such as segmentation faults, core dumps, application crashes, etc.) and interprets them in a structured format and emits them to ArcSight for analysis. ArcSight's powerful data analysis capabilities then enable detection of a live zero-day attack by correlating elements of Zerotect events.

Memory-based fileless attacks usually take a few attempts to get right. When some of these attempts fail, they produce side-effects in the form of crashes and faults. These "faults" logs are not structured and are usually split across multiple entries in the kernel log buffer. Zerotect reinterprets these unstructured and separate events into one unified entry in the Common Event Format (CEF).

Once ArcSight consumes these events, they reveal interesting patterns that can confirm a live zero-day attack in progress. ArcSight provides the analytics, graphing and monitoring tool at scale so that security teams are able to further drill deeper into what processes were attacked, what pattern the attacker is following and so forth.

ArcSight and Zerotect enable a new class of zero-day detection by enterprises and SOCs improving their security posture and situational awareness.

Use Cases

This section describes important use cases supported by this integration.

- Fileless memory-based attack detection
- General crash analysis and remediation
- New vulnerability (CVE0 disclosure

Fileless memory-based attack detection

A fileless memory-based attack is a specific kind of cyber-attack wherein the attacker does not modify anything in the filesystem and operates purely in memory. Due to this, traditional anti-malware solutions that rely on detecting changes to a file system are of limited use in detecting these attacks.

Zerotect overcomes this by looking for the symptoms or side-effects of such an attack being conducted, rather than looking for the attack itself. Due to having to operate purely within memory, memory-based attacks are very fragile and require some trail-and-error before they succeed. Any time such an attack fails, it produces various system "faults" (attempting to execute illegal instructions, accessing memory in a manner not permitted, etc.)

Using ArcSight to analyse and organize such faults in a centralized location, it is possible to detect various patterns that indicate a zero-day attack in progress.

General crash analysis and upstream remediation

One of the main benefits of Zerotect, is whether it detects a credible attack or not, everything it does event on is generally a bad thing. “Faults” need to be caught and fixed for proper system function.

Thus, all events generated by zerotect, whether cyber-attacks or not, are useful and actionable, and credibly improve the posture of the system it runs on.

Since Linux does not have a centralized vendor where widespread program and system crashes are reported back to, Zerotect enables an organization to identify and raise bugs or fix issues in upstream software.

New vulnerability (CVE) disclosure

Since Zerotect can detect zero-day attacks that have never been seen before, it can lead Security Research teams and organizations to raise CVEs and perform responsible vulnerability disclosure.

CEF Integration

1. Configuration of Zerotect to output CEF events

Zerotect provides the ability to send events to a syslog endpoint in the CEF format. There are two primary ways to configure it, either using the command-line or using a configuration file.

Command-line parameters

The following command-line parameters are supported for configuring Zerotect with syslog.

CLI flag	Usage
--syslog	Sends all monitored data to syslog in the specified format. Unless a destination is selected, tries to send to standard syslog destinations when in order of unix socket, tcp and udp. Since the UDP destination will almost never fail, if there is no listener, logs will be lost. [possible values: text, json, cef]
--syslog-destination	The syslog destination type. If a destination is selected, the destination configuration flags are explicitly required and defaults are not used. [possible values: unix, tcp, udp]
-- hostname	Provide an explicit hostname for events generated from this zerotect instance. By default host name is picked up from /etc/hostname or /proc/sys/kernel/hostname in that order.
--syslog-server	The syslog tcp or udp server addr to send syslog events to to (when the destination is TCP or UDP.) (usually ip:port)

--syslog-local	The unix socket to send to (when the destination is UNIX.) (usually /dev/log or /var/run/syslog)
--syslog-unix-socket-path	The unix socket to send to. (usually /dev/log or /var/run/syslog)

To send zerotect events to ArcSight's syslog UDP endpoint, this is an example command:

```
zerotect --syslog cef --syslog-destination udp --syslog-local 172.17.0.4:57945 --syslog-server 3.134.109.171:1514
```

Configuration file

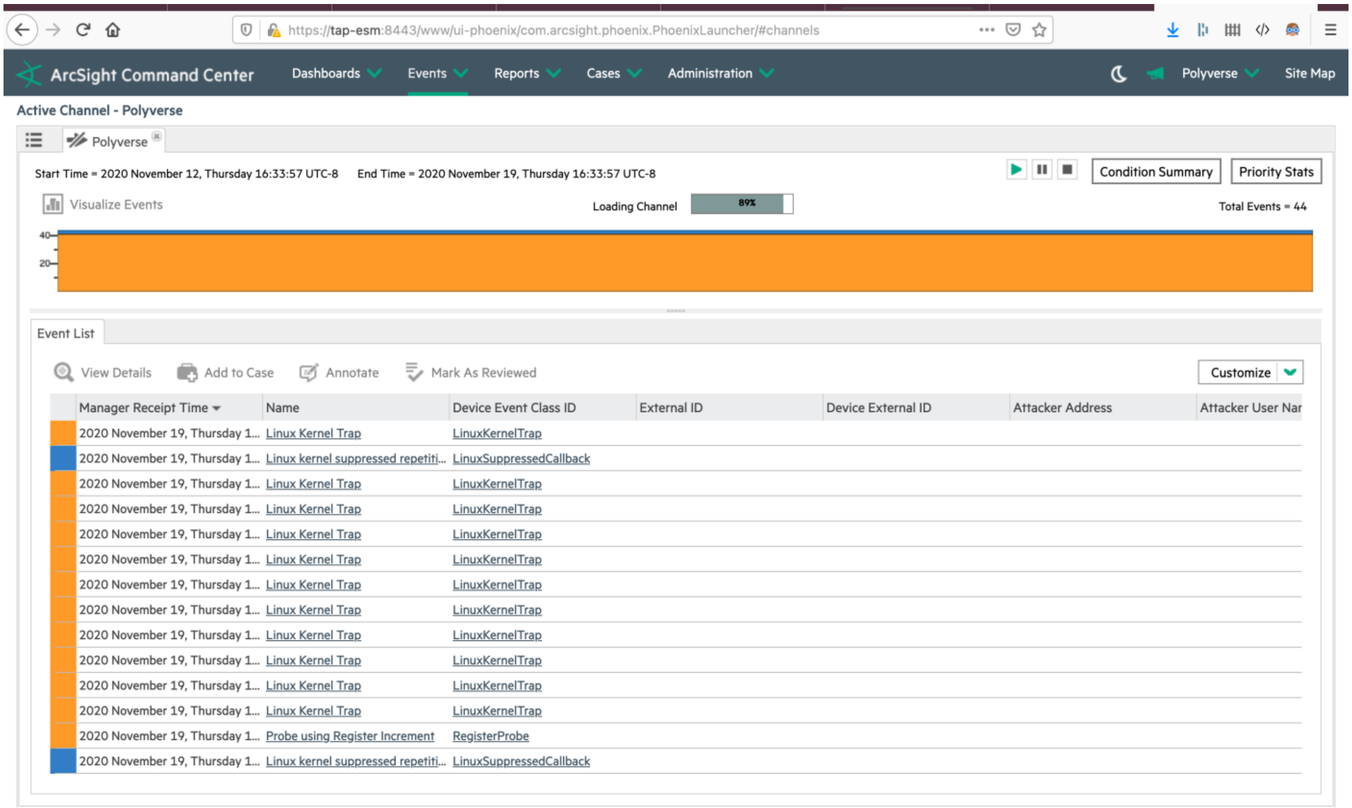
All zerotect CLI options are available through a configuration file. To enable this, zerotect must be launched with a single command-line parameter that provides it the configuration file to use, for example:

```
zerotect -configfile /etc/zerotect/zerotect.toml
```

An example configuration file that sends CEF-formatted events to ArcSight's syslog UDP endpoint is:

```
[syslog]
format = 'CEF'
destination = 'Udp'
server = '3.134.109.171:1514'
local = '172.17.0.4:57945'
```

These events will show up in ArcSight thus:



Active Channel - Polyverse

Start Time = 2020 November 12, Thursday 16:33:57 UTC-8   End Time = 2020 November 19, Thursday 16:33:57 UTC-8

Loading Channel 89%

Total Events = 44

	Manager Receipt Time	Name	Device Event Class ID	External ID	Device External ID	Attacker Address	Attacker User Nar
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟦	2020 November 19, Thursday 1...	Linux kernel suppressed repetiti...	LinuxSuppressedCallback				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Linux Kernel Trap	LinuxKernelTrap				
🟧	2020 November 19, Thursday 1...	Probe using Register Increment	RegisterProbe				
🟦	2020 November 19, Thursday 1...	Linux kernel suppressed repetiti...	LinuxSuppressedCallback				

2. Events

Zerotect generates 5 main types of events.

Event ID	Event Description
RegisterProbe	A Register Probe event is a conclusive localized detection of an attack where a Register is being probed in successive failed attempts. Basically if there are multiple events where a register such as “IP” (Instruction Pointer) is incremented by one byte at a time, that is a conclusive detection of an attacker searching for valid addresses using a brute-force probe. Example Event: *CEF:0|polyverse|zerotect|1.0|RegisterProbe|Probe using Register Increment|10|cn1=0 cn1Label=justifying_event_count cs1=RIP cs1Label=register dhost=hostnamecef dproc=nginx msg=Instruction pointer rt=471804323*
LinuxKernelTrap	A LinuxKernelTrap is a parsed and structred representation of a Linux Kernel “Trap” logged into the kernel log buffer. Example Event: *CEF:0|polyverse|zerotect|1.0|LinuxKernelTrap|Linux Kernel Trap|10|PolyverseZerotectInstructionPointerValue=0 PolyverseZerotectStackPointerValue=140726083244224 cn2=94677333766144 cn2Label=vmastart cn3=4096 cn3Label=vmasize cs2=Read cs2Label=access_type cs3=User cs3Label=access_mode cs4=false cs4Label=use_of_reserved_bit cs5=false cs5Label=instruction_fetch cs6=false cs6Label=protection_keys_block_access dhost=hostnamecef dpid=36275 dproc=a.out flexString2=Segfault at location 0 flexString2Label=signal fname=a.out reason=NoPageFound rt=471804323*
LinuxFatalSignal	A LinuxFatalSignal is a parsed and structred representation of a Linux Fatal Signal logged into the kernel log buffer. Example Event: *CEF:0|polyverse|zerotect|1.0|LinuxFatalSignal|Linux Fatal Signal|10|flexString2=SIGSEGV flexString2Label=signal rt=471804323*
LinuxSuppressedCallback	A LinuxSuppressedCallback is a parsed and structred representation of an event where the kernel suppresses multiple log entries when they happen rapidly and instead emits the function that was suppressed and a count of how many times it was suppressed. Under heavy attack, this event is crucial to capturing the complete signal. Example Event: *CEF:0|polyverse|zerotect|1.0|LinuxSuppressedCallback|Linux kernel suppressed repetitive log entries|3|cnt=9 dhost=hostnamecef*

	flexString1=show_signal_msg flexString1Label=function_name rt=471804323
ConfigMismatch	Zerotect has the ability to configure the system for optimal monitoring and to continually ensure that those settings stay intact. When a configuration that Zerotect was asked to ensure has a specific value, is then found to have a mismatching value, this event is generated. It may be a strong indicator that an attacker is attempting to suppress certain events from ever being generated. Example Event: *CEF:0|polyverse|zerotect|1.0|ConfigMismatch|Configuration mismatched what zerotect expected|4|PolyverseZerotectExpectedValue=Y PolyverseZerotectKey=/sys/module/printk/parameters/time PolyverseZerotectObservedValue=N dhost=hostnamecef rt=471804323*

3. Device Event Mapping to ArcSight Data Fields

Information contained within vendor-specific event definitions is sent to the ArcSight SmartConnector, then mapped to an ArcSight data field.
The following table lists the mappings from ArcSight data fields to the supported vendor-specific event definitions.

Zerotect Connector Field Mappings.

ArcSight Event Data Field	Polyverse Use-case
msg	An event description/message with additional explanation of what triggered it.
cs1,cs1Label = register	A machine register (i.e. ax, bx, etc.) that the event is about.
dproc	The process name (i.e. 'nginx', 'apache', etc.) the event is about.
cn1,cn1Label=justifying_event_count	When a detection event occurs, the number of raw events that justify the overall detection. For example, a live attack detection might result from 10 segmentation faults. The justifying event count in this case is 10.
dpid	The process id that the event is about.

PolyverseZerotectInstructionPointerValue	The value of the Instruction Pointer (IP) register. This is where the attacked process was executing code when it faulted.
PolyverseZerotectStackPointerValue	The value of the Stack Pointer (SP) register. This is where the attacked process stack frame was when it faulted.
reason	The reason for a Fault event
cs2, cs2Label=access_type	The type of Access that caused the fault (such as Read access or Write access).
cs3,cs3Label=access_mode	The mode of access that caused the fault (such as Kernel mode or User mode).
cs4,cs4=use_of_reserved_bit	Boolean. Whether use of reserved bits in the page table entry caused the segfault
cs5,cs5=instruction_fetch	Boolean. Whether an instruction-fetch (vs data read/write) caused the fault
cs6,cs6Label=protected_keys_block_access	Boolean. Whether memory protection keys block was accessed
Fname	File in which the Fault occurred. For example, a Fault might occur in glibc.so. This tells us specifically which shared library or file it occurred in, under the overall process.
cn2,cn2Label=vmastart	For the field Fname above, Virtual Memory Address start (where this file was placed by ASLR/OS.) If glibc.so was loaded at address 0x40000 through ASLR, then this value would indicate that.
cn3,cn3Label=vmasize	Virtual Memory size of the file's mapping. For example, if glibc.so was 3 megabytes big, this is where the size would be indicated.
flexString1, flexString1Label=signal	Fatal Signal Type (Fault, Trap, etc.)
PolyverseZerotectStackDump	The set of register values and any other arbitrary key-value pairs dumped by the kernel on a fatal signal

flexString2, flexString2Label=function_name	For a SuppressedEventCallback, the name of the function in the kernel, whose errors were suppressed/throttled
Cnt	For a SuppressedEventCallback, the number of times the function error was suppressed.
PolyverseZerotectKey	Zerotect can be configured to set and ensure system configurations are a certain value (such as logging of fatal signals, or exception traces.) Sometimes after Zerotect has set these to the desired values, the may be changed and mismatch. When configuration mismatches occur, this is the config key which was found to be mismatching
PolyverseZerotectExpectedValue	The value that was expected in the mismatching key
PolyverseZerotectObservedValue	The value that was observed for the mismatching key
dhost	The hostname from where the events are generated. Applies to all event types.

ArcSight Content for Zerotect

Prerequisites

Product Name	Version Information	Operating System
Micro Focus ArcSight	7.0	Windows

Support

Integration support information when an issue is outside of the ArcSight team's scope

In some cases the ArcSight customer service team is unable to help with issues that lie within the configuration itself in which case the certified vendor should be contacted for assistance:

Polyverse Customer Support
Phone - +1 855 765 9837

Email - support@polyverse.com
Instructions – Determine whether you have a Zerotect problem or a general ArcSight support request. For all zerotect support requests, we recommend emailing support@polyverse.com as a first step. Polyverse will assign a support engineer and handle the engagement.

Additional ArcSight Documentation

For more information about the joint solution, visit the Micro Focus ArcSight Marketplace:
https://marketplace.microfocus.com/arcsight/category/partner-integrations
For more information about Micro Focus Security ArcSight ESM:
https://software.microfocus.com/en-us/software/siem-security-information-event-management