WEBVTT

00:01.200 --> 00:05.600
We want to recap the architectural components
by examining a captured event.

00:06.640 --> 00:11.680
This starts at the time when the events arrive
at their first collection point, the Event Collector.

00:12.880 --> 00:19.560
We follow the events as they proceed through correlation,
accumulation, and storage on the Event Processor

00:19.920 --> 00:23.480
and finally end up as part of a larger event on the Console.

00:25.600 --> 00:28.240
Let's begin with what happens in an Event Collector.

00:28.960 --> 00:32.840
In this scenario, we follow a stack of Checkpoint Firewall deny events.

00:33.640 --> 00:40.360
The firewall denies many communication requests from an individual
IP source and logs those request denials.

00:41.360 --> 00:45.200
These FW Deny events arrive at the QRadar Event Collector.

00:46.200 --> 00:53.240
The overflow filter counts all the incoming raw events
to ensure that the license limit for the appliance is not exceeded.

00:54.120 --> 01:01.680
If the license limit is exceeded, the events are buffered and fed
back into the stream when the input drops below the license limit.

01:02.120 --> 01:08.520
If the buffer is already full, the new events are dropped
and a special event for the Console is generated.

01:09.280 --> 01:13.600
The Collector enforces
licensing limits that it inherits from the Processor.

01:14.360 --> 01:20.720
If your deployment uses Processors to collect events and flows,
then licensing is enforced on the Processor.

01:21.320 --> 01:27.800
In our case, the limit is not exceeded
and the FW Deny events are passed on to the Traffic Analysis module.

01:28.880 --> 01:33.080
The Traffic Analysis Module performs the auto discovery of log sources.

01:34.280 --> 01:41.640
If the log source is already known (like in our case: Firewall),
the records are handed over to the appropriate DSM module.

01:42.440 --> 01:47.760
If the log source is not known yet
but is recognized, a new log source is generated.

01:47.800 --> 01:51.400
Then, the event is handed over to the appropriate DSM module.

01:52.280 --> 01:56.440
If the event cannot be attributed to either a known or a new log source,

01:56.760 --> 02:00.720
the event is stored as "unknown" and listed as such on the Console.

02:02.000 --> 02:08.080
The individual FW deny events are now parsed inside the applicable
(Firewall) Device Support Module.

02:08.520 --> 02:15.920
The event ID is extracted from the event data,
and a QID (QRadar identifier) is assigned to the event.

02:16.640 --> 02:25.240
This QID is used later in the CRE (Custom Rules Engine) to evaluate and
correlate our events together with other events and flows.

02:26.160 --> 02:29.360
All events are then passed through the coalescing filter.

02:29.880 --> 02:33.720
An unidentified event has a "stored" low level category.

02:34.920 --> 02:42.080
Here, duplicate events (examined within 10 second intervals)
are combined into one event with a counter,

02:42.320 --> 02:48.840
which helps to reduce storage space and processing capability
when data is handed to the Event Processor.

02:50.080 --> 02:56.080
In our case, many FW Deny events are coalesced
because they occurred within 10 second intervals.

02:56.760 --> 03:04.200
These normalized FW Deny events (with QID) are now ready
to be sent off to the Event Processor for further processing.

03:07.640 --> 03:10.680
In the Event Processor, just as in the Event Collector,

03:11.000 --> 03:17.200
the overflow filter counts the incoming normalized events
to ensure that the license limit for the appliance is not exceeded.

03:17.840 --> 03:24.000
Again, if the license limit is exceeded, the events are buffered
and fed back into the stream when the input is below the license limit.

03:24.560 --> 03:30.040
And, if the buffer is already full, again the new events are dropped
and a special event for the Console is generated.

03:31.280 --> 03:35.160
The CRE evaluates every single event against every active rule.

03:35.920 --> 03:40.680
If none of the rules are triggered on the event,
the event is dropped from further processing.

03:41.600 --> 03:43.240
If at least one rule triggers

03:43.240 --> 03:47.240
(which happens in our FW Deny events example

03:47.240 --> 03:51.280
because the number of events within a certain time period exceeds a
threshold value in a test rule),

03:51.600 --> 03:54.640
then the event is properly marked for further processing.

03:55.480 --> 03:59.440
This way the Magistrate on the Console knows how to handle this event

03:59.880 --> 04:04.160
(it creates a new offense and adds the event
to any number of existing offenses).

04:05.160 --> 04:06.360
In our case,

04:06.360 --> 04:14.240
the number of accumulated FW Deny events is sufficient evidence to
instruct the Magistrate that these events are worthy of an offense.

04:15.120 --> 04:23.280
If you have configured any live streaming views on the Console,
the CRE can also stream every incoming event to the log activity tab.

04:24.000 --> 04:29.360

This way all our FW Deny events are displayed
in a streaming Dashboard on the Console.

04:30.440 --> 04:36.320
The event storage component is responsible for storing
all events and flows in the Ariel database.

04:37.120 --> 04:40.160
The filter then passes on the data to the Accumulator.

04:40.920 --> 04:47.920
The Accumulator manages all the defined searches such as Reports
and Dashboards, that were set up by an analyst on the Console.

04:48.480 --> 04:54.440
Based on the search parameters, the Accumulator stores
data in the Accumulations Ariel database.

04:55.040 --> 05:00.680
This data will be used later by the Console to display results
through the GUI or by creating reports.

05:01.960 --> 05:07.680
The Host Profiler also receives the event data and searches
for any new host or port events.

05:08.800 --> 05:17.240
If any new hosts or ports are detected, they are sent
to the Console's Vulnerability Information Server.

05:18.840 --> 05:25.320
Now, in the Console, our processed and coalesced FW
Deny events are received from the Event Processor.

05:26.920 --> 05:33.800
Once again, the overflow filter counts the incoming normalized events
to ensure that the license limit for the appliance is not exceeded.

05:34.320 --> 05:40.640
If the license limit is exceeded, the events are buffered
and sent back into the stream when the input is below the license limit.

05:41.120 --> 05:46.640
And, if the buffer is already full, the new events are dropped
and a special event for the Console is generated.

05:47.120 --> 05:51.000
The Magistrate receives our FW Deny events from the Event Collector.

05:51.880 --> 05:59.160
Based on the Index Property and Index Property Value,
the Magistrate knows that these events need to be raised as an offense.

05:59.480 --> 06:05.200

Before creating the new offense,
the CRE inside the Magistrate makes sure if these events

06:05.240 --> 06:11.120
should either be assigned to a new offense
or whether they can be attributed to other existing offenses.

06:11.720 --> 06:19.320
Collecting this additional data also helps to provide a clearer review
to analysts in the GUI (by displaying related events and flows).

06:20.120 --> 06:24.200
In case the Magistrate needs to access
additional event and flow records,

06:24.200 --> 06:31.040
it uses the Ariel Proxy to communicate with Ariel Query Servers
that are on other the Event Processor appliances.

06:31.480 --> 06:37.160
In addition to the Magistrate component,
the Console also houses the Anomaly Detection Engine.

06:37.760 --> 06:41.520
It examines behavioral, anomaly, or threshold-based rules

06:41.520 --> 06:46.960
that can be used to create new offenses or add
additional evidence and details to existing offenses.

06:48.160 --> 06:50.880
Based on collected event and flow data,

06:50.880 --> 06:55.840
the Vulnerability Information Server component on the Console receives
information about

06:55.840 --> 06:59.960
new hosts or ports that are not yet contained in its Asset database.

07:00.720 --> 07:04.400
Those new assets are added to the PostgreSQL Asset database.

07:08.480 --> 07:15.000
In this video, you experienced how a set of FW Deny events,
which originated on a Checkpoint firewall log source,

07:15.000 --> 07:21.800
traversed a set of QRadar SIEM components comprised of Event Collector,
Event Processor and Console,

07:21.920 --> 07:29.800
until they are displayed in a security analyst's user interface,
where they are linked to an offense; all in a fraction of a second.

07:30.200 --> 07:36.160
Employing this powerful collaboration allows security analysts to make timely and accurate decisions

07:36.160 --> 07:39.840
regarding maintaining the security posture of their organization.

07:42.600 --> 07:46.680
This concludes the QRadar Architecture course. Thank you