Cribl Double

CRIBL AMAZING DATA STORIES

YOUR GUIDE TO THE DATA ENGINE FOR CYBERSECURITY

Cribl Amazing Data Stories

Your Guide to the Data Engine
for Cybersecurity

Contents

Foreword

by Clint Sharp, Dritan Bitincka, and Ledion Bitincka, Cribl

At Cribl, we stand for choice. Choice for what to do with your data and what tools best fit your needs, today and in the future. After all, your data is your data. That's why the Cribl product suite is vendor-agnostic by design and purpose-built for you, the IT and Security professionals who keep the business running.

Your needs are unique and ever-changing. And what got you through the past 10 years won't get you through the next 10 years. Cribl is here to help you transform your data strategy and, ultimately, we want data to work for you, freeing up your time to focus on the many mission-critical tasks at hand.

From the outset, we've taken a first principles approach to innovating for you. We seek to understand your biggest problems, then we build the right solutions and continuously iterate to ensure our products work for you. As the founders, we have firsthand experience with many of the challenges facing IT and Security professionals today, but we don't pretend to know everything. We're insatiably curious and partner closely with you, our customers and users, as we continue to build our products.

Before we dive in further, we want to give you a brief overview of our product suite. We started back in 2018 with Cribl Stream, the industry's leading observability pipeline that can collect data from almost any source, process billions of events per second, and transform and route data to optimize your analytics, storage, and retention.

Since then, we've broadened the scope of the company to meet your needs across data's entire lifecycle. We launched Cribl Edge to move intelligence closer to the data source and simplify managing fleets of agents, enabling you to deal with thousands more agents than you ever dreamed possible.

Next came Cribl Search, the industry's first search-in-place solution. Search gives you the insights you need, while minimizing the cost and complexity of ingesting and shipping data before analysis.

Most recently we launched Cribl Lake, a turnkey data lake solution that you can set up and configure in a few clicks, offering automated optimization of storage and open formats for easy access and retrieval without being a data expert.

That's where we are today on this journey, together with all of you, as we empower you to transform your data strategy. Now, you might be asking yourself: So what's this book all about? This is our guide for you, full of tips and techniques to get the most value out of your data, based on insights from the people who build and support our products and solutions for customers. The stories in this book illustrate a variety of approaches to getting your data to the right places, in the most optimized formats, and in the most efficient ways.

Whether you skim the chapters to see what's possible, or directly apply these techniques using the resources we provide, we hope this book will help you discover new ways to unlock the full value of IT and Security data.

We also hope you have fun reading it. We know this is serious work, but we can still have fun doing it.

So we invite you to dive right in and join us on this ongoing journey. It's not science fiction. It's not a mystery. It's Cribl, the Data Engine for IT and Security.

Preface

by Michael Katz, Cribl

It was midnight in the SOC. (It always feels like midnight here.) A cold wind blew the door open. I thought, good, the air conditioning's working. Then, in walked a dame wearing a million-dollar hoodie. That's what I noticed first, followed by the brightest eyes, the purplest hair, the whitest hat, and the highest red high-top sneakers I've ever seen. Nerds like that are one in a million.

"I've got a tip for you, Gumshoe," she said. She dropped a classy teal-colored business card on my desk. It seemed to float down in slow motion. (I thought, "Isn't it a little early in this movie for slow-mo?" Whatever.)

When I looked down, the card had just one word printed on it: "Cribl," it said.

That was it. The clue that changed my life. No longer did I need to pay big bucks to analyze a mess of irrelevant evidence, like events containing benign well-known IP addresses. Stash that innocuous stuff on the cheap, in reliable long-term storage like Cribl Lake or S3. Focus on where the bad guys have left their fingerprints—like malicious domains, or access patterns that smell fishy.

Everybody's got an angle. But there ain't enough cash to go around, data volumes being what they are these days, to send every event to those big analytics shops downtown. See, we gotta be selective if we're gonna uncover the really crooked patterns, secure our infrastructure, investigate break-ins if they do occur, crack cases, and send the bad guys packing.

I followed the lead on that teal card, and I was glad I did. Cribl helped me put my money where it matters. I've got better overall data security, faster and

leaner analytics, integrated data storage, and search-in-place. When crooks do attack, I can crack cases faster. And with my savings, I've even got a little extra to take to the track. I'm betting on a winning horse. After my winning bet on Cribl, I'm feeling lucky.

–

We get testimonials like this—but real ones, not made up—from our customers all the time. We're gratified by how well Cribl products have helped SecOps defenders up their game. In the following pages, Cribl's ace internal practitioners share tips on the techniques they've built for customers with Cribl Stream, Cribl Edge, Cribl Search, and Cribl Lake.

There are two sides to every amazing story, and you'll notice that there are two sides to this book. This side focuses on techniques of particular interest to Security sleuths. The flip side has a brief overview of Cribl's products and online resources, and then presents tips and techniques relevant to both Security and IT practitioners. We hope you'll read both!

There are a million stories in the city. Here, we present a few of our best. Turn the page and see where they take you.

–

(Pssst...yeah, you. If you ain't already in on the action, we got a hot tip for you: Sign up for a free Cribl.Cloud account at `cribl.io/solved` to start solving your own security hard cases. For the most up-to-date version of this book's contents, see `cribl.io/amazing`.)

Acknowledgments/ Editor's Note

by Michael Katz, Cribl

Grigori Melnik had the vision for this book and set it in motion. Abby Strong, Joel Vincent, Erin Sweeney, and Meredith Torres kept it moving. Christopher Gales shaped it from an amorphous space blob into something flightworthy, adding a delicate balance of creativity and realism. Eric Wolfe and Mara Sahleanu provided graphics and illustration magic, developing some imagery in collaboration with `ideogram.ai` and Canva.

Mara and Alyssa Houk at Cribl, and Tamara Kreiss and Lesley Harrison at Bay Area Graphics, provided the overall design, layout, and production talent that gave this bookness. Douglas Howatt deftly copy-edited the whole manuscript and kept it real. Maliha Balala, Emil Mikhailov, Ed Bailey, Jackie McGuire, Igor Gifrin, and Josh Biggley provided crucial connections, content, encouragement, and more illustrations. Jennifer Marandola graciously allowed me a break from developing ongoing Cribl technical guidance to assemble this, working from the blueprint of a von Neumann probe. Marissa Dahlson, Diane Chiu, and Matt Lanza helped arrange my space boarding pass.

The greatest thanks go to all the authors for their ingenuity in writing these recipes, and for their patience in reworking text and graphics with me. And to Clint Sharp, Dritan Bitincka, and Ledion Bitincka for launching the generational starship that is Cribl. Any errors, misstatements, or anachronisms are my bad.

SECTION 01

Optimize Security Data

01 | Route and Optimize Security Data

Unleashing the Cribl CrowdStrike Pack's model infrastructure

by Ahmed Kira, Cribl

PROBLEM

CrowdStrike FDR (Falcon Data Replicator) is one example of a class-leading endpoint monitoring solution. It collects a wealth of activity data from each managed endpoint, including network connectivity, DNS requests, process activity, and health checks. It reports more than 400 event types in all. These events are a gold mine for threat hunters and blue teams looking for unusual or malicious activity. But the resulting treasure trove is extremely voluminous, making it costly to place all this data in a SIEM (security information and event management) system.

SOLUTION

Cribl Stream offers a cost-effective solution, giving you the choice to send a full-fidelity copy of these logs to inexpensive object storage – such as Amazon S3, Azure Blob Storage, Cribl Lake, or your own object store managed by Cribl – while sending only events of interest to a SIEM.

WHY CONSIDER THIS APPROACH?

To accelerate this demonstration and your own setup, this chapter is written around the Cribl CrowdStrike FDR Pack, which offers predefined logic to address the common challenges of processing all this data. (You can apply the same principles to importing Cribl Packs for other SIEM destinations, or to building your own Cribl Stream optimization Pipelines.)

In this representative Stream case study, the end result you'll see is asset-level enrichment (for example, obtaining `ComputerName` from the `aid` field). You'll also see an impressive 40–95% reduction in data volumes sent to the SIEM. This reduction isn't just a matter of dropping unwanted events, but a combination of:

- Aggregating network events.
- Removing DNS requests that map to the 5,000 top popular websites.

Dropping unwanted events, based on the `event_simpleName` field value.

- Removing unwanted fields.
- Removing null fields.
- Removing duplicate fields.
- Sampling noisy "External API" events.

In these screenshots from Stream's Pipeline diagnostics, you can see some examples of reduction in event volume and/or size, by transformation type.

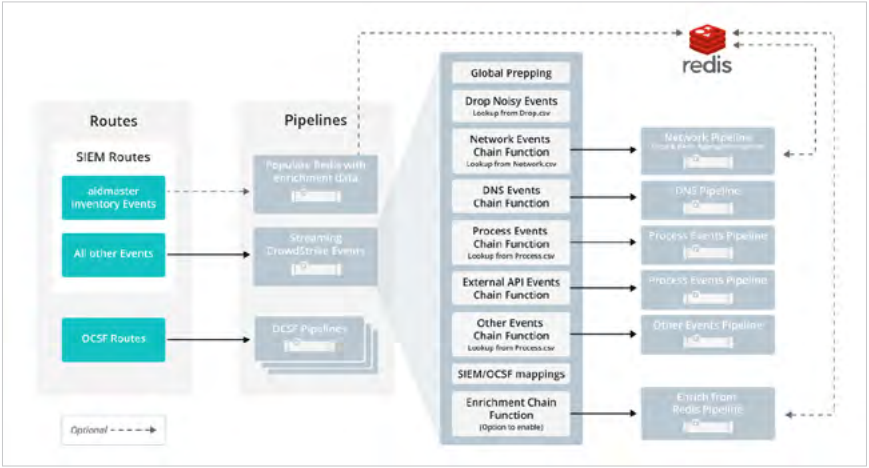
Representative Stream optimization of security data

HANDS-ON RESOURCES

The CrowdStrike FDR Pack is a free resource, which provides its own instructions and sample data files. You can directly import the Pack to Cribl Stream, or download it, from the Cribl Packs Dispensary at packs.cribl.io. Browse the Dispensary for Packs that offer similar predefined optimization logic tailored to other SIEM and analytics destinations.

SCENARIO

The diagram below shows the Pack-based workflow we'll examine. The Pack optionally uses Redis for enrichment and stateful aggregation across Worker Processes. As a UI for the Redis service, we used Redis Commander (`npmjs.com/package/redis-commander`).



Default workflow using Cribl CrowdStrike FDR Pack

WHAT WE'LL DO

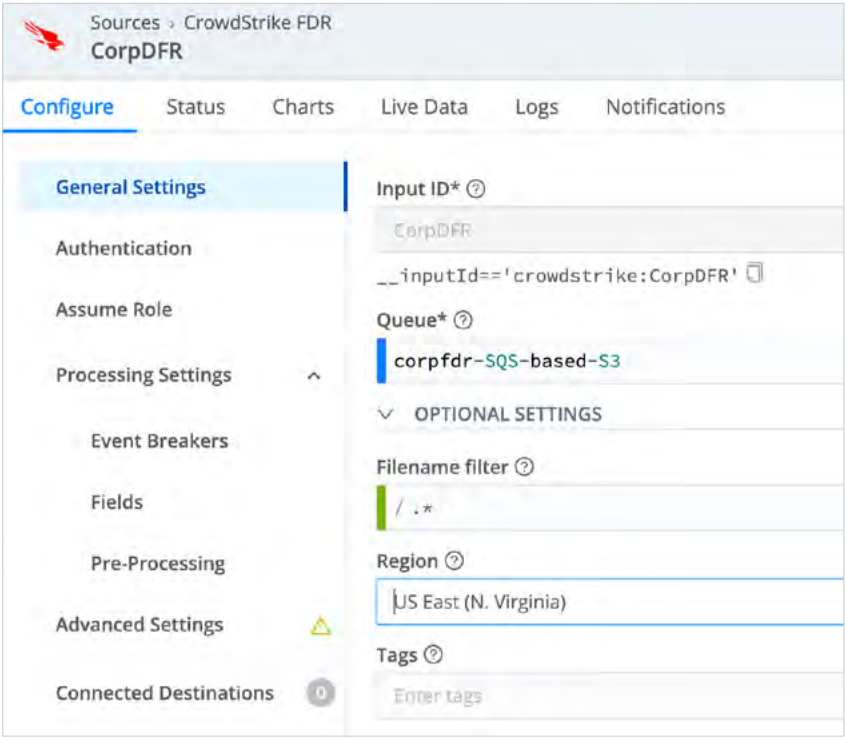
We'll cover the following techniques for isolating and enriching high-value events:

- Capturing data from CrowdStrike FDR.
- Enabling/disabling chained Stream Pipelines, for applications like DNS filtering.
- Controlling which events to drop.
- Enriching events with computer names.
- Enabling Redis-based network aggregation.
- Smart processing of DNS events.

CAPTURING DATA FROM CROWDSTRIKE FDR

The CrowdStrike FDR Pack is designed to work with CrowdStrike FDR logs written to the CrowdStrike-provided Amazon S3 bucket. Contact your CrowdStrike team to obtain access to CrowdStrike FDR. When this is set up, CrowdStrike will provide an SQS queue URL, an S3 bucket URL, and an access key and secret key.

Within Cribl Stream, you will typically configure a CrowdStrike FDR Source with these details, as shown on the next page.

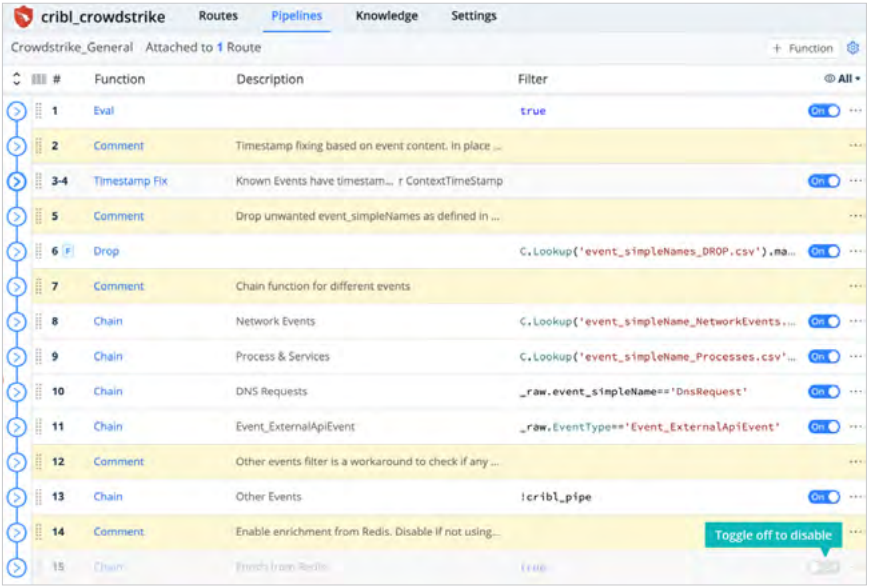


Configuring a CrowdStrike FDR Source in Cribl Stream

(You could instead choose to configure an Amazon S3 Source in Stream. But note that an S3 Collector would not work here.)

ENABLING OR DISABLING THE PROCESSING YOU WANT OR DON'T WANT

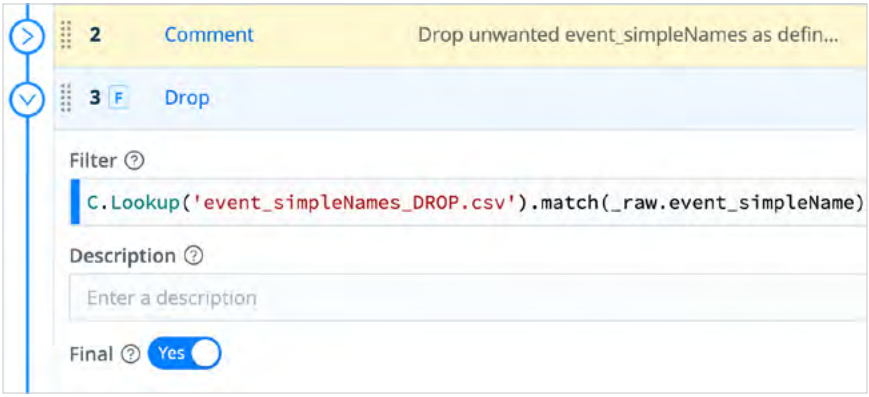
As illustrated in the diagram above, there is one master Pipeline that processes all streaming CrowdStrike events. Within that Pipeline are various Chain Functions with filters. To enable/disable any processing, just toggle that row's slider on or off. For example, if you don't have Redis, ensure that the Enrichment Chain Function is toggled off, as you see below.



Dropping unwanted events

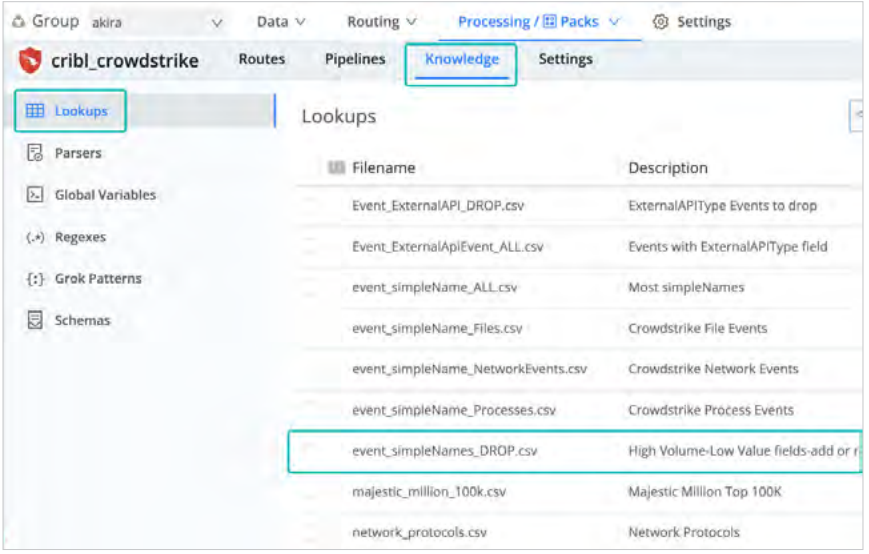
DROPPING UNWANTED EVENTS

One of the first things you'll likely want to do is control which CrowdStrike events Cribl Stream keeps and which it drops. To simplify this process, the Pack ships with a lookup file (CSV) containing `event_simpleNames` values to drop. The Pack's main Pipeline, `CrowdStrike_General`, contains a Drop Function that relies on this lookup to drop matching events.

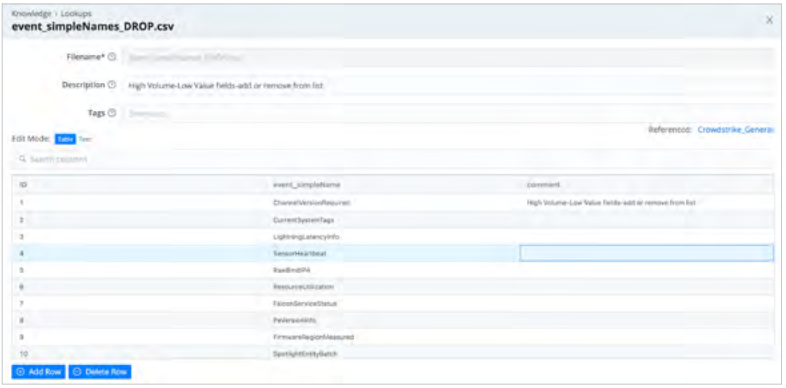


Drop Function's default configuration

To control the list of events being dropped, click into the Pack's own **Knowledge** > **Lookups** section.



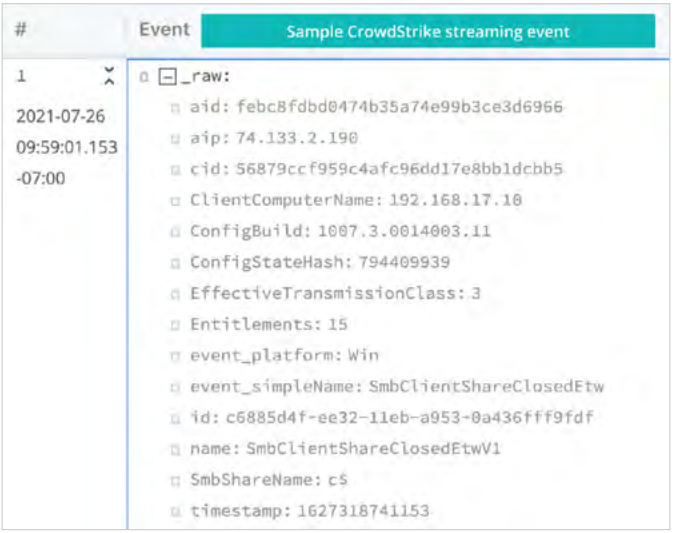
Customizing the lookup file of event names to drop



Editing the lookup file's rows of events to drop

ENRICHING EVENTS WITH COMPUTER NAMES

Enrichment is valuable because the streaming CrowdStrike events include a computer ID only as the `aid` field. The actual Computer Name or hostname is not populated in those events.



Sample CrowdStrike streaming event, omitting computer/host names

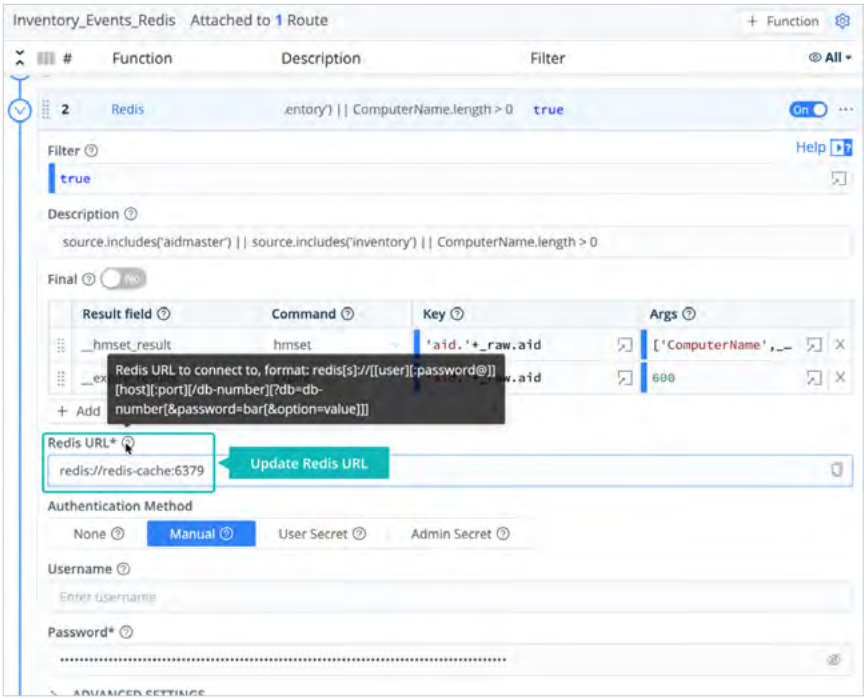
Instead, CrowdStrike exposes inventory information in separate events, which are typically published nightly.

#	Event	Sample CrowdStrike inventory event
1	# _time: 1650503131.895	
2022-04-20 18:05:31.895 -07:00	a AgentLoadFlags: 0	
	a AgentLocalTime: 1630690863.6720002	
	a AgentTimeOffset: 504.852	
	a AgentVersion: 6.27.14003.0	
	a aid: a19c185c931a4607961efab81784eb58	
	a aip: 73.168.3.7	
	a BiosManufacturer: VMware, Inc.	
	a BiosVersion: VMW71.00V.16722896.B64.2008100651	
	a ChassisType: Other	
	a cid: 56879ccf959c4afc96dd17e8bb1dcbb5	
	a City: Darien	
	a ComputerName: SE-SMC-MAC11-BL	
	a ConfigIDBuild: 14003	
	a Continent: North America	
	a Country: United States	
	a cribl_breaker: Break on newlines	
	a event_platform: Mac	
	a FalconGroupingTags: FalconGroupingTags/VMware	
	a FirstSeen: 1620238836	
	a HostHiddenStatus: Visible	
	a MachineDomain: none	
	a OU: none	
	a PointerSize: none	
	a ProductType: 1	
	a SensorGroupingTags: -	
	a ServicePackMajor: none	

Sample CrowdStrike nightly inventory event

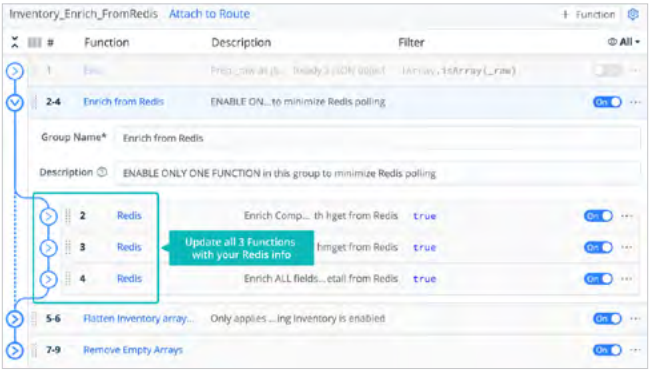
Because of how this information is sent, Cribl Stream can use Redis to enrich the `ComputerName` and/or other asset information in each streaming event. To enable this enrichment:

1. Stand up a Redis server. (See the end of this chapter for links to a setup tutorial, and to a video about Stream/Redis interaction.)
2. Within the Pack, edit the `Inventory_Events_Redis` Pipeline, and open its first Redis Function.
3. As shown below, update the Redis URL, and create a user or admin password that you'll be able to reference in all other Redis Functions. Then resave this Function.



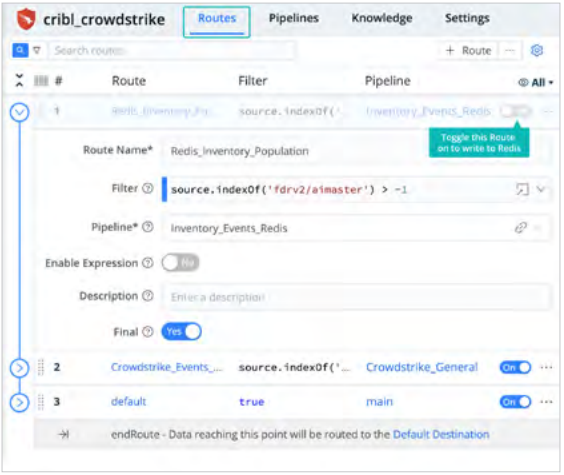
Configuring first Stream Redis Function with URL and authentication details

4. In the `Inventory_Enrich_FromRedis` Pipeline, update all other Redis Functions with the same details you set in the first Function. (This Pipeline works in the opposite direction, closing the enrichment loop by pulling data back in from the Redis store.)



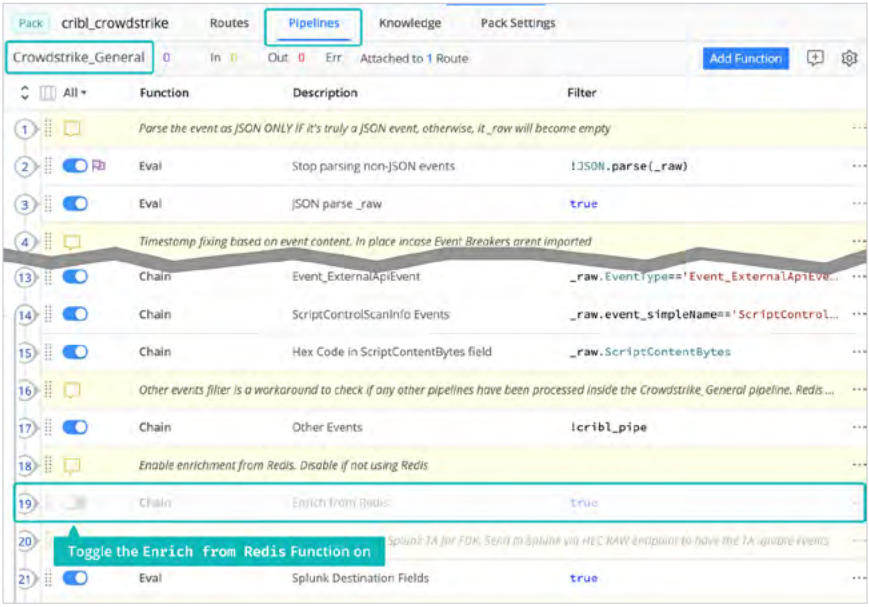
Next, update the remaining Redis Functions to match

5. Still working within the Pack, enable the Route that writes inventory events to Redis.



Enabling the Pack's included `Redis_Inventory_Population` *Route*

6. Enable the Chain Function that enriches events from Redis data. This Function enriches events by chaining processing to the Pack's separate `Inventory_Events_Redis` Pipeline. (As with other details here, this Function's position in the latest Pack might differ from this screenshot.)

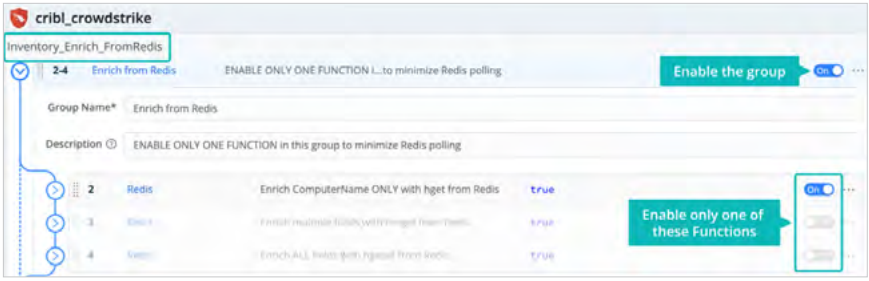


Enabling the Chain Function that enriches Redis events

By default, this chaining populates all contents of the CrowdStrike inventory event in Redis, making them available for enrichment.

However, if all you care to enrich is `ComputerName`, it would be wise to update the `Inventory_Events_Redis` Pipeline to write only those fields into Redis.

The Pack's predefined Pipeline actually offers you three enrichment options: Enrich only the `ComputerName` field, or enrich a set of fields that you control, or enrich all available fields. These options are exposed as three separate Functions in the `Inventory_Enrich_FromRedis` Pipeline. To prevent unpredictable interactions, you should enable only one of these Functions.



Enable the enrichment group, then enable only one nested enrichment Function

If you want to closely control which fields are enriched, edit the third "multiple fields" Function, and add the desired fields to its `Args` section. Maintain the same format, with comma and single quotes.

Customizing which fields to enrich

After you save, commit, and deploy this change, every event will be enriched with the information you want.

STATEFUL AGGREGATION THROUGH REDIS

The Pack offers two methods for aggregating the voluminous network connectivity events: Either enrich natively in Cribl Stream via an Aggregations Function (the default), or enrich statefully across all your Stream Worker Processes via Redis. You should enable only one of these groups (not both) within the `CrowdStrike_Network` Pipeline.

The aggregation works by counting the number of times the client connects to the same server on the same destination port. The aggregation logic assumes that the lower port in a network event is the server port, and that the higher port reflects a client-side source port. This logic preserves the values of the unique source ports, and passes them as a list in the aggregated event.

For example, assume that CrowdStrike reports the following seven events:

```
Source IP: 192.168.1.1 Source port: 12345 Destination IP: 172.16.9.9 Destination port: 53
Source IP: 192.168.1.1 Source port: 12346 Destination IP: 172.16.9.9 Destination port: 53
Source IP: 192.168.1.1 Source port: 12347 Destination IP: 172.16.9.9 Destination port: 53
Source IP: 192.168.1.1 Source port: 12348 Destination IP: 172.16.9.9 Destination port: 53
Source IP: 192.168.1.1 Source port: 12345 Destination IP: 172.16.9.10 Destination port: 389
Source IP: 192.168.1.1 Source port: 12345 Destination IP: 172.16.9.10 Destination port: 389
Source IP: 192.168.1.1 Source port: 12345 Destination IP: 172.16.9.10 Destination port: 389
```

Aggregation will distill these into the following two events:

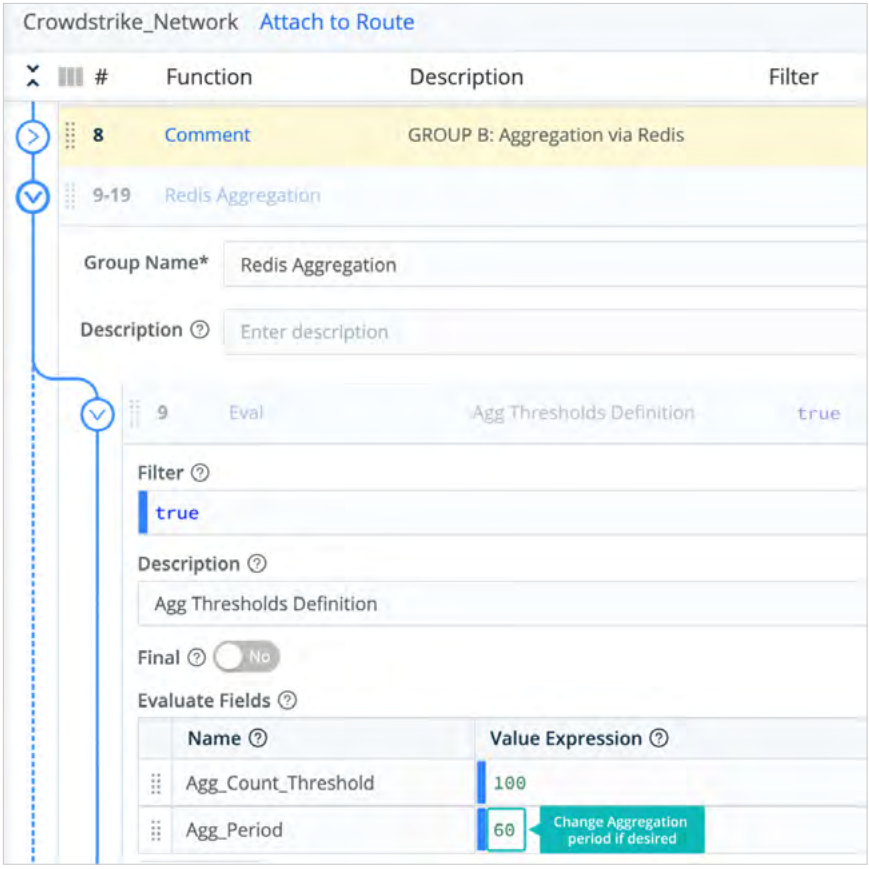
Event 1:

```
Count: 4 Source IP: 192.168.1.1
Source ports: [12345,12346,12347,12348]
Destination IP: 172.16.9.9 Destination port: 53
```

Event 2:

```
Count 3 Source IP: 192.168.1.1
Source ports: 12345
Destination IP: 172.16.9.10 Destination port: 389
```

The aggregation period is set to a default range of 10 to 30 seconds, but you can easily modify these defaults in the `Redis Aggregations` group's Eval Function.



Configuring the interval for Redis aggregation (in a separate Eval Function)

To enable the Redis aggregation, the steps include:

- Click into the Pack's `CrowdStrike_Network` Pipeline.
- Disable `GROUP A: Cribl Native Aggregation`.
- In `GROUP B: Aggregation via Redis`, edit all Redis functions to update the Redis URL and user or admin secret.

- Enable `GROUP B`.
- Save, commit, deploy, and enjoy.

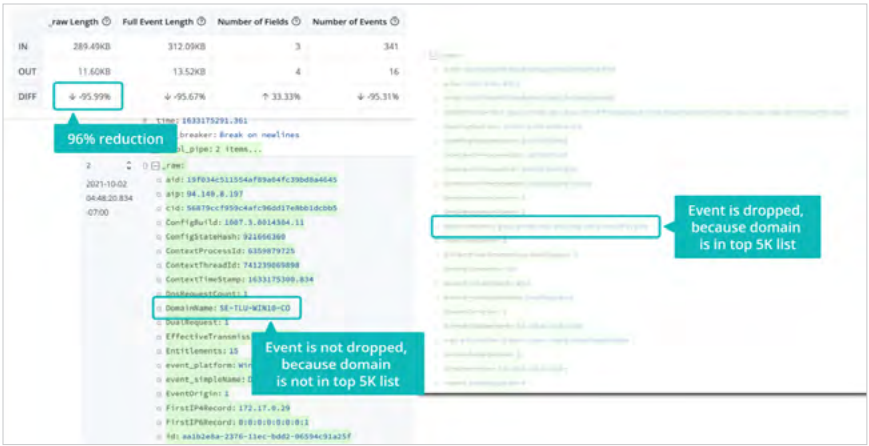
The result will render aggregated events similar to these:

Aggregated events

DO YOU REALLY NEED ALL DNS EVENTS?

DNS data is very voluminous, but has huge value for Security Ops teams. However, do you really need to keep DNS records for known addresses? Examples would be records for someone accessing Gmail, Facebook, and other non-sensitive endpoints. You will also have events that show you file transfer activity and process-level information, many of which will be redundant in a different way. The DNS records you truly need are for malicious addresses, not the top 10, 5,000, or even 100,000 addresses.

The CrowdStream FDR Pack ships with a lookup file containing the Majestic Million top 5,000 URLs. The Pack's `Crowdstrike_DNS` Pipeline contains several Lookup Functions that work with this CSV. If the last four portions of an event's domain match against this list of common domains, Stream will drop the events. (An example matching domain might be `my.docs.google.com`.)



DNS lookup preserves unusual events, while dramatically reducing routine data flow

VARIATIONS

The CrowdStrike FDR Pack accounts for the most typical use cases, and for the largest-volume event types, to help you maximize value from the combination of Cribl and CrowdStrike. But the Pack is just a starting point. You can modify it and add content to it to suit your needs.

Cribl support and peer advice are readily available on Cribl's online Community resources. And on the Cribl Packs Dispensary, you'll find other Packs to optimize data for other SIEMs like Exabeam (the focus of this book's next chapter), SentinelOne, and Microsoft Sentinel.

WHAT'S NEXT?

For details about the resources listed in this chapter, see:

- Cribl Packs Dispensary (packs.cribl.io).
- Cribl CrowdStrike Pack demo video (youtube.com/watch?v=WohpshjFtzE).
- Stream CrowdStrike FDR Source (docs.cribl.io/stream/sources-crowdstrike).
- Stream Amazon S3 Source (docs.cribl.io/stream/sources-s3).
- Cribl Lake (docs.cribl.io/lake).
- Redis server setup, third-party tutorial (flaviocopes.com/redis-installation).
- Redis Commander (npmjs.com/package/redis-commander).
- Other Redis GUI options (redis.com/blog/so-youre-looking-for-the-redis-gui).
- Cribl Redis Lookups demo video (vimeo.com/504479828)
- Aggregations Function (docs.cribl.io/stream/aggregations-function).
- Redis Function (docs.cribl.io/stream/redis-function).

- Eval Function (docs.cribl.io/stream/eval-function).
- Sampling Function (docs.cribl.io/docs/sampling-function).
- Dynamic Sampling Function (docs.cribl.io/stream/dynamic-sampling-function).
- Suppress Function (docs.cribl.io/stream/suppress-function).
- Cribl Stream Data Preview (docs.cribl.io/stream/data-preview).

–

Cribl Stream's capabilities are always evolving, and you might see some differences from the UI screenshots and instructions provided in this edition. Luckily, Cribl Packs keep evolving, too – expect the Pack illustrated here to be progressively updated with new capabilities. (If you've customized the Pack's Pipelines, Stream ensures that importing an upgraded Pack won't overwrite them. Details at docs.cribl.io/stream/packs/#upgrading.)

02 | Optimize High-Value Data for Cybersecurity Platforms

Filtering Options Using Cribl Stream and Cribl Packs

by Jim Apger, Cribl

PROBLEM

Security-related inbound data often includes low-value events. These events drain resources that you could redirect to expanded discovery and analysis.

SOLUTION

This chapter shows how to filter low-value events out of syslog data ingested from sources like Palo Alto Networks, before you send your data on to common security analytics platforms. (We use Exabeam as our analytics example here.) We'll show how to enable relevant Cribl Stream Functions, integrations, and routing.

WHY CONSIDER THIS APPROACH?

By filtering out and dropping low-value data, your security organization can create additional capacity for new data sources, while reducing storage costs. If you track a cybersecurity framework (CSF) like MITRE ATT&CK, these filtering capabilities can help you address visibility gaps and increase your coverage of attack tactics and techniques.

HANDS-ON RESOURCES

If you aren't already using Cribl Stream, sign up for a free Cribl.Cloud account at cribl.io/solved to get your data flowing, with or without Packs.

As shown below, you can directly import many of this chapter's techniques from Cribl's Exabeam Pack for Palo Alto. It's a free resource on the Cribl Packs Dispensary at packs.cribl.io.

For details on the Cribl Stream Source and Destination integrations we reference below, see docs.cribl.io/stream.

THE END UP: CONFIGURE EXABEAM TO RECEIVE CRIBL STREAM DATA

Let's start on the output side – how Cribl Stream should relay inbound syslog data to Exabeam. As of this writing, we recommend creating a Webhook Cloud Collector in your Exabeam console: Select **Collectors** > **Cloud Collectors** > **New Collector**, and then select the **Webhook** tile.

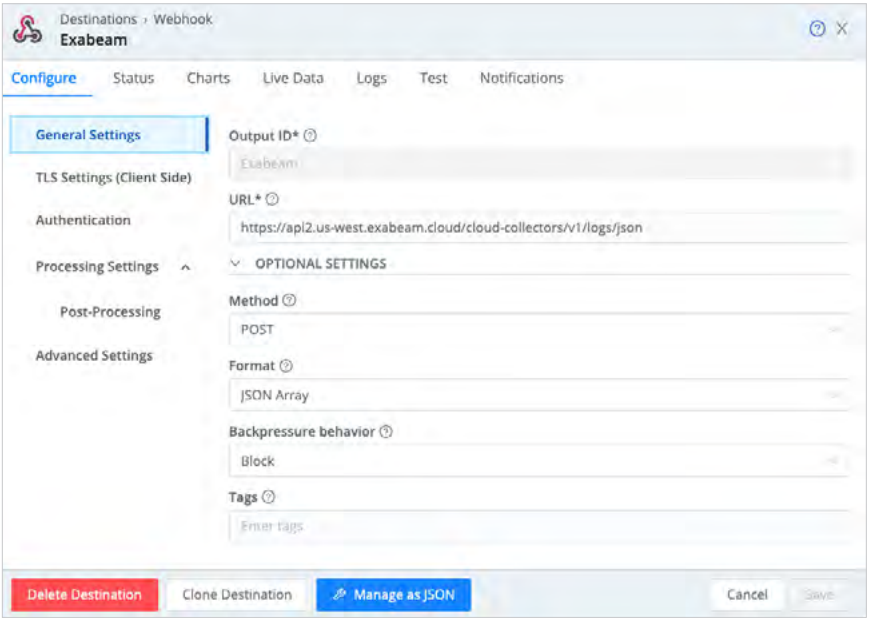
Name your Webhook Collector, and select **JSON** as the format. This will display an **AUTHENTICATION TOKEN** and a **LINK TO LOGS**, which you will use in Cribl Stream next.

OUTPUT: CONFIGURE CRIBL STREAM TO SEND TO EXABEAM

Now, in Cribl Stream, configure a corresponding Webhook Destination to send data to Exabeam.

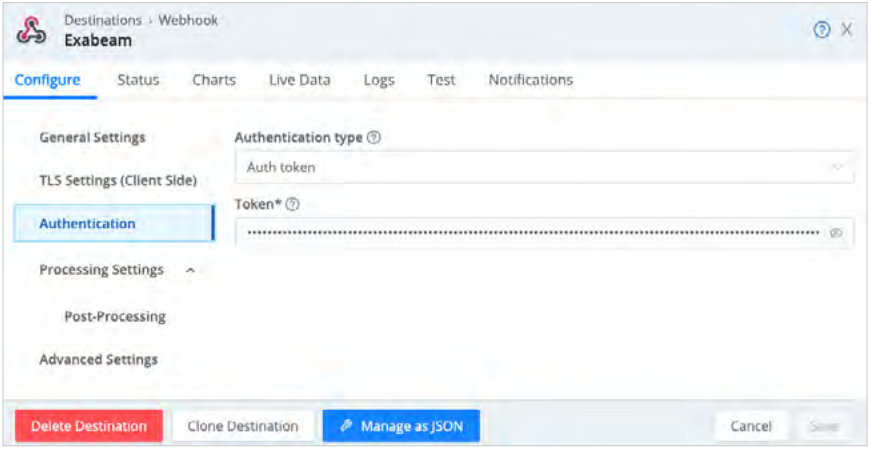
In your Stream Worker Group, select **Data** > **Destinations** > **Webhook**. Next, select **Add Destination** to open and configure the modal shown below.

We've set the URL field by pasting the **LINK TO LOGS** value from the Exabeam Cloud Collector.



Stream Webhook Destination, configured for Exabeam – General Settings

On the **Authentication** left tab, we've pasted the **AUTHENTICATION TOKEN** value the Exabeam Cloud Collector.



Stream Webhook Destination, configured for Exabeam – Authentication settings

TEST OUTBOUND THROUGHPUT (OPTIONAL)

In the above screen capture, note the **Test** tab available at the upper right. You can select this tab to send a few events over to Exabeam, and verify that they appear in the Exabeam console. (If you send a customized event, make sure it's a JSON event wrapped in an array.)

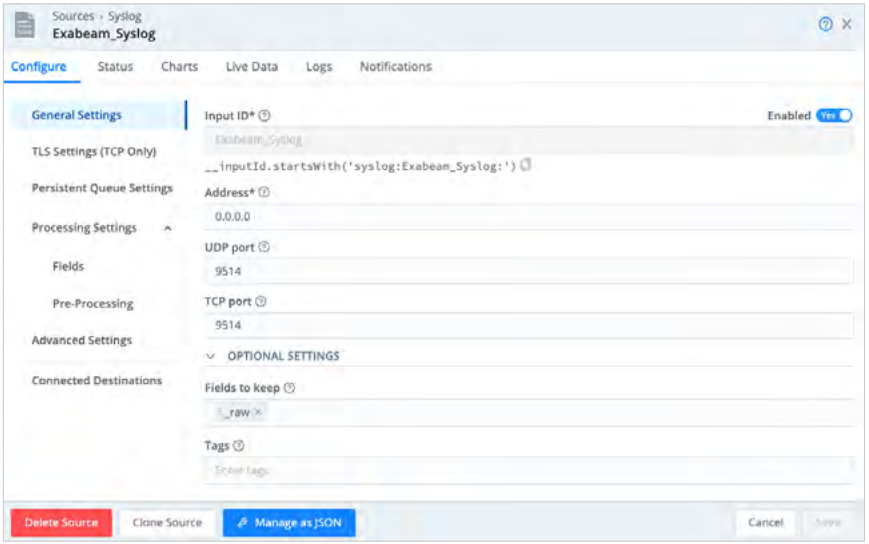
INPUT: CONFIGURE CRIBL STREAM TO RECEIVE SYSLOG DATA

In our scenario, we want Cribl Stream to ingest syslog data and then transmit it exactly as Stream received it. To set up the ingest side:

In your same Stream Worker Group, select **Data** > **Sources** > **Syslog**. You can then enable one of the default Syslog Sources, which will select the default TCP and/or UDP port.

RETAINING _RAW FOR DATA FIDELITY

We'll need to add one very important custom setting to the Syslog Source, as shown below. In **Fields to keep**, enter `_raw`. This way, we'll retain events' original `_raw` field as we process the events and send them on to Exabeam.



Retaining _raw to ensure data fidelity

Before we complete connectivity in Stream, we want Cribl's Exabeam Pack for Palo Alto available. The Pack will provide the Stream Pipelines and Functions that optimize our syslog data – so we won't need to write those. We'll plug this Pack into Stream in place of a Pipeline.

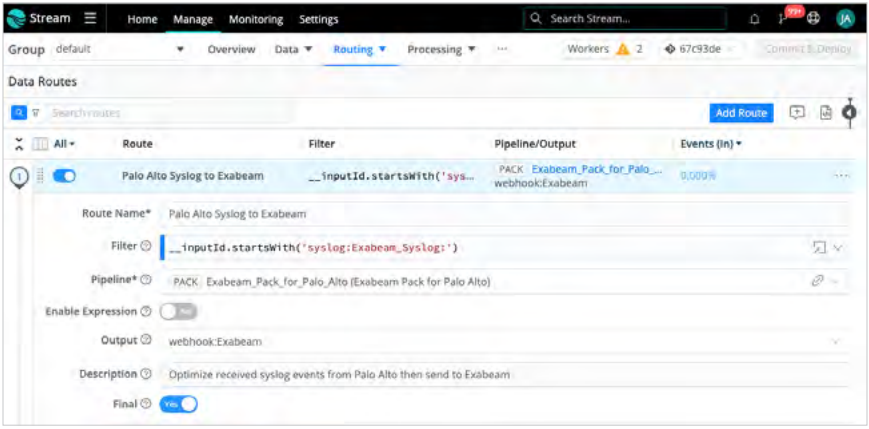
If you're working through this scenario as you read, you can download the Pack from the Cribl Packs Dispensary at `packs.cribl.io`, or install it directly from within Stream by selecting **Processing** > **Packs** > **Add Pack** > **Install from Dispensary**.

CONNECT EVERYTHING: ROUTE INBOUND TO OUTBOUND DATA

To complete the circuit, we need to connect our Syslog Source to our Exabeam Webhook Destination. We do this with a Route.

With the focus on your same Worker Group, select **Routing** > **Data Routes** > **Add Route**. Then configure your new Route as shown below.

The **Filter** expression says: Where incoming syslog data bears the Syslog Source's Input ID, this indicates syslog data destined for Exabeam. So send the event to the Exabeam Pack, then route it to our corresponding Webhook Destination.



Route configuration for PAN syslog to Exabeam

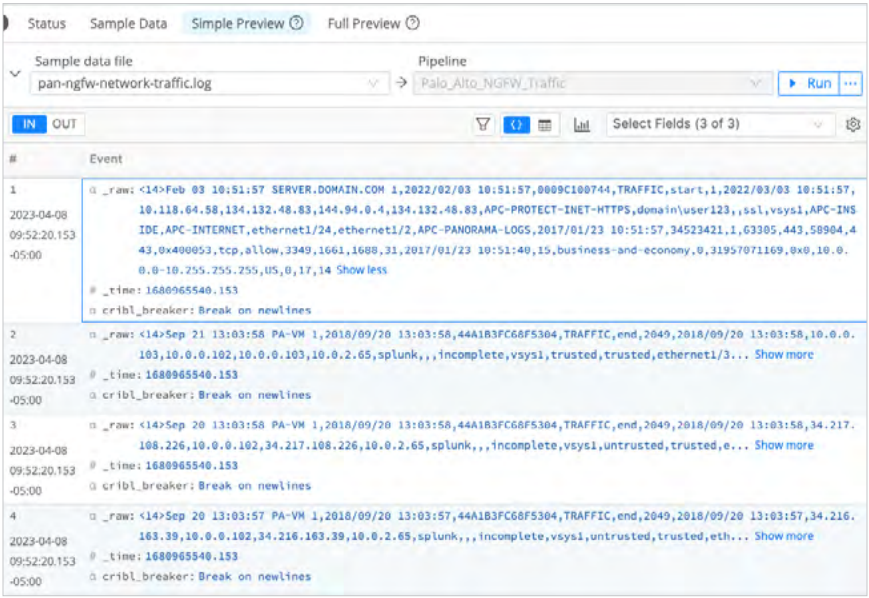
ONE ROUTE OR MANY?

With the **Filter** expression shown above, and with the Pipeline configuration we've imported from the Pack, we're assuming that this Syslog Source will receive only Palo Alto Networks (PAN) data. Configure your PAN sender to relay data to Cribl Stream via syslog.

If you had multiple data sources sending events into this Syslog Source, you'd need multiple Routes. Each would have a tighter **Filter** expression than we've shown above, tuned to capture specific events that you want to direct into a corresponding Pipeline (or Pack).

CHECK INBOUND EVENT STRUCTURE

After you've configured Palo Alto Networks to send syslog events to Cribl Stream, you can confirm the inbound data structure by performing a capture on your Syslog Source in Stream. You should see events looking roughly like the screenshot below – containing only `_raw`, `_time`, and `cribl_breaker` fields. (That last field identifies the Stream Event Breaker that isolated these fields.)



Data preview – inbound syslog Events

STREAMLINE DNS DATA

Let's switch gears a bit. So far, we've looked at Palo Alto Networks (PAN) traffic delivered to Cribl Stream via syslog. Let's now look at DNS Security logs from PAN. DNS data is highly valuable within a security operations center for multiple practitioners – including analysts, investigators, threat hunters, and threat intelligence engineers.

Unusual DNS requests and responses hold valuable potential as Indicators of Compromise (IOCs), especially during investigations after an incident. However, the challenge lies in the massive log volume that these requests and responses generate for security teams to navigate.

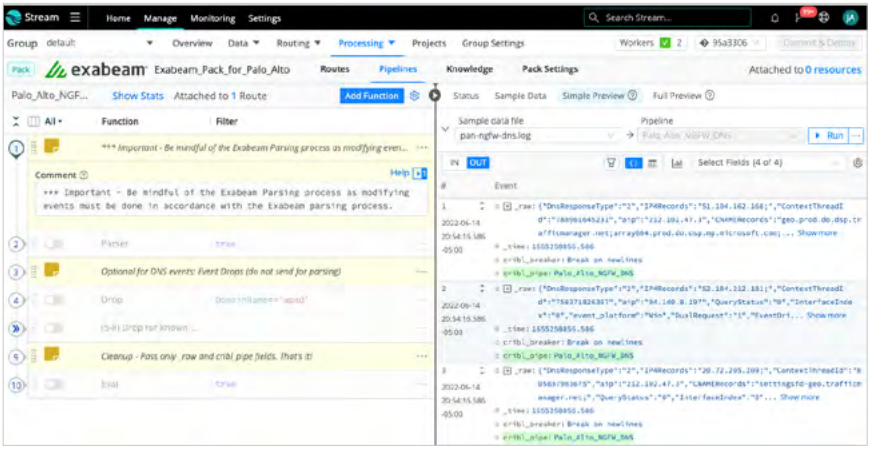
LESS DATA, MORE INFORMATION — WITH NO COMPROMISE

Built into Cribl's same Exabeam Pack for Palo Alto are options geared to solving that problem: What we generally care about in the DNS data are the anomalies within the giant dataset. The data we seek is typically in domains seen for the first time, newly registered domains, and domains generated by a Domain Generating Algorithm (DGA).

You can use Cribl Stream to send a drastically reduced DNS dataset to your security analytics platforms. At the same time, you can divert the full-fidelity feed to low-cost object storage or a data lake for long-term retention – whether that's Cribl Lake, or a lake on any other host.

UNPACKING THE DNS PIPELINE

To see how to accomplish this massive event reduction, let's look at the DNS Pipeline from the same Exabeam Pack for Palo Alto. If you're following along: From within your Worker Group, select **Processing** > **Packs** > **Exabeam Pack for Palo Alto**. Then, from the Pack's own submenu, select the **Pipelines** tab, where we can select `Palo_Alto_NGFW_DNS` to examine its configuration, shown below.



Palo Alto DNS Pipeline, from Cribl Pack – Functions and comments

There, the first (yellow or shaded) comment reminds us to proceed with caution, to make sure we don't violate Exabeam parsing conventions. This is why (if you look at the toggles on the left) this Pipeline comes out of the Pack with all Functions disabled by default. This ensures that we're simply passing the original events directly to Exabeam, as a bump in the wire. From this baseline, you can now choose which use cases you'd like to enable for the event drops you need.

SAMPLING PIPELINES' EFFECTS

Notice that in the **Simple Preview** pane on the right, we've preloaded a `pan-ngfw-dns.log` sample data file. By previewing the Pipeline's changes to this sample data, we can quantify individual Pipeline Functions' transformations before we commit any changes into production.
Cribl Packs include similar sample data corresponding to every Pipeline. You can also use Cribl Stream's Capture feature to store your own sample data files, to rerun data and test your Pipeline's transformations as you build it out.

PARSE, DROP, EVAL

In the same `Palo_Alto_NGFW_DNS` Pipeline shown above, the optional Drop Functions – which drop uninteresting events – will usually be preceded by a Parser Function that captures relevant context from the original event. Also, the Drops will usually be *followed* by an Eval Function, to clean up this context, so that we're left with the original `_raw` field.

If you enable the Parser Function, it will extract the request's domain name resolution into a field called **DomainName**.

The first Drop Function drops Web Proxy Auto-Discovery (WPAD)–related events. WPAD is a browser proxy discovery process that most organizations will elect to drop, because it is uninteresting for security investigations. As you can see, to drop these in Stream, we've just configured a Drop Function to match events that reveal `DomainName=='wpad'`.

The second *group* (collapsed) of four Drop Functions drops *well-known* domains, by rank.

VARIATIONS

Many more interesting security-related use cases are built into the Exabeam Pack for Palo Alto, as well as into other Exabeam-specific Packs available in the Cribl Packs Dispensary.

–

WHAT'S NEXT?

Cribl Stream's capabilities are always evolving, and you might see some differences from the UI screenshots and instructions provided in this edition. Luckily, Cribl Packs keep evolving, too – expect the Pack illustrated here to be progressively updated with new capabilities. (If you've customized the Pack's Pipelines, Stream ensures that importing an upgraded Pack won't overwrite them. See docs.cribl.io/stream/packs/#upgrading).

SECTION 02

Enrich, Observe, Analyze, Respond

03 | Better Threat Hunting with Cribl Search Data Enrichment

Crowdsourcing API Endpoints and Lookups to Thwart Evildoers

by Raanan Dagan, Cribl

PROBLEM

To detect anomalies most effectively, SIEM (security information and event management) systems need log events to arrive enriched with ample, and up-to-date, contextual information. Maintaining and updating this enrichment data can be cumbersome.

SOLUTION

Cribl Search facilitates enrichment by enabling you to uncover and analyze data at rest, directly from its source. Cribl Search can easily reach out and query data already collected in Amazon S3 buckets (or S3-compatible stores), Amazon Security Lake, Azure Blob Storage, Cribl Lake, Google Cloud Storage, and more.

Cribl Search's `externaldata` feature and lookup operators help you keep your lookup table up to date to best serve your needs. This means that you can run concurrent searches, using that same up-to-date lookup table, with minimal effort.

WHY CONSIDER THIS APPROACH?

By searching data where it lives, Cribl Search can dramatically speed up your search process by avoiding the need to move data before analyzing it. Search-in-place is useful in keeping threat intelligence current, and can also accelerate incident investigations and response.

HANDS-ON RESOURCES

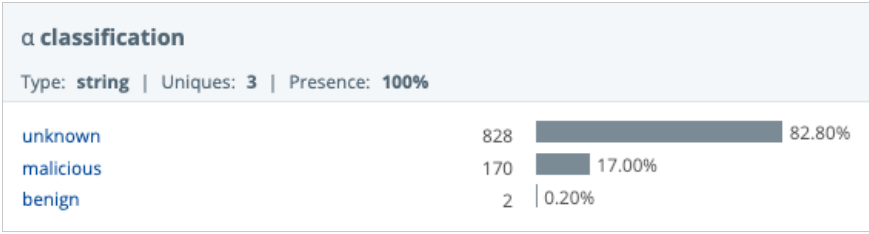
To apply the integration laid out here, you'll need a GreyNoise account and API key.

To access Cribl Search, sign up for a free Cribl.Cloud account at `cribl.io/solved`. Search will be ready to use immediately – start with its sample datasets or import your own working data. You'll also immediately have free instances of Cribl Lake, Cribl Stream, and Cribl Edge. When you need more search frequency or data capacity, you can readily upgrade to a paid account.

For a guided tutorial on Search basics, take our free **Cribl Search** Sandbox for a spin at `sandbox.cribl.io/course/overview-search`.

SCENARIO

In this example, Cribl Search uses a local lookup to enrich VPC Flow Logs with security classification data from the GreyNoise threat intelligence APIs. Search isolates events and IPs that GreyNoise classified as `malicious`, and sends them onward to a SIEM and security analytics tool.



Isolating events classified as malicious

WHAT WE'LL DO

We'll cover the following techniques:

- Ingest threat intelligence data to Cribl Search from an API endpoint.
- Use Cribl Search operators to isolate potential IOCs (indicators of compromise).
- Export the results to a lookup table.
- Configure Cribl Search to update the lookup table.
- Set the whole ingest/update cycle to repeat on a defined schedule.

NOTABLE SEARCH OPERATORS FOR ENRICHMENT

Let's dive into how this threat-hunting approach works. We'll use two main Cribl Search operators for enrichment, and we'll show you how to apply them in the steps below :

- The `externaldata` operator (`docs.cribl.io/search/externaldata`) fetches data from HTTP(S) URLs – including, as we'll apply it here, public APIs.
- The `lookup` operator (`docs.cribl.io/search/lookup`) retrieves fields from lookup tables, returning all rows by default.

VERIFYING GREYNOISE API ACCESS

We can first test the GreyNoise API, independent of Cribl Search. This `curl` command will validate that all is working:

```
curl --request GET \
--url
‘https://api.greynoise.io/v2/experimental/gnql?query=’last_
seen:1d’&size=1000 ‘ \
--header ‘accept: application/json’ \
--header ‘key: greynoise_enterprise_key’
```

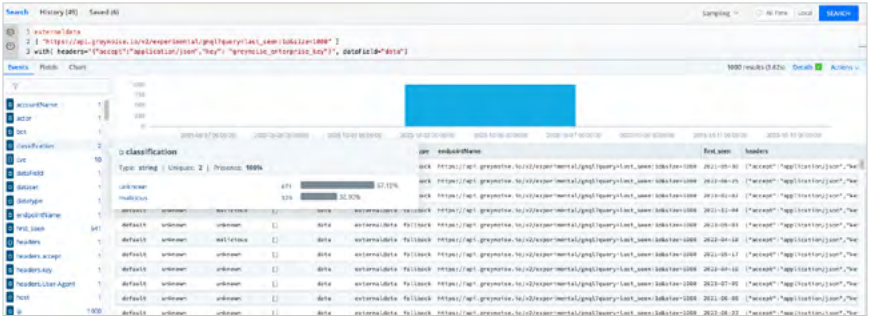
INGESTING THE ENRICHMENT DATA INTO CRIBL SEARCH

Once API access is verified, we can assemble the Cribl Search query below to pull fresh indicators from GreyNoise threat intelligence APIs, isolate potentially malicious IP addresses, and stage these indicators for lookup.

The first step is to pull in data from GreyNoise threat intelligence APIs, using Cribl Search’s `externaldata` operator. We can accomplish this with an initial query like this:

```
externaldata
[
“https://api.greynoise.io/v2/experimental/gnql?query=last_
seen:1d&size=1000” ]
with( headers=’{“accept”:”application/json”,”key”:
“greynoise_enterprise_key”}’, dataField=”data”)
```

Running that query gives us results like those shown below. Next, we're interested in the GreyNoise `classification` field, whose distribution we've zoomed up here from the left nav:



GreyNoise data ingested via Cribl Search's externaldata operator

ISOLATING IOCS AND NOURISHING A LOOKUP TABLE

Now that we've examined the raw GreyNoise ingest, we can build out the query to filter IP addresses of interest, and to store them to a lookup table. We'll add the following steps:

- Use the project operator to extract only the IP (`ip`) and classification fields.
- Use the `where` operator to consider only `malicious` IPs.
- Export those results to a local lookup table named `badguys.csv`.
- Use the `fieldMapping` option to map fields to lookup table columns.
- Use the `mode=overwrite` option to update the lookup file.

The full Cribl Search query looks like this:

```
externaldata
[
"https://api.greynoise.io/v2/experimental/gnql?query=last_
seen:1d&size=1000" ]
with( headers='{"accept":"application/json","key":
"greynoise_enterprise_key"}', dataField="data")
| project classification, ip | where classification=="malicious"
| export description="Bad Guys"
fieldMapping="classification:ip_status,ip:source_address" to lookup badguys
mode=overwrite
```

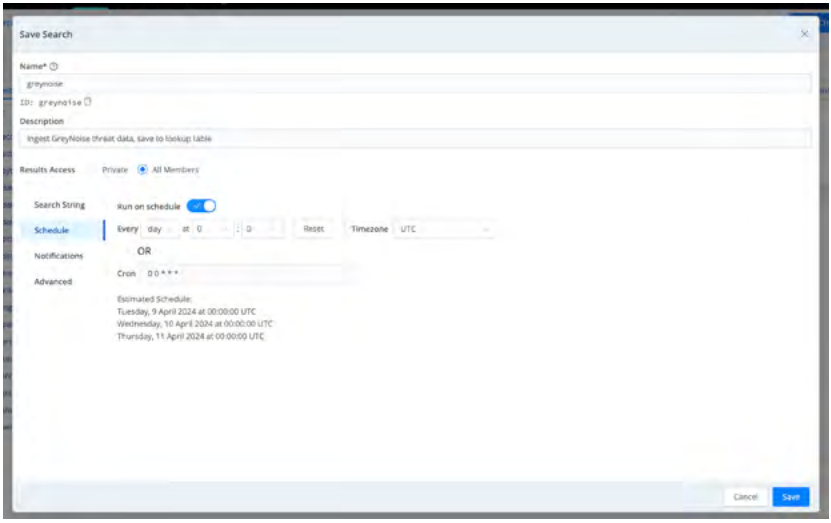
Here's how this looks in the Cribl Search UI:

Updating the lookup file

ONCE IS NOT ENOUGH

Once we're satisfied with our query and its initial results, we can save it by selecting **Actions** > **Save**.

From the resulting **Save Search** modal, we can select the **Schedule** left tab to schedule the query to run automatically on a reasonable interval, like once a day.

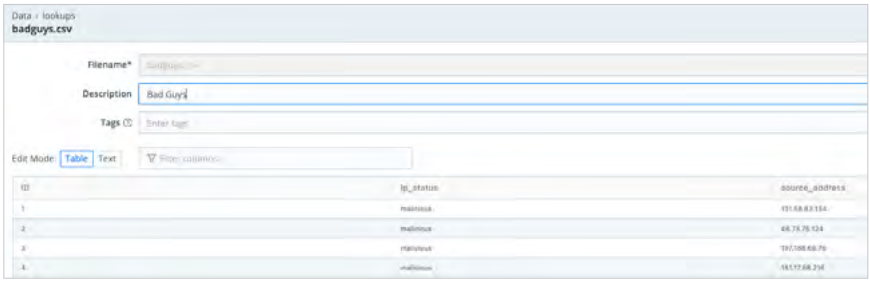


Scheduling the query to run periodically

LOOKUP TABLE NOW ISOLATES ONLY THE BADDIES

With all this in place, any query against this lookup file will now find only the malicious IPs reported in the last 24 hours.

To verify the table's contents, we can select **Data** > **Lookups** from Cribl Search's top nav, and then click on the new or updated lookup file. Notice the column names mapped from GreyNoise: `ip_status`, `source_address`.



Lookup file's structure

ENRICHING LOG DATA WITH IOCS

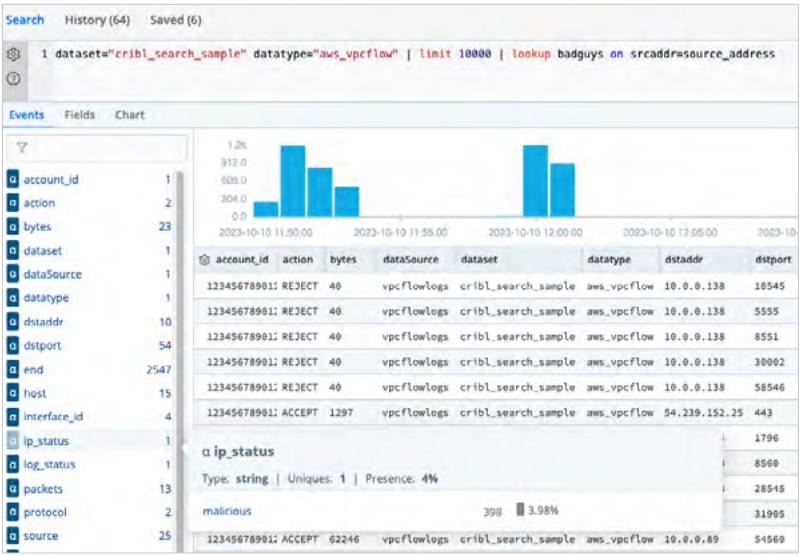
Now that we know we can rely on this periodically refreshed lookup table to identify recently reported malicious IPs, we can enrich VPC Flow Logs data from our `badguys.csv` lookup.

Why not start with sample data? Cribl Search's out-of-the-box `cribl_search_sample` dataset includes AWS S3 VPC Flow Logs data. This data contains source address IPs (`srcaddr` field) that might prove to be malicious.

Here's a query that isolates this datatype, and maps its `srcaddr` field to the lookup table's `source_address` column:

```
dataset="cribl_search_sample" datatype="aws_vpcflow" | limit 10000
| lookup badguys on srcaddr=source_address
```

After running this search, we can check the `ip_status` field from the left nav for `malicious` values.

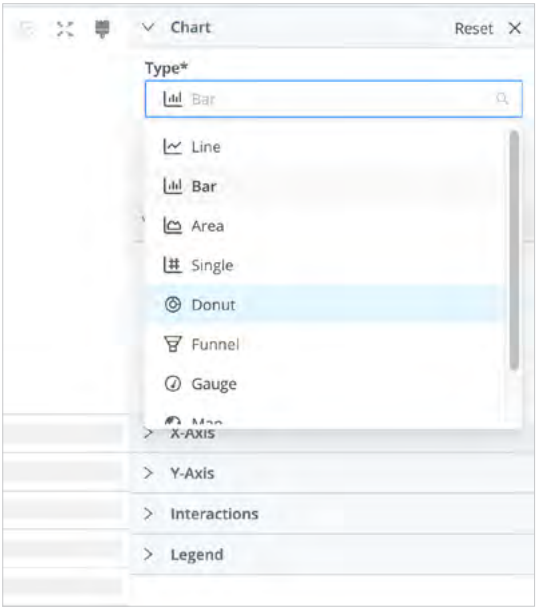


Isolating malicious events

To take this a couple of steps further, we can tune the query to return only the `malicious` hits, and to summarize these by destination. The full query now looks like this:

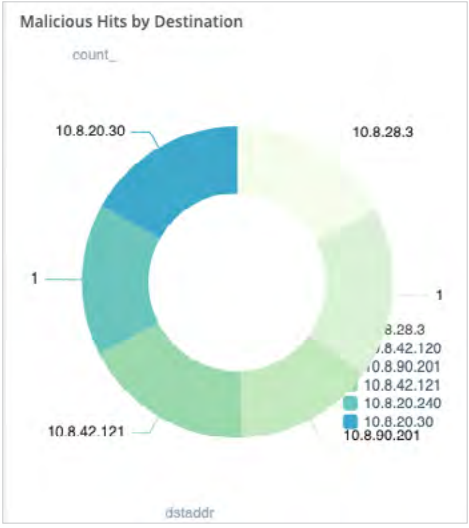
```
dataset="cribl_search_sample" datatype="aws_vpcflow" | limit 10000
| lookup badguys on srcaddr=source_address | where ip_status=="malicious"
| summarize count() by dstaddr
```

After running this, we can visualize the data's distribution more clearly. On Cribl Search's **Chart** tab, we can click the **Format** (paintbrush) button, then set the **Type** to a **Donut** chart:



Selecting a different chart type

That will enable us to see the distribution of destinations in a format like this:



Malicious hits only, summarized by destination

FORWARDING THE DATA

Finally, we can send the events and IPs that GreyNoise classified as `malicious` to our SIEM and security analytics tool. Cribl Search has both narrowed and enriched the data we're sending through. So we're limiting downstream analysis (and license usage) to high-value events only.

The `send` operator, by default (when invoked with no parameters), forwards Cribl Search results to a Cribl Stream HTTP Source. This gets the data into Stream, where you can route the results to your SIEM service of choice. Here's how to append `send` to a Search query:

```
dataset="cribl_search_sample" datatype="aws_vpcflow" | limit 10000
| lookup badguys on srcaddr=source_address
| where ip_status=="malicious" | send
```

VARIATIONS

We've seen how Cribl Search's `externaldata` and `lookup` operators enable you to automatically query a useful API to keep a lookup table of threat indicators up-to-date. This, in turn, enables you to run multiple searches against current log data and indicators, with minimal effort.

The use case we explored here looks up VPC Flow Logs log data against GreyNoise indicators.

This demonstration is representative of many data types and threat intelligence sources, and provides an easy-to-replicate pattern for your own needs.

WHAT'S NEXT?

For details on the Cribl Search and other resources we've applied here, see our online documentation:

- `externaldata` operator (docs.cribl.io/search/externaldata).
- `lookup` operator (docs.cribl.io/search/lookup).
- `project` operator (docs.cribl.io/search/project).
- `where` operator (docs.cribl.io/search/where).
- `export` operator (docs.cribl.io/search/export).
- `send` operator (docs.cribl.io/search/send).
- Other operators (docs.cribl.io/search/operators).
- Scheduled Searches (docs.cribl.io/search/scheduled-searches).

–

To learn more about GreyNoise threat intelligence APIs, see GreyNoise's own documentation at docs.greynoise.io.

04 | If You Can't Search It, Does It Even Exist?

Expediting Investigation and Incident Response with Cribl Search

by Alexandria Crusco, Steven Litras, Ana Monroy Zolano, and Aaron Thummel, Cribl Security Team

PROBLEM

TL;DR: Attacks are increasing, data is increasing, and separating valid attacks from false positives has become exhausting.

The fight against attackers grows in intensity each day for IT/Security incident first responders, defenders, and employees of organizations. The chances of falling victim to an undetected attack are rising, leaving responders to fight what feels like a losing battle.

According to IBM's *Cost of a Data Breach Report 2023*, data breaches' average cost reached an all-time high in that year of U.S. $4.45 million. This had increased 2.3% from the 2022 average cost, and increased 15.3% from the 2020 average.

As Security professionals, our tooling leaves us with alert fatigue and a false sense of security: We have *all this data*, so we must be catching attacks, *right*?

SOLUTION

Cribl Search offers a powerful response to mountains of data, relentless attackers, and security tools that do not always fit every incident scenario.

Especially when used in combination with Cribl Stream and Cribl Lake, Search delivers the next generation of incident investigation. Ultimately, it enables defenders to accelerate their incident response processes, and to reduce attacks' potential impact on their organizations.

Through this chapter's incident scenario, we'll highlight applying these Cribl Search techniques:

1. How to build Search queries relevant to your environment (i.e., Search + your SIEM == true power).
2. How to build meaningful dashboards (i.e., Search == a powerful enrichment tool).
3. How to accelerate incident response (i.e., Search == the GOAT investigation tool).
4. Bonus: How to use Cribl Stream to accelerate Search's own performance (i.e., Search + Stream == besties together).

WHY CONSIDER THIS APPROACH?

Incident responders and/or security engineers can use these techniques to increase the speed at which they detect, respond, communicate, and remediate potential threats. This approach is powerful in combination with your organization's SIEM, or you can use it by itself.

Consider this approach if:

- You currently have an internal struggle with massive amounts of data. The data has subsumed what little time your resources have to triage and correlate data to identify potential threats or incidents.
- Incident investigation is not *simple* within the organization's current processes.
- You want to accelerate incident response, to reduce impacts on your organization.

The attacker scenario laid out below is assumed to be *medium* in complexity. Re-engineer this when considering more complex attack use cases.

This approach relies on the MITRE ATT&CK framework (attack.mitre.org), and will be easy to follow by anyone with a basic understanding of investigation methodology.

WHAT WE'LL DO

This scenario will reveal Cribl Search applications like detection building, custom dashboarding, enrichment, and searching in place. As you can see from this outline, you'll have to read all the way through to find out whether good triumphs over evil!

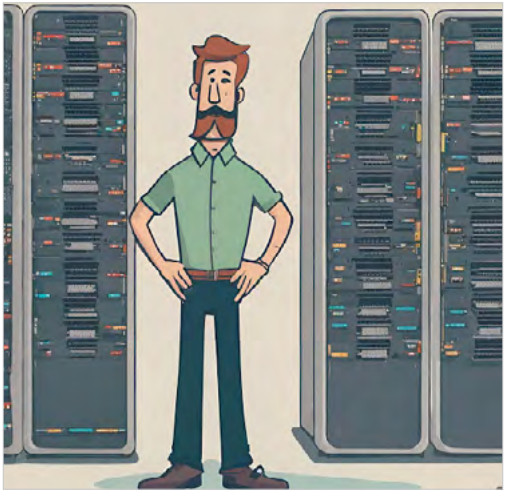
1. The Calm Before the Attack
2. The First Clue...
3. First Suspect: AWS
4. Next Suspect: Okta
5. The Final Suspect: Google
6. The End

THE CALM BEFORE THE ATTACK

"Detective" Steve Searchwell logged onto his trusty computer for another day of riveting security work. To be clear, he referred to *himself* as "Detective," even though no one else did, and was hoping the name caught on as he made a name for himself as the best security investigator in the industry.

Like most Security professionals, "Detective" Searchwell used a SIEM (specifically, Panther), in combination with Cribl Stream, to help ingest all the needed logs from his organization's many active applications and tools. This setup was working pretty well.

However, "Detective" Searchwell was struggling with the massive amount of alerts the SIEM generated each day. Luckily, Searchwell had configured all his needed datasets in Cribl Search the week prior. He was excited to start using Search for investigative work, in the hope of maturing the detections for his organization's environment, as well as potentially reducing alerts.



"Detective" Searchwell, in plainclothes guise

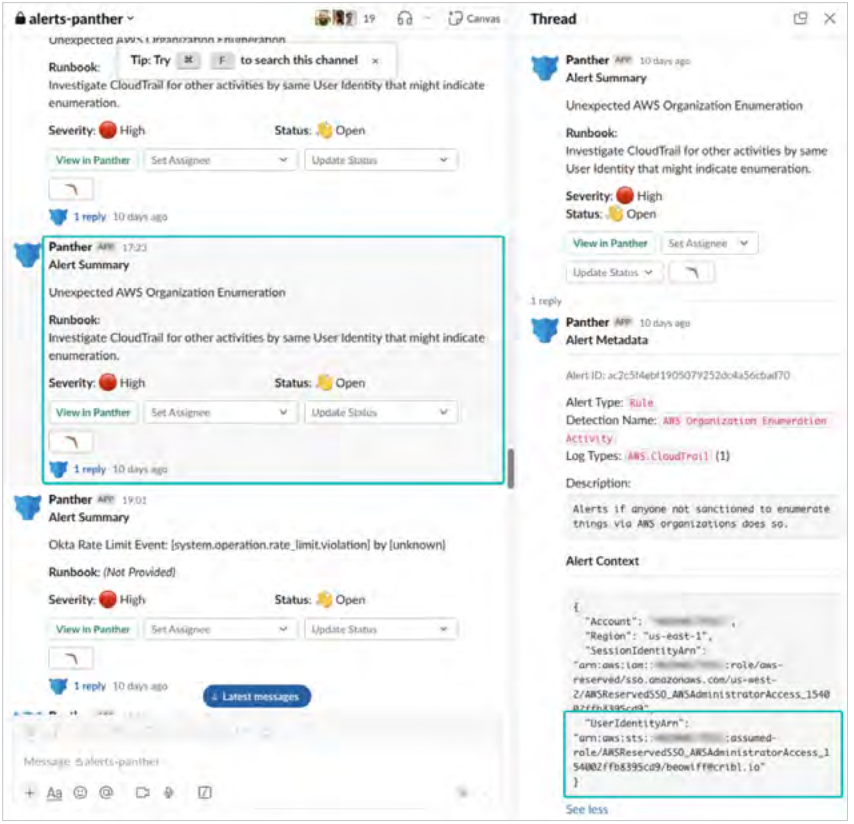
WHY DOES THIS MATTER?

Most Security professionals are similarly bogged down in alerts, noise, and potential rabbit holes. In this instance, Searchwell was able to configure datasets in Search to assist him in maturing his investigation process. This powerful combination of his SIEM and Cribl Search allowed Steve to put to the test the phrase "no one knows your environment better than your Security professionals".

And this is an important note for organizations: The Security personnel tasked to protect the environment every day are going to be the quickest to understand root causes during investigative scenarios. They need the tools to get to the root cause quicker.

THE FIRST CLUE…

Most security incidents start with a single notification, or seeing something strange in your environment. Searchwell was looking through the alerts in his SIEM, and saw a notification that could potentially be an indicator of compromise (IOC). It looked like some account was trying to explore the AWS environment with an enumeration effort:



Enumeration alerts from Panther SIEM

In this case, Searchwell has a detection on his SIEM that alerts him when someone makes certain AWS Organization calls. These calls could indicate that someone, specifically an admin named Ben Wiff, is trying to explore the Organization's structure – accounts and OUs (organizational units). This type of activity is not a common occurrence within Searchwell and Wiff's environment, so Searchwell puts on his detective hat and decides to investigate further.

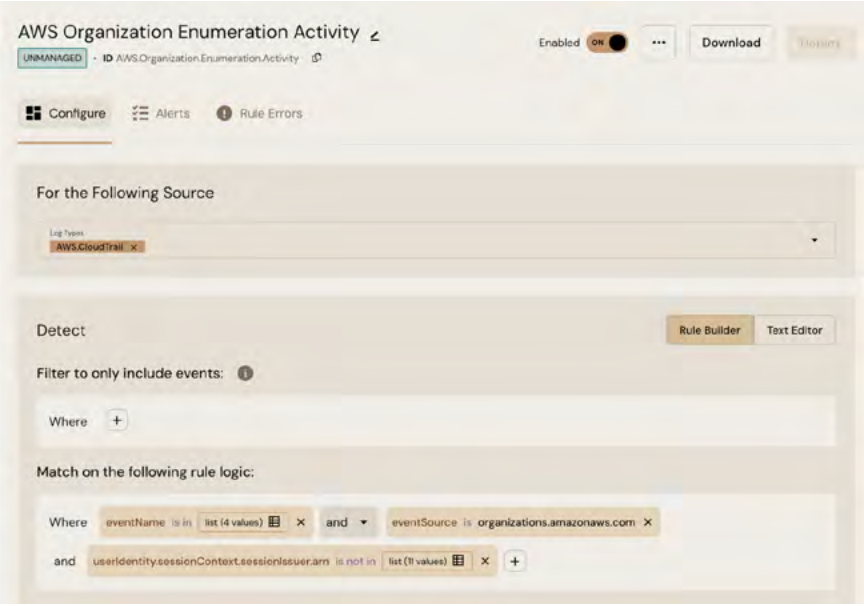
At this point, all we know is the alert. The alert is a start, but we need to look at other things to understand the context in which the activity occurred. Though Searchwell's company uses Panther as its SIEM, this could really be any SIEM, or even the Cribl Search product itself (assuming the company can build its own detections).

ELIMINATING FALSE POSITIVES

The true art of investigative work is to understand where the truth is – to pull the thread and ask the questions that reveal the root cause of an investigation. As a Security professional, Searchwell knows that there is a chance that this activity might be legitimate in nature.

Searchwell begins by making a list of things his company does that would make this type of alert "legitimate." A number of processes regularly enumerate the AWS environment; these processes would do things like ensure that tagging is up to standard, maintain an active catalog, do maintenance, or even security scanning.

For example, the enumeration detection has logic that ignores detected enumeration from AWS roles that are expected to enumerate. As you can see at the bottom of the Panther screenshot below, 11 roles are in the "is not in" clause. This simply minimizes the noise, to help Searchwell pay attention only to anomalous behavior. It's entirely possible that Ben simply forgot that he's supposed to use one of those roles for that kind of work. But by having those roles as "exceptions," we minimize the noise that this alert could create, and only have to investigate the ones that deviate from policy.



Panther SIEM logic to ignore expected roles

UNDERSTANDING YOUR ENVIRONMENT

At this point, it's important for us to remember that Searchwell is a self-proclaimed "Detective" due to his in-depth understanding of his organization's environment. If you ever find yourself in Searchwell's shoes, it's best to remember the following: To be able to properly triage events in your environment, it's critical that you *understand* your environment.

In Searchwell's case, he could easily have written off this activity as "common." However, Searchwell hones in on this alert, because he's noted that an administrator is not using any of the 11 roles that legitimately make organization enumeration calls.

FIRST SUSPECT: AWS

Now that Searchwell has determined things are definitely amok, he must use some of the tools in his belt to glean more information about Ben Wiff's activity. At this point, Searchwell turns to his newly configured Cribl Search tool, and begins to ask himself questions about the enumeration alert. What would he like his first query in Search to be?

Searchwell knows that there are many different reasons why an attacker might want to enumerate an AWS environment. For the sake of being thorough, he begins to write down his hypothesis. If this is an actual attack, an attacker might enumerate an environment to list out OUs and accounts, to get the lay of the land or the attack surface (like our alert example above).

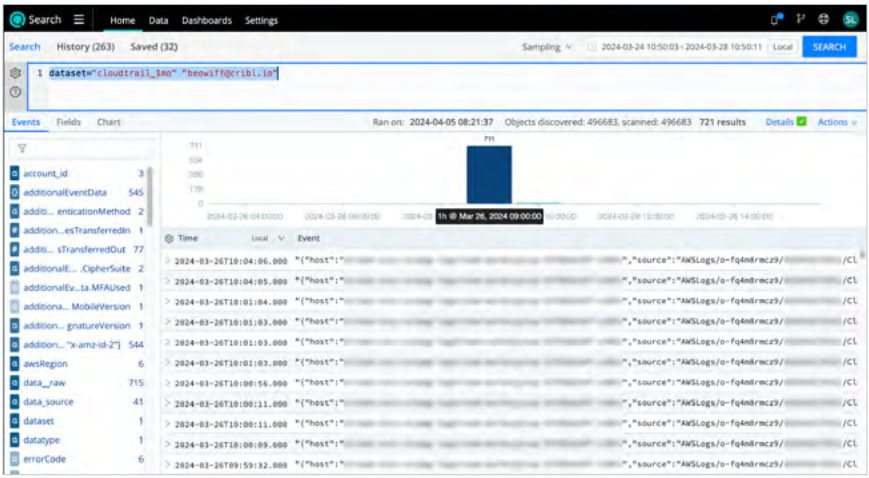
Searchwell continues pondering the enumeration and finds himself wondering, "Now, what would an attacker do next?" This thought leads to Searchwell's next step: After the enumeration, did Ben Wiff's account attempt to connect to each account for the purposes of testing the user's access? This question leads us into Searchwell's first Search query.

Sidenote: Start with What You Know

Cribl Search is a powerful tool, and there are many times that understanding what you are looking for can help you get the exact answer you need through your Search query. There are times, however, when you might not have as educated a guess as we are outlining here. In which case, you could simplify your search, by starting with what you know instead of with the hypothesis you want to test. This could be as simple as writing a Search query that goes something like this:

```
dataset="your_dataset" "beowiff@cribl.io"
```

In this instance, the dataset would be where your AWS CloudTrail logs are being stored. (This is assuming that you've added these logs' location to your Cribl Search instance as a dataset. See the end of this chapter for links to Search guides that cover this and other techniques.) Searchwell ran this query, and got a lot of results:



Initial search results from CloudTrail logs

OH, WE BE QUERYING, WE BE QUERYING

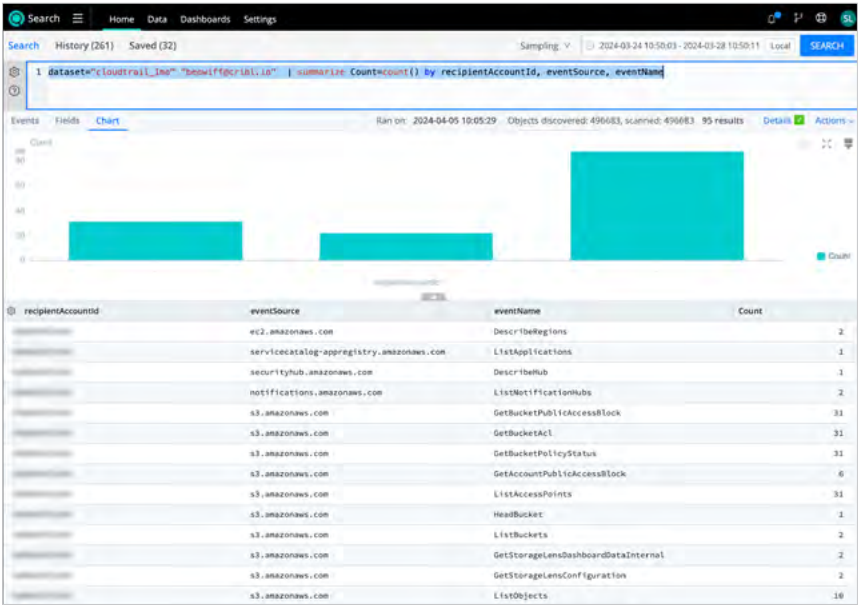
Searchwell started searching across CloudTrail logs to examine the suspected "attacker's" actions on AWS in greater depth. What had the user's account been doing, in what AWS accounts? The easiest way to do this was to summarize the data by a few fields:

1. `recipientAccountId` – the account where the action happened.
2. `eventSource` – the AWS service invoked.
3. `eventName` – the action taken.

To do this, Searchwell appended the following to his query:

```
| summarize Count=count() by recipientAccountId, eventSource, eventName
```

And the results were a bit easier to work through:



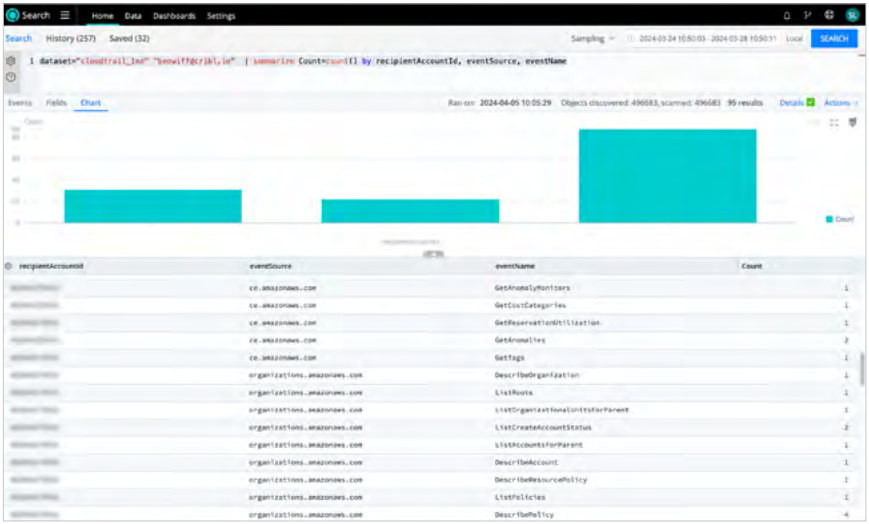
Search results after summarization

As Searchwell is reading through the results, he starts to have a suspicious feeling. Ben Wiff is a highly technical employee of his organization, with a very solid understanding of the environment. But Searchwell sees that Ben Wiff was doing some serious clicking through S3, revealed by multiple counts on `eventNames` (actions) like: `GetBucketAcL`, `GetBucketPublicAccessBlock`, and more. Why?

Searchwell's instincts are starting to kick in. This behavior is more typical of an attacker attempting to map an attack surface, than of Ben Wiff's normal day-to-day activity. Alas, Searchwell must not get ahead of himself. He scrolls further down through the results.

As he enumerates the same query, Searchwell notices that the actions taken by `beowiff@cribl.io`'s account were a bit more widespread than just listing S3 buckets' attributes. Our potential attacker had logged into all of the accounts that Ben has access to, and had clearly explored each.

A couple of things jumped out to Searchwell as potential problems: `DescribeOrganization` and `ListRoots` are very alarming enumeration methods of the most powerful AWS accounts.



Summarized search results, showing DescribeOrganization and ListRoots actions

Searchwell pauses. It is clear that he must go figure out if there was foul play on Ben Wiff's account, so he heads to his next suspect to build his case.

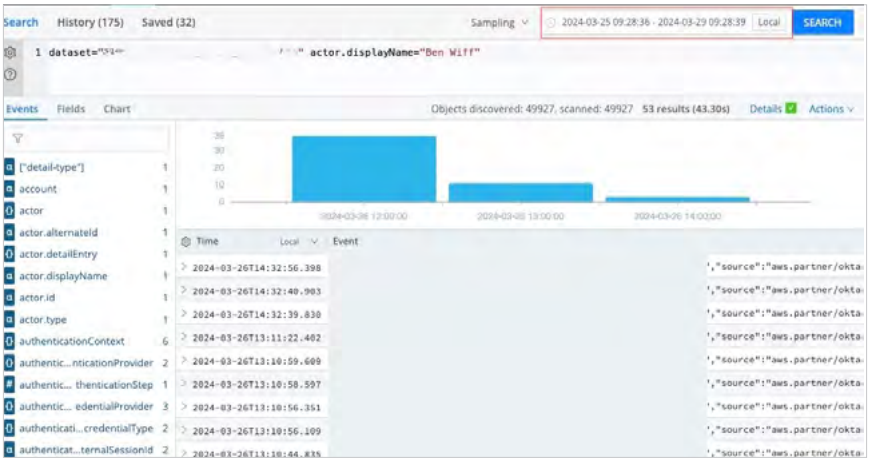
NEXT SUSPECT: OKTA

Searchwell clearly now understands the magnitude of the investigation he is analyzing. Most likely, an attacker was able to compromise the user's account. Upon compromise, the attacker then went through and identified valuable information about the environment. Searchwell knows that he now must be able to confirm the attacker's point of entrance, so he turns his attention to his organization's identity provider, Okta.

There are many ways to confirm whether a user's Okta account has been compromised. However, the first thing that Searchwell decides to do is look for failed login attempts. The first query he writes in Cribl Search looks like this:

```
dataset="your_dataset_okta_logs" actor.displayName="Ben Wiff"
```

Searchwell sets the appropriate time range in Search's time picker, and hits **Search**. This returns a lot of information. Searchwell combs through the 53 results, which look something like this:



Initially retrieved Okta logs

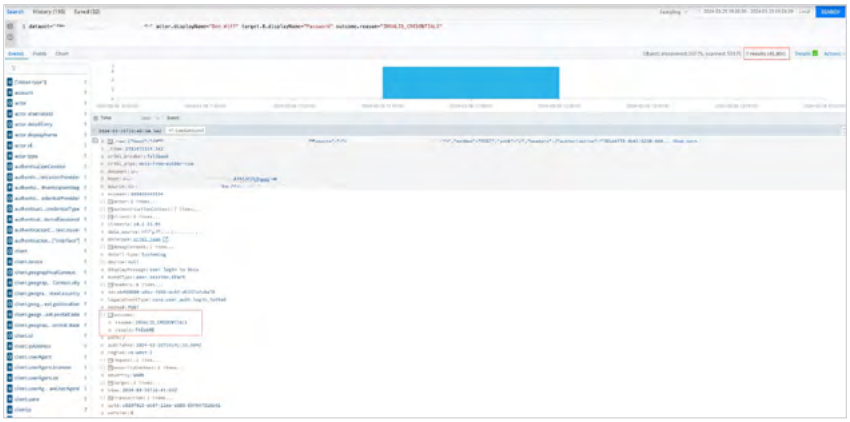
As he is reviewing the results, he identifies his next point of interest: the `target display name` field. Searchwell feels like he is on to something, and starts to refine his Search query so it looks something like this:

```
dataset="your_dataset_okta_logs" actor.displayName="Ben Wiff"
target.0.displayName="Password"
```

After running this search with the same time range, Searchwell begins to see where things went awry. This search narrows down his results to 11, and Searchwell finally finds his sought-after needle in the haystack. He hones in on the field `outcome reason`, and refines his search once more so that it looks something like this:

```
dataset="your_dataset_okta_logs" actor.displayName="Ben Wiff"
target.0.displayName="Password" outcome.reason="INVALID_CREDENTIALS"
```

Yes! Searchwell finally is able to see the seven invalid logins from the "attacker," which are then followed by a successful authentication.



Failed login event

This now confirms for Searchwell that the attacker most likely gained access via Okta, which leads to the next step in his investigation. He must see what other things the attacker did upon gaining access to this user's account, and then build a foolproof case against the initial access point.

Sidenote: A Viable Attack Path, or Someone Who Can't Type Their Password?

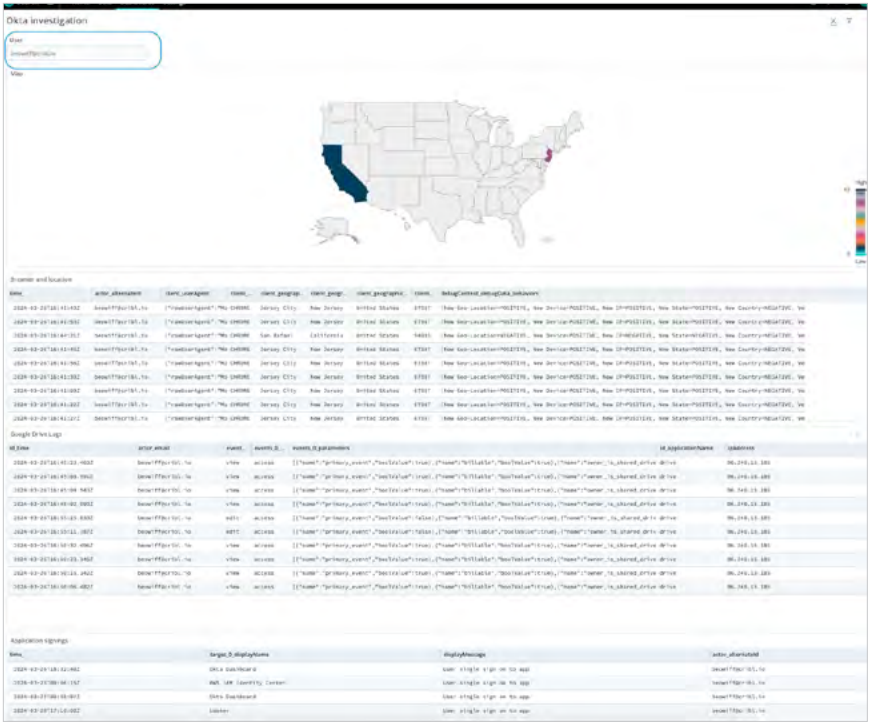
There are plenty of reasons for seven unsuccessful login attempts. Maybe someone hasn't had their coffee yet, and types their password wrong seven times. That's possible. However, when we are looking at the context of the scenario we are running through, this is a very interesting thread to pull on.

Incident investigation is about finding *viable* threads to investigate further. As the "best" Detective, Searchwell would not just stop at his discovery of the failed attempts on AWS followed by a successful authentication, and call the investigation "closed." He would now take these things and look at the next step – more clues to confirm or dismiss his hypothesis.

THE HOLISTIC VIEW – OKTA

With a good lead to follow, Searchwell now wants to save some of his more in-depth queries into a dashboard. This dashboard should help solve the case and determine the root cause of this investigation.

After some organizing and analysis of Okta logs, Searchwell will end up with this dashboard. (We'll show you below how he put it together.) It displays a map and three tables.



"Detective" Searchwell's very helpful dashboard

At a glance, Searchwell is able to use this Cribl Search dashboard to paint a picture of the initial compromise through the map. He sees simultaneous logins in different states, better known as "geographically improbable" alerts.

Simultaneously, Searchwell wants the option to disable whichever other applications the "attacker" browsed to after the initial compromise. The second table below the map shows events on Google Drive, which we'll get to below in (spoiler alert:) "The Final Suspect: Google."

The last table shows that the "attacker" looked across the Okta Dashboard, the AWS Identity Center, and Looker – in all, actions better known as "application sign-ons." All of this information is very valuable to Searchwell for a few reasons:

- Searchwell can continue to narrow down his searches to focus on pertinent information about what actions the attacker might have taken.
- Searchwell can communicate with other members of his team, or his superiors, about the scope of the incident.

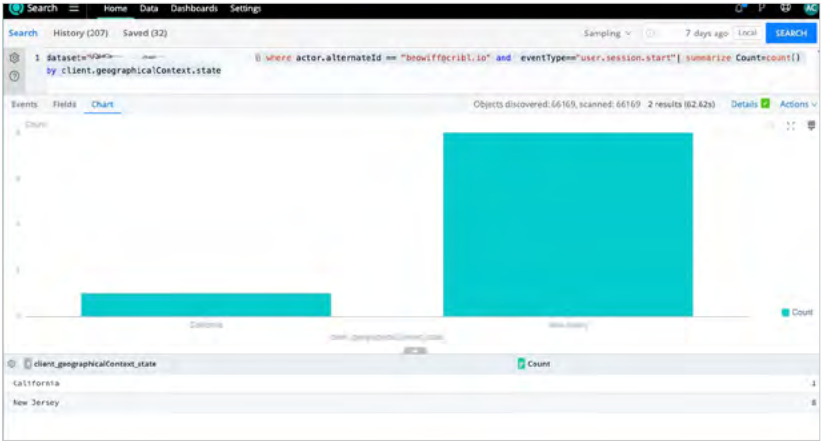
GEOGRAPHIC QUERY RECIPE

But first, let's take a deeper dive into how to write a "geographically improbable" query and then save it as a map, as displayed above. The query looks like this:

```
dataset="your_dataset_okta_logs"| where actor.alternateId == "beowiff@cribl.io" and eventType=="user.session.start"| summarize Count=count() by client.geographicalContext.state
```

This query essentially looks for the actor ID `beowiff@cribl.io`, and for sessions' event type, while summarizing the count for different geographic regions – specifically, by state. Put simply: Tell us how many times a session was started in different states. This allows Searchwell to understand whether there is any anomalous activity.

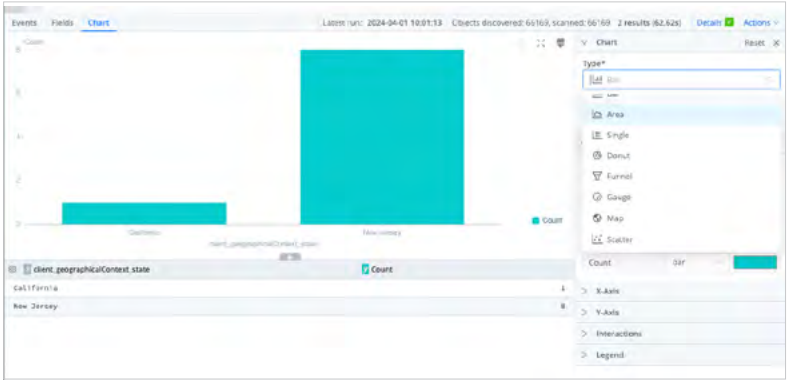
This is especially interesting for Searchwell, as it confirms that this user's account was compromised. The several failed login attempts were followed by a session starting in a location that is not typical for this user. When you hit **Search** on this query, with the correct time range input, you get results like those shown below.



Suspiciously bicoastal logins

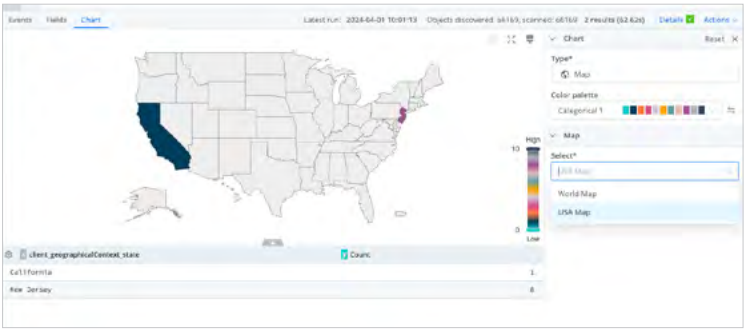
DEFINING A MAP

On his way to the dashboard we showed above: Now that Searchwell has his query, he wants to save this to a dashboard. But before doing this, Searchwell thinks to himself, "How nice would it be to visualize this as the U.S. map?" Well, Searchwell is in luck. To do this, he clicks on the **Format** (paintbrush) button in the chart's top right corner, and then opens the **Type** drop-down to get these choices:



Reformatting a bar chart to a different visualization type

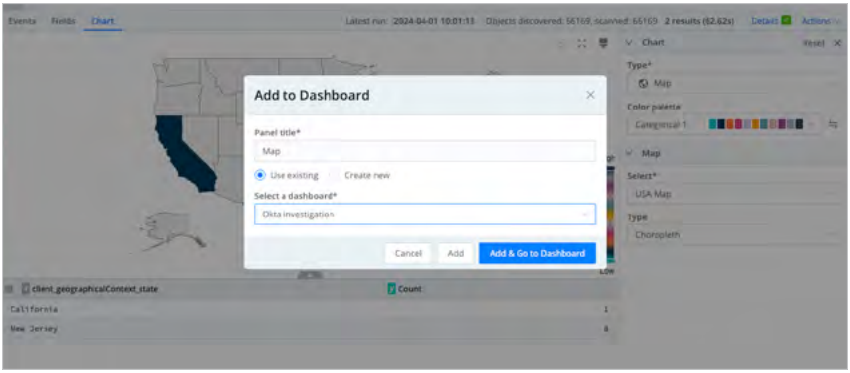
This allows Searchwell to select a **Map** chart type for his Search – and, eventually, for his dashboard. So now he has a map, but it's actually rendering as the entire globe, which is not as relevant for Searchwell. So he selects the **USA Map** subtype, and his chart is now perfect.



U.S. map, where something is rotten in the state of abnormal

FROM VISUALIZATION TO DASHBOARD

Now that he has the visualization he wants, Searchwell will add it to his existing **Okta Investigation** dashboard. In Cribl Search's top right corner, he selects the **Actions** menu and then **Add to Dashboard**. Next, in this modal, Searchwell can title his map and select its select its destination dashboard.



Adding map visualization to a dashboard

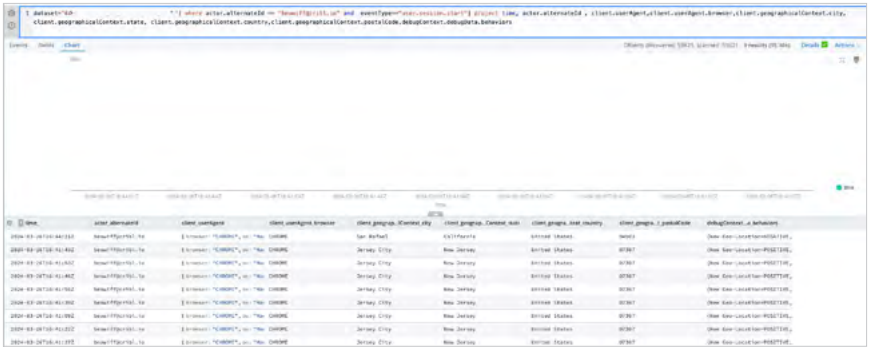
He clicks **Add & Go to Dashboard**. Yay! For each of the subsequent queries we'll explore below, Searchwell will follow this same process to add his relevant queries to his **Okta Investigation** dashboard.

MOAR GEOGRAPHIC DETAILS, PLEASE

The next query that Searchwell runs is to render more detailed information about the geographically improbable logs. Diving a bit deeper, Searchwell wants to understand the following parameters: time, actor ID, user agent, browser, city, state, postal code, and any debug behaviors. He uses a query like this:

```
dataset="your_dataset_okta_logs"| where actor.alternateId == "beowiff@
cribl.io" and eventType=="user.session.start"| project time, actor.
alternateId, client.userAgent,
client.userAgent.browser, client.geographicalContext.city,
client.ge ographicalContext.state, client.geographicalContext.country,
client geographicalContext postalCode,
debugConte xt.debugData.behaviors
```

And this query renders results like those shown below. Searchwell is fine with a table for this visualization, so he again selects **Actions** > **Add to Dashboard**, and moves on.



Search returns detailed time, geographic, and technical data

Searchwell now feels confident that he has collected all the information he would need to confirm and report the attacker's initial access point in this scenario. Interestingly, while he was going through these logs, he discovered his final suspect: Google.

Sidenote: How to Build a Dashboard?

Search is an exploratory tool. As you start searching, you usually end up evolving your searches as you go – adding here a clause, there a clause (everywhere a clause clause). Building a dashboard is as simple as the **Add to Dashboard** flow you saw above. If you don't have an existing dashboard, the modal offers a **Create new** button to create and name your first one.

Even better, when you want to have a dynamic dashboard – like the ones we're using – you can edit the dashboard and select **Add Input** for each desired enrichment. For our AWS dashboard, we added a `user` text input, and a time picker. Then we edited each of the panel's searches to use those inputs.

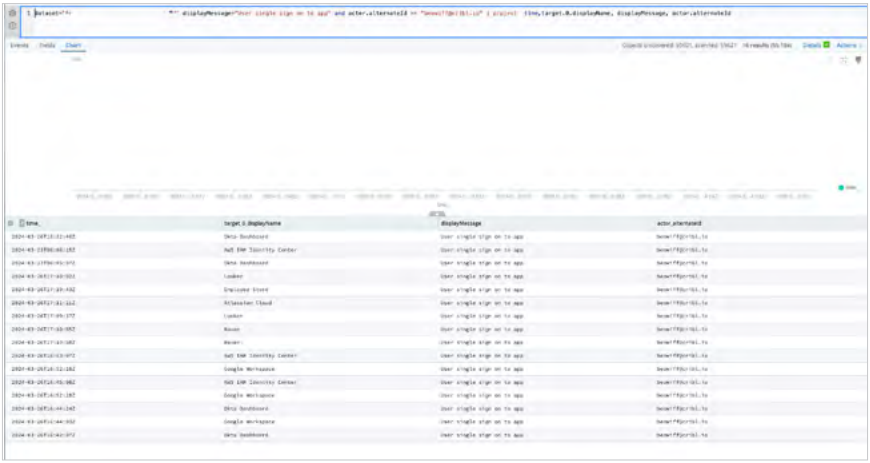
For details on how to do this, see Cribl's *Editing a Dashboard* documentation, linked at the end of this chapter.

THE FINAL SUSPECT: GOOGLE

While exploring Okta logs, Searchwell noticed that the attacker had enumerated a number of applications within the compromised user's Okta Dashboard. Prior to picking which application Searchwell would explore next, he wanted to be able to see **all** the logged-in applications during the defined period. So Searchwell wrote a query of this form and added it to his Search dashboard:

```
dataset="your_dataset_okta_logs" displayMessage="User single sign on to app" and actor.alternateId == "beowiff@cribl.io" | project time,target.0.displayName, displayMessage, actor.alternateId
```

This query rendered the following *interesting* information:

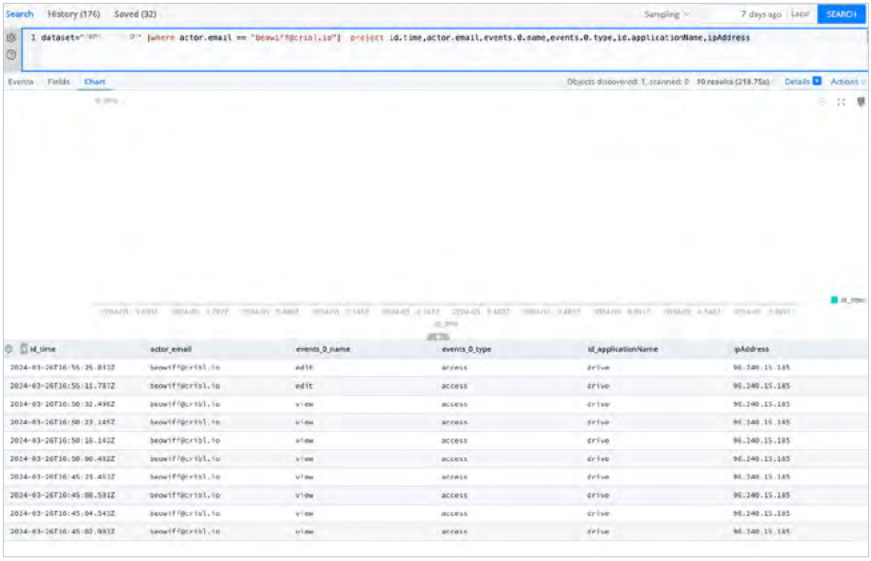


Search returns suspect account's application behavior

In this table, Searchwell hones in on the `Google Workspace` application authentications, as there is a potential for "sensitive" company data to be accessed there. Searchwell does not panic. He has gotten this far with Cribl Search, and is going to finish this job with one final query. What exactly did the attacker access or have access to in Google Drive? Searchwell writes a query like this:

```
dataset="your_dataset_google_logs" | where actor.email ==
"beowiff@cribl.io" | project id.time, actor email, events.0.name,
events.0.type, events.0.parameters, id.applicationName, ipAddress
```

This query returns the following results:



Search displays suspect accounts actions on Google Drive

Searchwell is able to see that the attacker was able to view and edit several different items within the organization's Google Drive. Searchwell continues to enumerate the log files and grabs the files' IDs.

With these IDs, he uses the Google Admin tool (or GAM) to look up the files. This shows Searchwell that the attacker has added an external email to files titled `IPO Plans` and `1Password Password and AWS Passwords`.

Result? The team disables the compromised user's Okta account, and begins cleanup work to ensure that there are no backdoors or remaining entry points for the attacker.

THE END

Searchwell's boss names him the best "Detective" in the SOC and commends him for using Cribl Search to identify this attacker. We must allow Searchwell to monologue his excellency:

My comrades, today, we were able to fight back against our adversary with a tool unlike any other. Search is the investigative tool of the future. If we had not set up Search last week, we might have missed the clues to solve this crime. The initial access via Okta, the AWS enumeration, and the access and exfiltration of sensitive company data was a doozy of an attack. However, we were able to mitigate this crime through our own investigative superpowers.

Tomorrow, my friends, there will be a new adversary. But as Theodore Roosevelt once said, "It is not the critic who counts; not the man who points out how the strong man stumbles, or where the doer of deeds could have done them better. The credit belongs to the man who is actually in the arena, whose face is marred by dust and sweat and blood; who strives valiantly; who errs, who comes short again and again, because there is no effort without error and shortcoming; but who does actually strive to do the deeds; who knows great enthusiasms, the great devotions; who spends himself in a worthy cause; who at the best knows

in the end the triumph of high achievement, and who at the worst, if he fails, at least fails while daring greatly, so that his place shall never be with those cold and timid souls who neither know victory nor defeat."

Through this incident, Searchwell wielded the power of Search in combination with alerts from his SIEM. Searchwell learned how to write powerful queries, often starting with what he knew and diving deeper, and finally discovering how to make dashboards to show a holistic picture of an attack.

Searchwell's honor, well-earned

VARIATION

Search is a powerful tool, and there are many ways to configure it and use it to investigate potential incidents. There might be times, however, when Search needs some extra juice. Let's see how to use the Parquet object storage format to increase Search speeds.

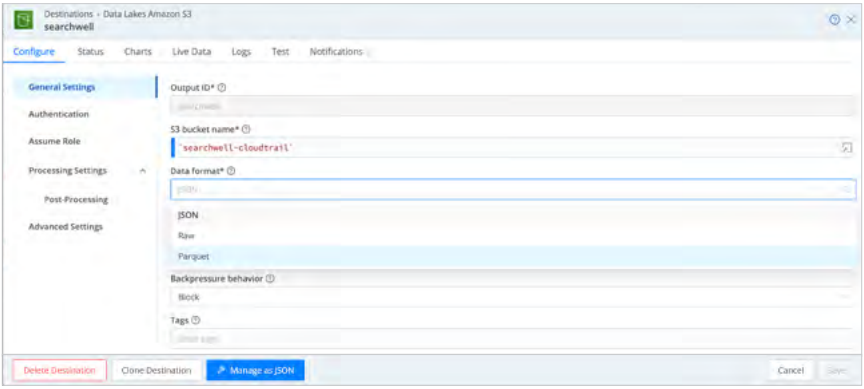
Put Some Juice in That Search — How to Increase Search Speed

Search does its searching across S3 buckets, using a partitioning scheme that's defined when writing to S3. Step One in making Search perform well is to get the partitioning scheme and data format right.

We use Cribl Stream to populate our S3 buckets, using an Amazon S3 Data Lake Destination that defines a partitioning scheme. In the corresponding Cribl Search dataset, we use this matching scheme as our **Bucket Path**:

```
<bucket name>/${*}-${_time:%Y}/${*}-${_time:%m}/${*}-${_time:%d}/${*}-${_time:%H}
```

And because CloudTrail is an *immense* dataset for an AWS environment of any reasonable size, we also want to optimize the data format. Using JSON is not performant on a dataset of this size. So when configuring our S3 Data Lake Destination in Stream, we instead set the output **Data format** to Parquet, which is significantly more performant.



Sending data to S3 in Parquet format, for faster Search retrieval

Cribl Search recognizes the Parquet format, and efficiently retrieves the structured data.

WHAT'S NEXT?

For a guided, live walk-through on configuring Cribl Search for yourself, take our free **Cribl Search** Sandbox for a spin at sandbox.cribl.io/course/overview-search.

If you aren't already using Cribl Search or other Cribl products, sign up for a free Cribl.Cloud account at cribl.io/solved. You'll be ready to start your first search immediately.

For details on the Cribl Search and other resources we've applied here, see our online documentation:

- Datasets and Dataset Providers (docs.cribl.io/search/#datasets).
- KQL/Kusto Query Language (docs.cribl.io/search/your-first-query).
- Editing a Dashboard (docs.cribl.io/search/editing-dashboards).
- Cribl Stream > S3 Data Lakes Destination (docs.cribl.io/stream/destinations-data-lake-s3).
- Cribl Lake (docs.cribl.io/lake).

5 | Streamline Windows Monitoring with Cribl Edge

Use Fleets to Centralize Management with Fewer Agents

by Ahmed Kira, Sri Kotikelapudi,
and Paul Dugas, Cribl

PROBLEM

Monitoring and administering a large number of servers and clients can require installing multiple agents. Parsing through individual log files per machine is laborious. Managing the agents, themselves, becomes a burden – requiring shelling into individual machines to update and restart agents.

This all adds up to redundant, tedious work. Practitioners find this burden manageable up to a few dozen or even hundreds of machines, but a fleet of thousands rapidly becomes almost impossible for a team to manage.

SOLUTION

Cribl Edge eliminates these tedious processes by providing a centralized, comprehensive and reliable platform for monitoring and analysis across your Windows machines. Edge collects and processes logs, metrics, and other data from Windows Servers (2016, 2019, or 2022). Edge Nodes can also run on “always-on” workstations with current Windows versions, sleep/hibernate options disabled, and an uninterrupted network connection. By using Edge to route this data to your favorite system of analysis or visualization, you can identify and investigate issues faster, and monitor applications more effectively.

WHY CONSIDER THIS APPROACH?

Notably, by running Cribl Edge on Windows domain controllers, you can collect security-related Windows Event Logs events from Windows Nodes – such as logins/logouts, failed logins, and password changes. Collecting this data centrally on the domain controller removes the need to instrument a fleet of laptops or other clients. Some other agents provide this facility, but Cribl Edge handles it particularly effectively.

Beyond the Security realm, Edge offers the same centralized collection, processing and routing for Windows performance metrics, and for logs and metrics generated by specific applications running on your Windows Servers. This allows you to troubleshoot application-specific issues and ensure their smooth operation.

HANDS-ON RESOURCES

Although Edge Nodes (instances) run on Windows hosts, the Leader that coordinates them is a Linux-only application. The fastest way to launch a Cribl Leader is to let Cribl run it for you: Sign up for a free Cribl.Cloud account at `cribl.io/solved`.

This will give you a Leader from which you can deploy Edge Nodes on your own systems. You'll also get free instances of Cribl Stream, Cribl Search, and Cribl Lake. When you need more capacity, you can readily upgrade to a paid account.

If you prefer to run a Leader on your own infrastructure, you can download an Edge (and Stream) Linux package at `cribl.io/download`. This chapter shows you how to install Edge onto Windows machines via script, but if you'd rather download a Windows `.msi` installer to run yourself, the same download page provides that option.

At the end of this chapter, you'll find links to individual topics of Cribl Edge online guidance. The starting point for all Edge documentation is `docs.cribl.io/edge`.

SCENARIO

This chapter presents a simple but realistic overview of a whole Cribl Edge life cycle for Windows monitoring: Deploy a Node, assign it to a Fleet, examine a host's state and logs, configure specific inbound integrations to monitor files and metrics, route data to downstream services, and execute and process PowerShell commands.

WHAT WE'LL DO

We'll walk you through the following techniques:

- Create a Windows-specific Fleet and Subfleet, to centrally monitor multiple machines that share the same configuration.
- Deploy Cribl Edge Nodes on Windows machines, by automatically generating and running a bootstrap script.
- Create Mapping Rulesets to set and switch Nodes' Fleet assignments.
- Explore a Windows Node to snapshot the host system's current state.
- Locate and collect custom Windows event logs and send them to your chosen visualization or analysis tool (or to any supported destination).
- Collect Windows metrics and send them to any supported destination.
- Collect application logs using the File Monitor Source.

As a bonus, we'll show how to use Cribl's Exec Source to execute PowerShell commands and process them in Cribl.

DEPLOY CRIBL EDGE ON WINDOWS

This section covers creating a Windows-specific Fleet, creating a logs-specific Subfleet, deploying ("bootstrapping") Windows Nodes, and assigning the Nodes to the Fleet and Subfleet.

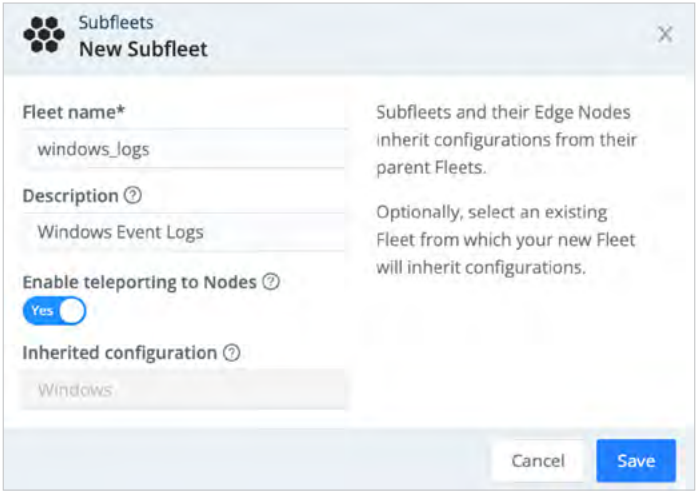
It's a snapshot of a broader, very powerful Cribl Edge feature that facilitates managing as many as several thousand hosts. To learn more about this feature, see “Variation: Using Fleets to Manage Many Diverse Hosts” below.

CREATE A WINDOWS-SPECIFIC FLEET

Fleets and their corresponding Subfleets allow you to author and manage shared configuration settings for a particular set of Edge Nodes. At a minimum, Cribl recommends separating Linux versus Windows machines into different Fleets per OS.

The first step is to configure a Fleet and name it `Windows`. (We've chosen this literal name to keep the demonstration simple. Fleet names must be unique across your deployment, so in production, select Fleet names that will convey each Fleet's purpose. For better practices on designing and naming your Fleets, see the references at the end of this chapter.)

To set up our mapping illustration later, let’s also create a Subfleet within the `Windows` Fleet. We're naming this Subfleet `windows_logs`.



Creating a Subfleet

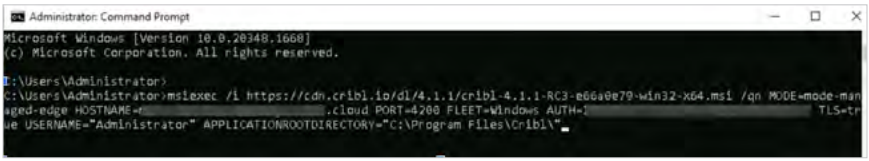
USE THE BOOTSTRAP SCRIPT TO ADD AN EDGE NODE

The Fleet provides logical structure on your Cribl Leader, whether running in Cribl.Cloud or your own infrastructure. The next step is to deploy Cribl Edge software on individual Windows machines.

From the Edge UI, you can readily generate a script to deploy these Edge Nodes. Below is a composite screenshot of generating this script for Windows.

Generating a bootstrap script for a Windows Edge Node

To add the Edge Node, paste the script into your Windows Command Prompt and execute it. (You must run the Command Prompt as an administrator for this to work.)



Pasting the script onto the command line

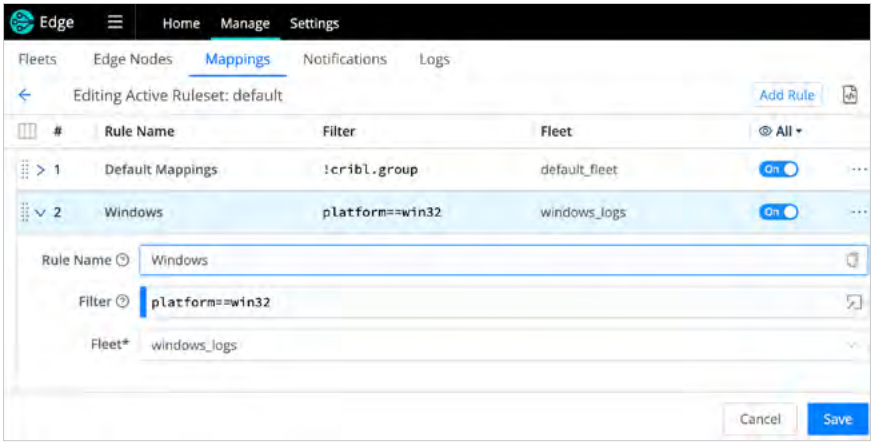
You'll need to do this on each target Windows machine, but only for initial setup there.

After deployment, you'll have the option to edit each target machine's `instance.yml` file to reassign the Edge Node to a different Fleet, to modify TLS settings on its communications with the Leader, or to customize other options. (Reference at the end of this chapter.)

USE A MAPPING RULESET TO SWITCH FLEETS

Next, we want to assign the newly deployed Windows Edge Node to the `windows_logs` Subfleet that we created for it. We can do this efficiently within Edge's Mapping Ruleset – where we can set one rule for all matching Nodes.

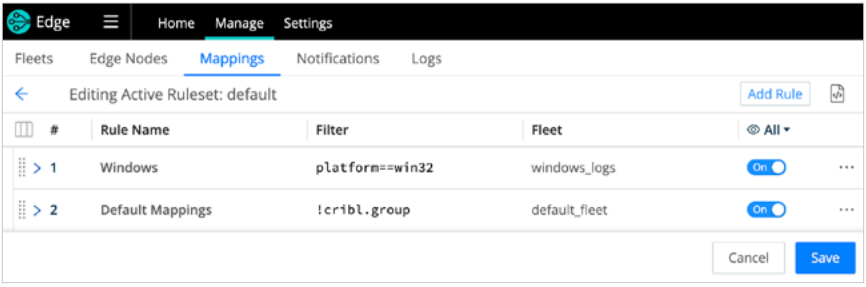
First, we add a new platform-specific rule in the existing `default` Ruleset:



Mapping to the SubFleet

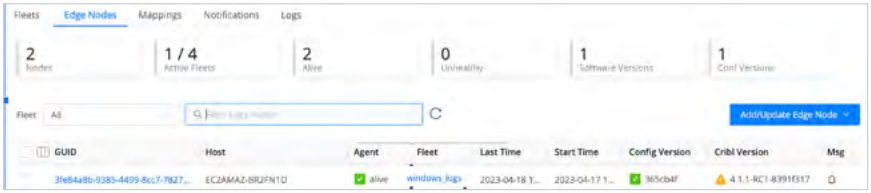
An important note: Only one Mapping Ruleset can be active at a time – although a ruleset can contain multiple rules. So it's important to keep your Fleet assignments within one ruleset. This example uses the `default` ruleset.

We want to reorder the rules, so that the new rule is first:



Reordering the rulesets

After we resave the Ruleset, Edge Nodes that match the `platform` value will now be assigned to the new SubFleet.



Viewing Edge Nodes

EXPLORE THE WINDOWS NODE

On your Leader, from the Fleet's **Explore** menu, you can select the **System State** tab to see snapshots of the host system's current state. The time intervals are configurable at the upper right.

Edge System State tab

You can specify which collectors are enabled, and the polling interval, by configuring the System State Source (see the reference at the end of this chapter).

CONFIGURE THE WINDOWS EVENT LOGS SOURCE ON CRIBL EDGE

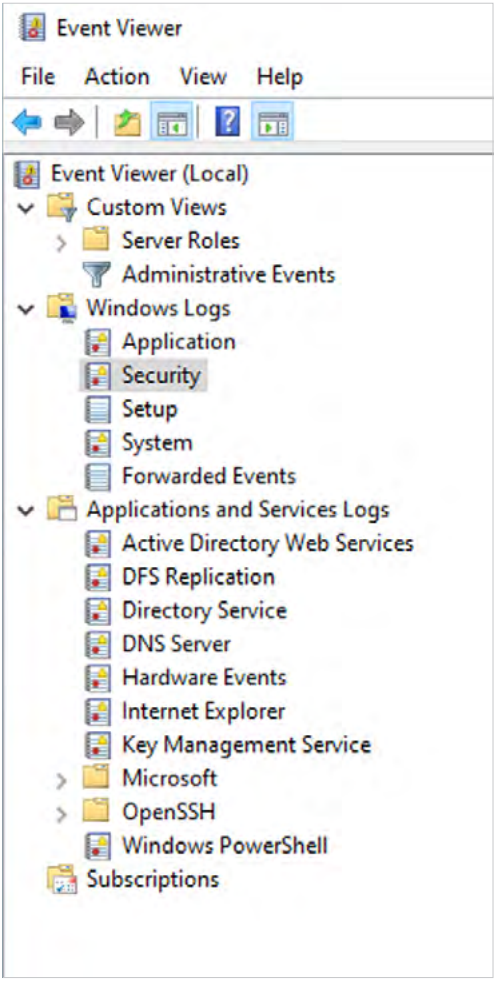
Next, we'll look at how to get Windows Event Logs into Edge, and on to your chosen destination. We must first identify the logs we want, and then configure Edge's input (Source) to match.

LOCATE WINDOWS EVENT LOGS ON THE SERVER

Event logs in Windows Servers are classified into the following broad categories:

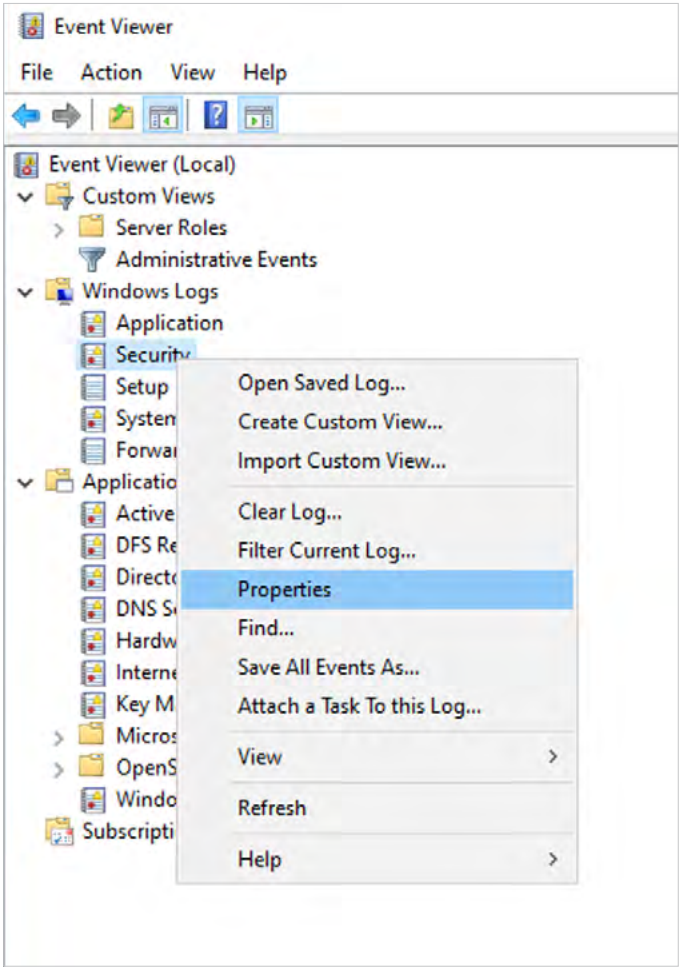
- **System**: Events related to the system and its components. Failure to load the boot-start driver is an example of a system-level event.
- **Applications and Services**: Events related to software, or to an application hosted on a Windows machine, get logged. For example, if a user encounters a problem in loading the app, it will be logged.
- **Security**: Events related to the safety of the system. The event gets recorded via the Windows auditing process. Examples include failed authentications and valid logins.
- **Setup**: Events that occur during the installation of the Windows operating system. On domain controllers, this log will also record events related to Active Directory.
- **Forwarded Events**: Contains event logs forwarded from other machines in the same network.

To find out what other event logs might be of interest, examine your Windows Server's Event Viewer.



Examining Windows Event Viewer

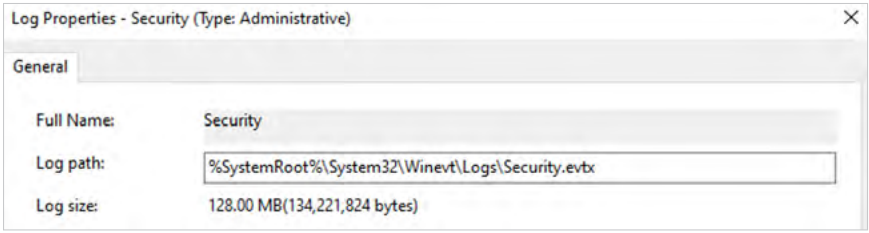
You can get each event log's accurate name by right-clicking and selecting **Properties**.



Viewing log properties

GET THE LOG'S FULL NAME

From the **Properties** modal, we'll copy the **Full Name**. When we configure Edge's **Windows Event Logs** Source below, we'll paste this into the **Event Logs** field.



Copying the full name from the Event Viewer UI

Alternatively, you can execute the following PowerShell command to list a particular log's name. In this example, we're using a `*` wildcard:

```
powershell Get-WinEvent -ListLog Hardware*
```

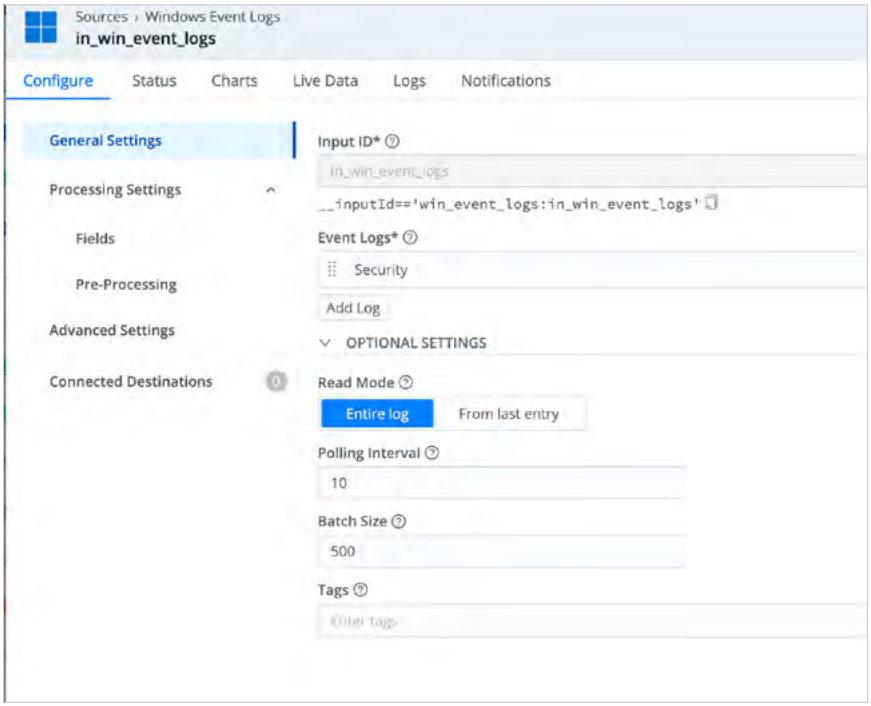
```
PS C:\Users\[redacted]> powershell Get-WinEvent -ListLog Active*

LogMode   MaximumSizeInBytes RecordCount LogName
-------   ------------------ ----------- -------
Circular            1052672          94 Active Directory Web Services
```

Log name from PowerShell

From the PowerShell output, the **LogName** field's value is what you'd copy to paste into the Edge UI. From the above example, you'd enter `Active Directory Web Services` in the Edge UI.

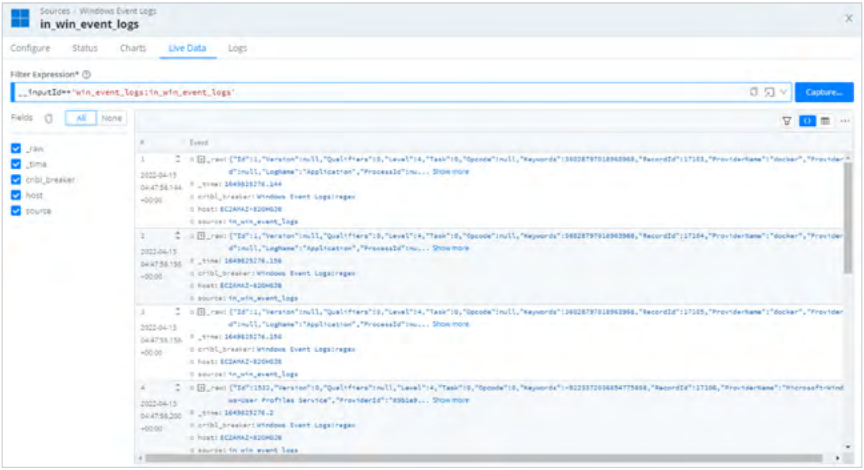
CONFIGURE THE WINDOWS EVENT LOGS SOURCE

Once we have the accurate names of the event logs we want to collect, we can configure and enable the `Windows Event Logs` Source in Cribl Edge. An example configuration modal is below.



Configuring Windows Event Logs

After we've configured the required fields (and any desired optional settings), we'll save, commit, and deploy the changes. Before moving on to the next step, we should check the Source's **Live Data** tab to make sure the logs are generated.



Verifying log generation via Source's Live Data tab

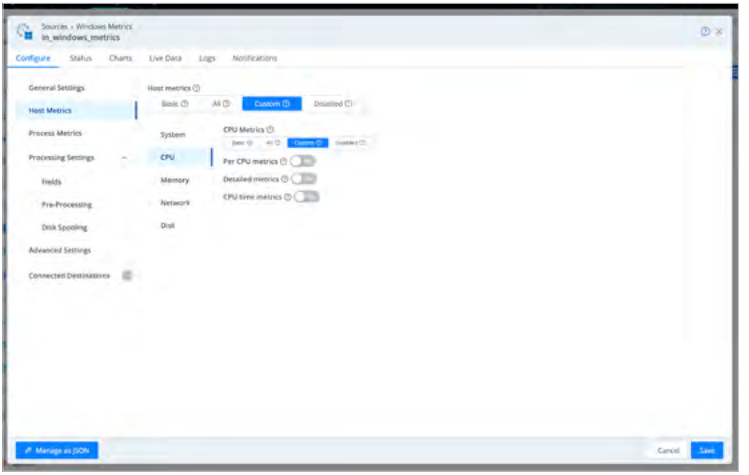
We can now send the event logs to a visualization tool of our choosing, or to any of Cribl Edge's other supported Destinations. We also have the option to route the logs through Cribl Stream for further processing on Stream Worker Groups with greater capacity.

CONFIGURE THE WINDOWS METRICS SOURCE

We'll also take a quick look configuring and enabling Edge's Windows Metrics Source.

The key requirement here is to decide the level of granularity for the metrics you want to collect. On this Source's **Host Metrics** left tab, we recommend selecting the **Custom** button, and then setting the following granularity options for three of the configurable types:

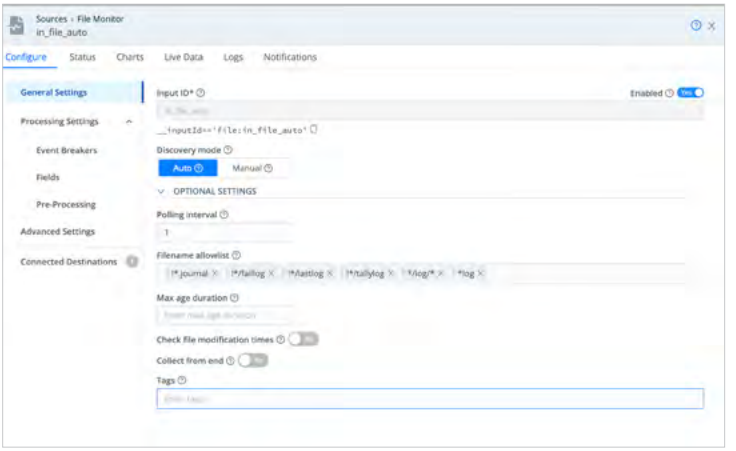
- **CPU**: Enable **Per CPU** metrics.
- **Network**: Enable **Per Interface** metrics.
- **Disk**: Enable **Per Volume** metrics.



Configuring a Windows Metrics Source

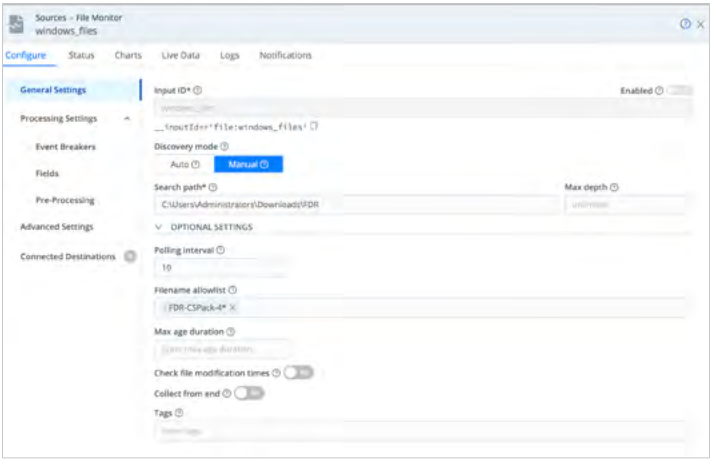
CONFIGURE THE FILE MONITOR SOURCE

To round out this walk-through, we'll take a quick look at enabling the File Monitor Source to gather application logs. The example configuration below monitors all files that end with variations on `.log`, as well as `.journal` files.



File Monitor Source in Auto mode

The Leader deploys with two File Monitor Sources preconfigured, respectively to operate in **Auto** and **Manual** modes. **Manual** enables you to specify a single path from which Edge will collect log files, along with a **Max depth** setting where you can limit collection to a specified depth of subdirectories.



File Monitor configuration in Manual mode

VARIATION: EXECUTING POWERSHELL COMMANDS

Here's the bonus topic. Edge's flexible Exec Source enables you to periodically execute a command and collect its `stdout` output. You'd typically configure and enable this in cases where Cribl Edge cannot accomplish collection with native Sources.

The Exec Source command is actually executed from a Windows command prompt, not from a PowerShell prompt. (PowerShell's execution is resource-intensive, so use it with caution. This applies especially to limited-resource systems, such as domain controllers, because the Source itself uses the host's CPU.)

The main thing to keep in mind when you configure Cribl's Exec Source is that your entry in the **Command** field will be slightly different from the corresponding PowerShell command.

PowerShell Prompt Command	Cribl Exec Source Configuration
`Get-Process`	`powershell Get-Process`
`# cd c:\mydir` `# .\myps-script.ps1`	`powershell` `c:\mydir\myps-script.ps1`

POWERSHELL EXAMPLE

To process the PowerShell command for a script like this:

```
Get-CimInstance win32_process | select ProcessId, Name, commandline,
HandleCount, WorkingSetSize, VirtualSize
```

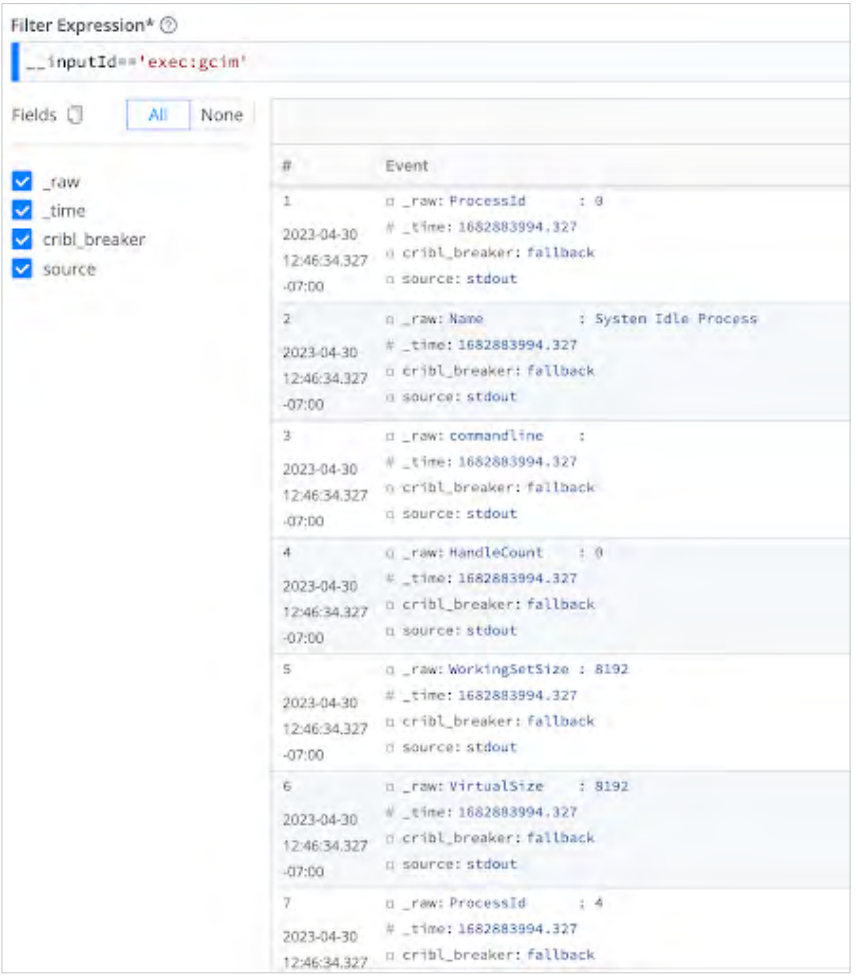
...you would enter the following into the Exec Source's **Command** field. (Add the `-command` parameter if your PowerShell version requires it:)

```
powershell [-command] "Get-CimInstance win32_process | select ProcessId,
Name, commandline, HandleCount, WorkingSetSize, VirtualSize"
```

Populating Exec Source's Command field

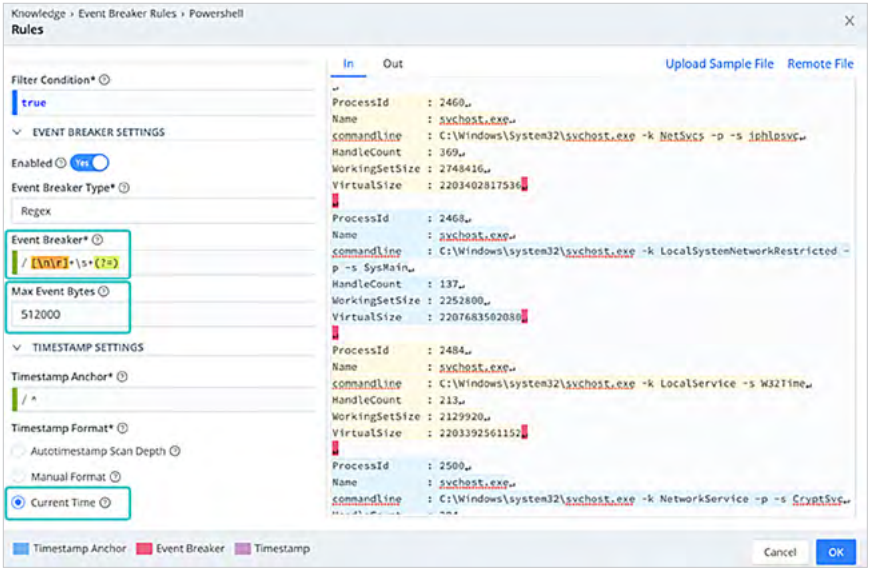
EXAMPLE OUTPUT

The output of the command, without using an Edge Event Breaker, will look something like this:



Output with unbroken events

With an Event Breaker, the output will look something like this:



Output with an Event Breaker

–

VARIATION: USING FLEETS TO MANAGE MANY DIVERSE HOSTS

Cribl Edge's hierarchical Fleets support heterogeneous deployments to monitor different types of hosts, in large numbers. You can use a top-level Fleet to set common configurations (such as Edge Destinations and processing Pipelines). Then you can set more-specific configs (like Edge Sources) in nested, descendant Subfleets.

For example, let's imagine (and analyze) a hierarchy of Fleet and Subfleets like this:

- **XYZ_Corp** – top-level Fleet for the entire company
    - **SiteA** – separate Subfleets for each data center
    - **SiteB**
        - **SiteB_Linux** – separate Subfleets for different platforms
        - **SiteB_Windows**
            - **SiteB_Windows_ISS** – separate Subfleets for different apps
            - **SiteB_Windows_SQL**

Here's how your organization might configure these levels:

The top-level **XYZ_Corp** Fleet could have Destinations common to all Edge instances, such as the company's central Cribl Stream cluster or notification systems. Routes and pre-processing Pipelines could be defined in common here for these Destinations.

The **Site** Subfleets could have additional local Destinations, or overrides of the parent Fleet's companywide instances. You could adjust the pre-processing Pipelines here to add site-specific properties useful to these local Destinations. You could also add local Routes to configure local logs and/or metrics storage.

On the platform-specific Subfleets, you'd enable Sources to collect platform-specific data. Examples would be File Monitor Sources for `/var/log` on Linux; Windows Event Logs Sources on Windows; and System Metrics or System State. To these Subfleets, you'd map Cribl Edge instances that run on generic hosts (those without application-specific monitoring needs).

The bottom-level, application-specific Subfleets are where you'd add Sources configured to collect logs and metrics from individual applications. To these Subfleets, you'd map Edge instances running on hosts that run each corresponding application.

This is just one illustration of how Edge's hierarchical Fleets provide a framework for managing large heterogeneous deployments. Fleets and Subfleets enable you to collect logs and metrics broadly, without having to independently manage each Node. This, in turn, facilitates managing as many as thousands of hosts.

WHAT'S NEXT?

You can explore Cribl Edge hands-on in our free **Introducing Cribl Edge** Sandbox (sandbox.cribl.io/course/edge). You'll find a hosted, fully functional instance of Cribl Edge, with source data set up for you to ingest and experiment with. The Sandbox walks you through deploying Edge Nodes, managing Fleets, configuring and managing Sources and Destinations, and exploring the Edge UI to discover insights into all the monitored data.

For further details on the Cribl Edge techniques outlined above, check out our online docs:

- Exploring Cribl Edge on Windows (docs.cribl.io/edge/explore-edge-win).
- Monitoring your Infrastructure with Cribl Edge (docs.cribl.io/edge/usecase-infrastructure-monitoring).
- Windows Observability Using Cribl Edge – online counterpart to this chapter (docs.cribl.io/edge/usecase-windows-observability).
- Installing Cribl Edge on Windows (docs.cribl.io/edge/deploy-windows).
- Fleets (docs.cribl.io/edge/fleets).
- Mapping Edge Nodes to Fleets (docs.cribl.io/edge/fleets#mapping).
- Fleets Hierarchy and Design (docs.cribl.io/edge/fleets-design).
- Setting Up Leader and Edge Nodes (docs.cribl.io/edge/setting-up-leader-and-edge-nodes).

- `instance.yml` file reference (docs.cribl.io/edge/instanceyml).
- Windows Event Logs Source (docs.cribl.io/edge/sources-windows-event-logs).
- Windows Metrics Source (docs.cribl.io/edge/sources-windows-metrics).
- File Monitor Source (docs.cribl.io/edge/sources-file-monitor).
- System State Source (docs.cribl.io/edge/sources-system-state).
- Exec Source (docs.cribl.io/edge/sources-exec).
- Cribl Edge to Cribl Stream (docs.cribl.io/edge/usecase-edge-stream).
- Event Breakers (docs.cribl.io/edge/event-breakers).
- Microsoft's *Start Command Prompt as an Administrator* topic.