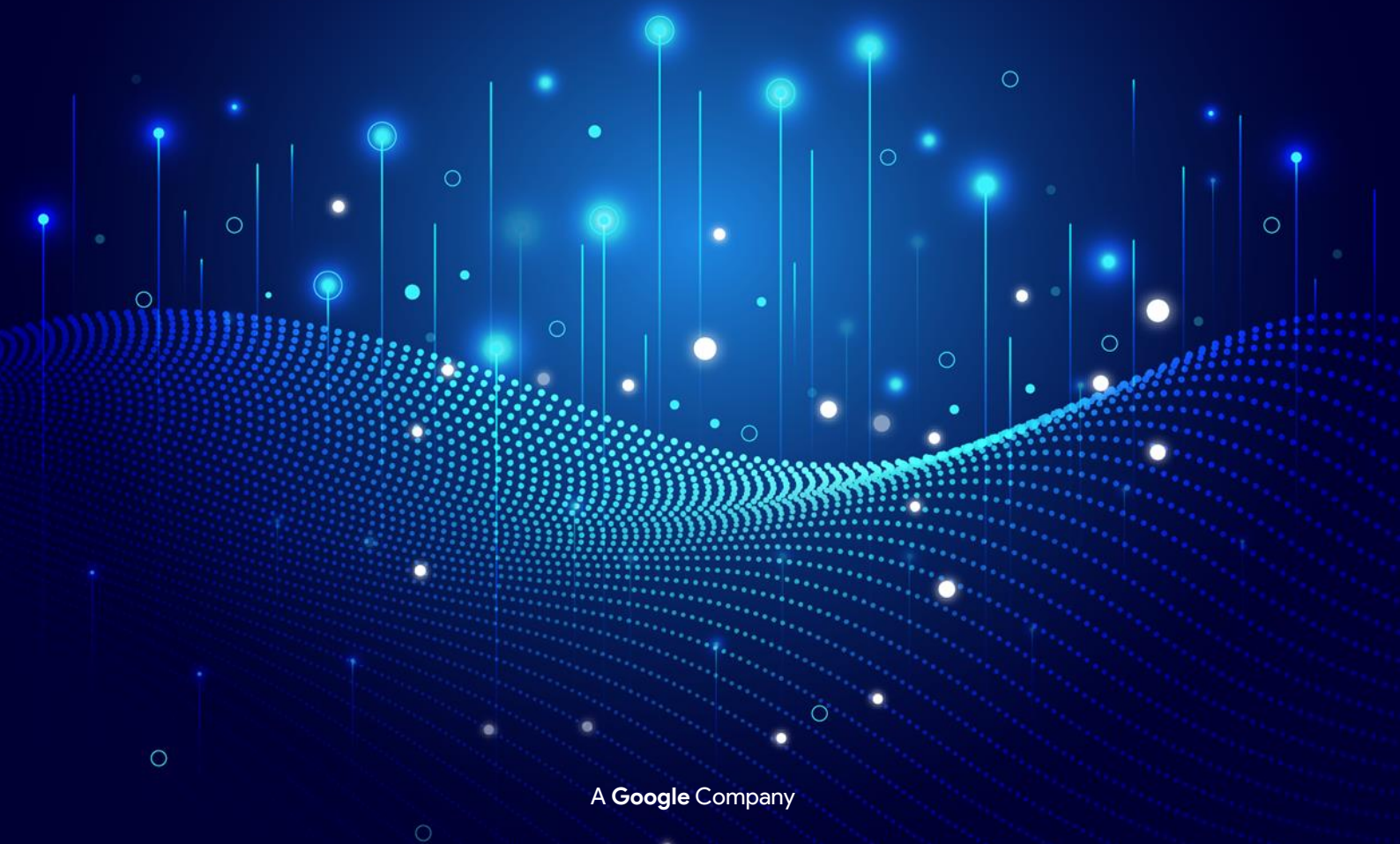
Siemplify
Solution Architecture

Chronicle & Siemplify Integration

A How to Guide



A **Google** Company

Table of Contents

Summary

The combination of Chronicle and Siemplify provides an unparalleled set of capabilities. In this article we will look at the steps to follow in order to integrate these two products. The main thing to remember when integrating with Chronicle is that Siemplify will require the service account details provided by your Chronicle provider. Once you have this information you are ready to head to the Siemplify Marketplace and unlock this powerful combination.

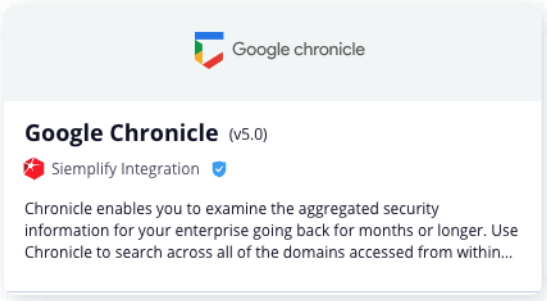
Implementation

Example Service Account

Your Chronicle Provider will provide the service account information. Just to be clear you will need to copy/paste the entire json file(from first to last bracket) into the configuration menus in Siemplify.

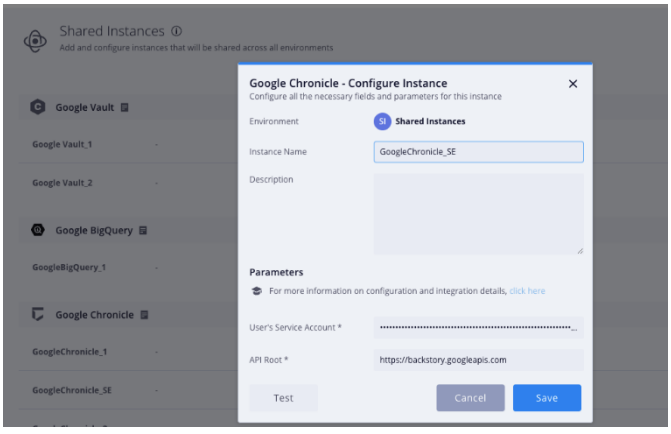
```
{
  "type": "service_account",
  "project_id": "project-id",
  "private_key_id": "key-id",
  "private_key": "-----BEGIN PRIVATE KEY-----\nprivate-key\n-----END PRIVATE KEY-----\n",
  "client_email": "service-account-email",
  "client_id": "client-id",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://accounts.google.com/o/oauth2/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/service-account-email
}
```

Installing the Integration

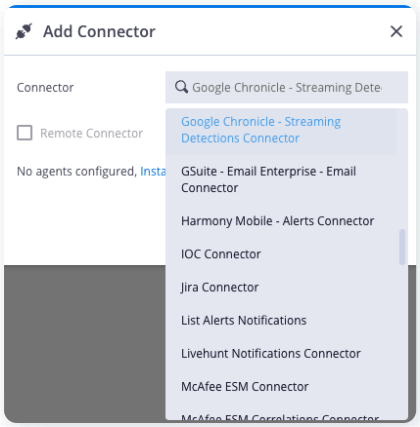


Goto the marketplace tab in your Siemplify Instance. Under Integrations you will search for Google Chronicle. You should find the following integration. There will be an arrow icon that you can click to install the integration on your instance.

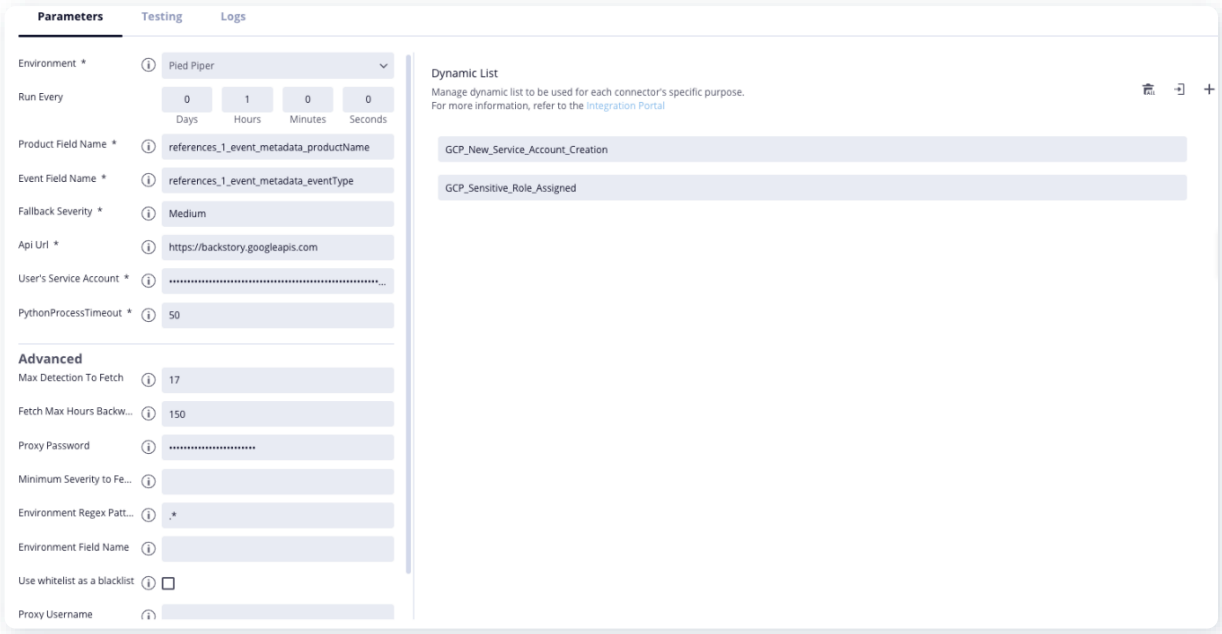
Now that you have downloaded the integration go to the settings->integrations tab. From here you can select the environment you want to install Chronicle in. In most cases this will be the shared instance. You will copy/paste the service account json contents into the “User’s Service Account” parameter, and the api root of chronicle in the “api root” parameter. Save your settings and click the test button and if everything is working a green check will appear. If something went wrong there will be a red marking and an error message.



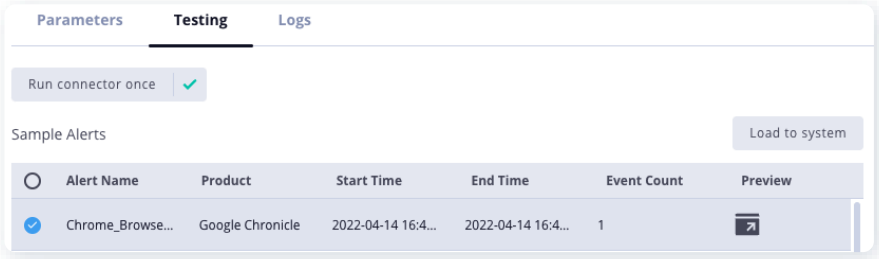
Installing the Connector



Once the integration has been installed in the shared instance you can now go to settings->connectors. Here you will click on the + button and open the menu shown to the left to add the Google Chronicle Connector. For the purposes of this demonstration we have chosen the most recent Chronicle Streaming Detections Connector.



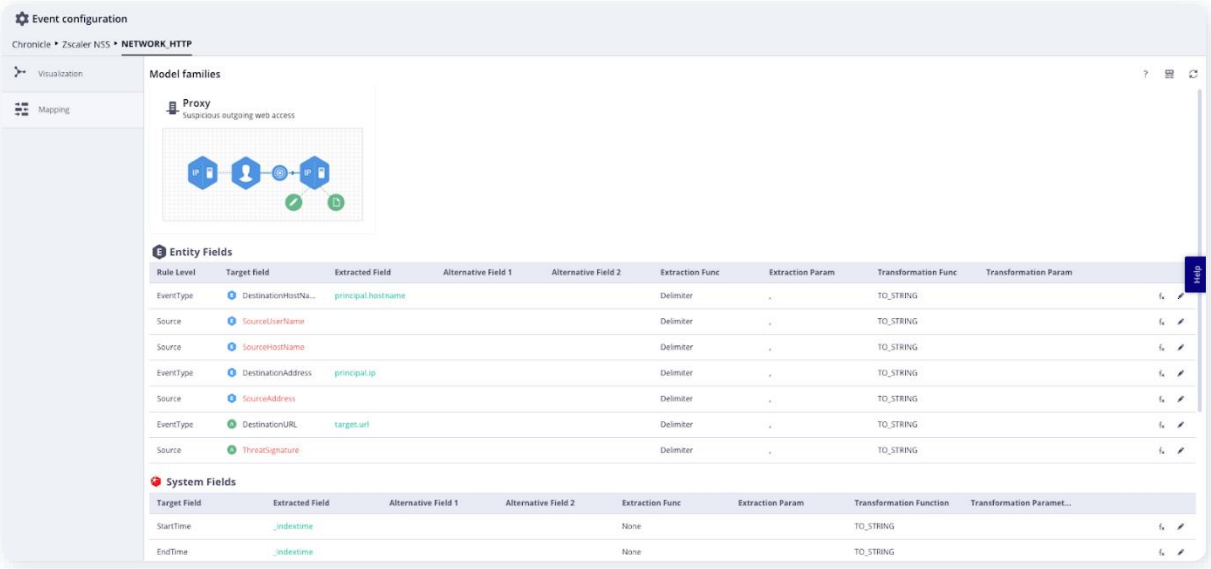
The figure above shows the configuration of the Google Chronicle Streaming Detections Connector. Important fields to note are the 'Product Field Name' and 'Event Field Name' which will determine how the ontology mapping of the alerts are handled. Similar to the integration configuration you must enter the 'api root' and the 'User's Service Account'. You then have options on how many detections to fetch at once and how many hours back the connector will look for alerts. To the right is the dynamic list that by default is used to determine which rules the connector will ingest. You have the option to use this list as a blocklist, meaning the rules listed would not be ingested, while all others will.



Once you have configured the connector and saved your changes, switch to the testing tab. Here you will push the 'run connector once' button to ingest a single test alert. Once this completes successfully you can select the case and click 'load to system' to make it appear in your cases tab. You can also use the 'preview' button to inspect the event fields related to the alert.

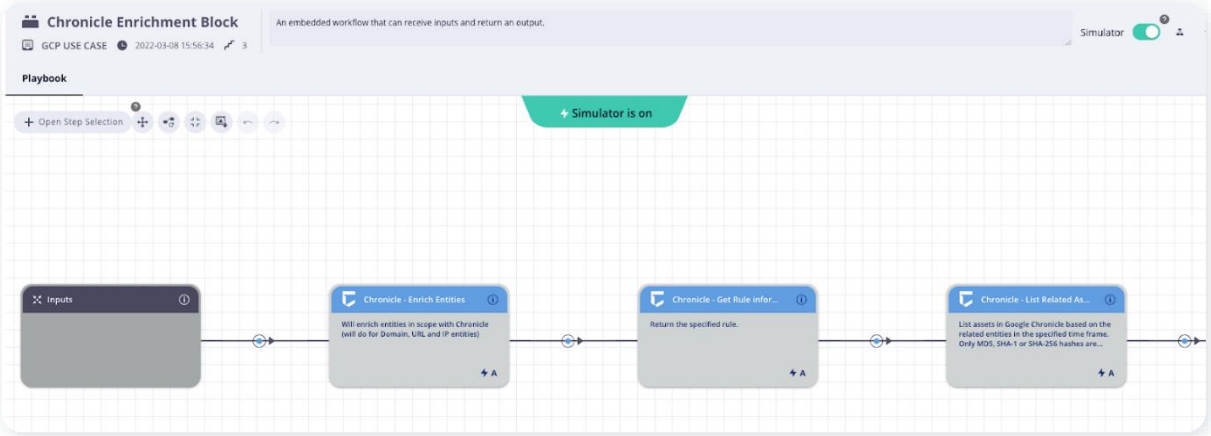
Ontology

Siemplify maps the event fields of alerts in a process known as ontology. The menu can either be accessed through the settings tab, or when clicking on the events tab of an alert there is a configuration button that will take you there as well. Below is a figure of the Zscaler alert that was ingested into Siemplify from Chronicle. Let's go over all its components.



The first important part of the ontology screen is the section that shows the Source-Product-Event levels of the alert. In this example it is Chronicle-Zscale NSS-Network Http. You can select any of these levels to make your mapping. They inherit the mapping left to right. So starting out on the product level is advisable, then making any custom changes per event type. The other important setting is the visual family that is chosen. In this example the Proxy family has been selected because it does the best job showing the entity relationships of events from Zscaler. Finally you can see the Siemplify Entities that are being mapped to the event fields. The fields in this example are green because they have been mapped and are present in the event. An important starting point when mapping are the StartTime and Endtime. These fields are used to build entity relationships. You also have a lot of flexibility with features like backup fields, extraction functions and even transformation abilities. For instance the time formats can be transformed which can be very helpful. Chronicle will have several data sources that will need to be mapped, but much of that data will be in UDM format which should make some of the job easier. However, it is always advisable to double check the raw event fields and make sure to account for any exceptions.

Playbook Actions



Once everything is running properly it is time to start using all the actions available when building playbooks. Beyond just ingesting alerts the integration lets you automate enrichments, search for related assets and run advanced queries. The block above is a good starting point. By putting these three actions in a block you can now utilize this block in any playbook that will be triggered by Chronicle.


Siemplify
Solution Architecture