CREST
DATA SYSTEMS

Zeek Automation

Technical Design Document

Version Control

Document Version	Date	Owner	Document Status	Comments
1.0.0	29th January 2021	Crest Data Systems	Initial Draft	Initial Draft
1.0.1	16th Feb 2021	Crest Data Systems	v1	v1
1.0.2	20th March 2021	Crest Data Systems	v2	v2

Table of Contents

Overview

Chronicle is a Security Information and Event Management (SIEM) tool used by Google enterprise customers to protect their assets, whether in the cloud or on-prem. By ingesting Google Cloud Platform (GCP) logs and transforming them into Universal Data Model events, Chronicle helps GCP customers with cloud threat detection and investigation.

Zeek (formerly known as Bro), a network security monitoring tool used to produce network security telemetry data. It is a passive, open-source network traffic analyzer. It is primarily a security monitor that inspects all traffic on a link in depth for signs of suspicious activity. More generally, however, Zeek supports a wide range of traffic analysis tasks even outside of the security domain, including performance measurements and helping with troubleshooting.

Fluentd is an open-source data collector for unified logging layers. It allows you to unify data collection and consumption for better use and understanding of data. Cloud Logging agent is an application based on Fluentd that runs on your virtual machine (VM) instances. In its default configuration, the Logging agent streams logs from common third-party applications and system software to Logging. You can configure the agent to stream additional logs too.

Business Case

Intended users

GCP Admin belonging to the Google Chronicle Enterprise Customers' Organization.

Problem to solve

This project aims to simplify the deployment of Zeek so that the GCP customers can feed raw packets from VPC Packet Mirroring and produce rich security telemetry for threat detection and investigation in our Chronicle Security Platform.

Goal

As a part of this project, a preconfigured Zeek + Fluentd golden image will be created which will be accessible from the GCP account. Terraform scripts will be provided to the customers via a repository to provision the GCP infrastructure.

Inclusions

Below are the intended modules desired as part of this project:

- Create a preconfigured Zeek + Fluentd golden image using Packer.
- Terraform scripts for provisioning infrastructure.
- User guide having information about how to run terraform scripts by GCP admin.
- Troubleshooting instructions.

Exclusions

- Creation of the Chronicle demo environment.
- Creation of the GCP service account.
- Zeek and Fluentd core functionality.

Assumptions

- Mirrored VPC already exists.
- Trust set up is not required since GCP Admin shall run the terraform scripts for provisioning.
- Users will be providing required parameters for infrastructure setup.
- Users will provide only one subnetwork for each region for collector vpc setup.
- If users will create mirrored VPC with custom subnets then they need to specify the appropriate regions in the collector VPC setup.

Modules

This section lists the detailed modules catered by the solution.

1. **Create a preconfigured Zeek + Fluentd golden image using Packer**

   Description - All the customers will have the common configuration of Zeek & Fluentd. To have stability in the machine image and for the super fast infrastructure deployment, Packer will be used to create the preconfigured Zeek + Fluentd golden image.

   The base image for this golden image will be the latest non-deprecated debian-10 image. The golden image will have the following things installed:

   - Git
   - Zeek
   - Fluentd

- Zeek & Fluentd dependencies
- Necessary Zeek and Fluentd configurations

Necessary Zeek and Fluentd configurations - The base Zeek package will be updated as per the requirements. The changes will be as follows:

- json-streaming-log.zeek file will be added to the Zeek package.
    - Source: gs://david-tu-public/json-streaming-logs.zeek
- Managed instance interface value will be set at runtime through terraform scripts.
- Zeek configuration file will have the following static values for specified variables:
    - Log expiration interval = 3 days
    - Disable default flag = true
    - Ignore checksum = true
- JSON streaming log files will have a new field named “vpc_id” which will be fetched at runtime from mirrored VPC instance metadata through terraform script and will be appended to the log files.
- JSON streaming log files will have a new field named “project_id” which will be fetched at runtime from collector VPC instance metadata through terraform script and will be appended to the log files.

Deployment - This golden image will be made public and will be available for all the Google Cloud projects.

Acceptance Criteria -

- Packer script to create Zeek + Fluentd golden image.
- The image should be available for all the Google Cloud projects.

Testing Criteria -

- When the instance is created from the golden image, it should have Zeek & Fluentd installed.

2. **Terraform scripts for provisioning infrastructure**

   Description - To provision the infrastructure on the GCP in an automated way, the GCP terraform provider will be used. Following are the GCP resources that will be provisioned using the GCP terraform provider:

   - GCP packet mirroring policies
   - Collector VPC
   - GCP forwarding rule
   - GCP Internal load balancer
   - GCP managed instance group (Backend for internal load balancer)
   - GCP instance template
   - GCP firewall rules:
     - Ingress rule for collector VPC
     - Egress rule for mirrored VPC
   - GCP health checks

   Configurable parameters :

   - Required parameters for collector VPC setup:
     - VPC
     - Region(s)
     - Subnet(s)
   - Optional parameters for packet mirroring policies:
     - Subnetworks
     - Tag
     - Individual instances
   - Optional parameters for traffic filtering:
     - Protocols
     - IP range
     - Traffic direction

   Deployment - A git repository consisting of the terraform scripts will be shared with the customers along with the series of steps to deploy the infrastructure on GCP.

   Acceptance Criteria

   - Terraform script should be able to manage multiple resources like GCP packet mirroring policies, internal load balancer, managed instance groups, etc.

- Terraform script should be able to accept the required parameters like VPC, region(s), and subnet(s).
- Terraform script should be able to accept the optional parameters like subnetworks, tags, individual instances for enabling packet mirroring policies into mirrored VPC instances.
- Terraform script should be able to accept optional parameters like protocols, IP range, and traffic direction for traffic filters into mirrored VPC instances.
- Terraform script should be able to get user access keys for account access.
- Terraform script should be able to access Zeek + Fluentd golden image for the setup of managed instance groups.
- Terraform script should be able to use GCP storage for state files

Testing Criteria -

- All the resources like GCP packet mirroring policies, internal load balancer, managed instance group, etc should be configured on GCP.
- All regional resources like internal load balancer, managed instance group and instance templates should be created in all the required regions.
- VPC peering should be created between mirrored and collector VPC, and it should be used for all regions.
- Packet mirroring policies should be applied to all instances within the specified subnetworks, tag, and individual instances.
- Traffic filters should be applied to mirrored vpc instances.
- Zeek + Fluentd golden image should be configured into all instances of the managed instance group.
- Packets should be forwarded to the internal load balancer from mirrored VPC network.
- State files should be updated on GCP storage.

Packaging and Deployment

The deployment for both the modules mentioned in the section above will be done in the following manner:

- Zeek + Fluentd golden image - The golden image will be made public and will be available for all the Google Cloud projects.

- Terraform scripts to provision infrastructure - A git repository consisting of the terraform scripts will be shared with the customers along with the series of steps to deploy the infrastructure on GCP.

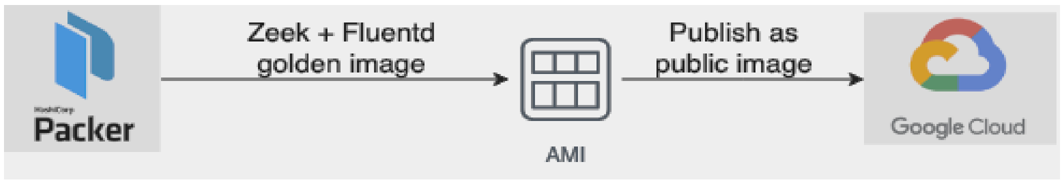
System Architecture

This section depicts the high-level system architecture for the solution and design considerations.

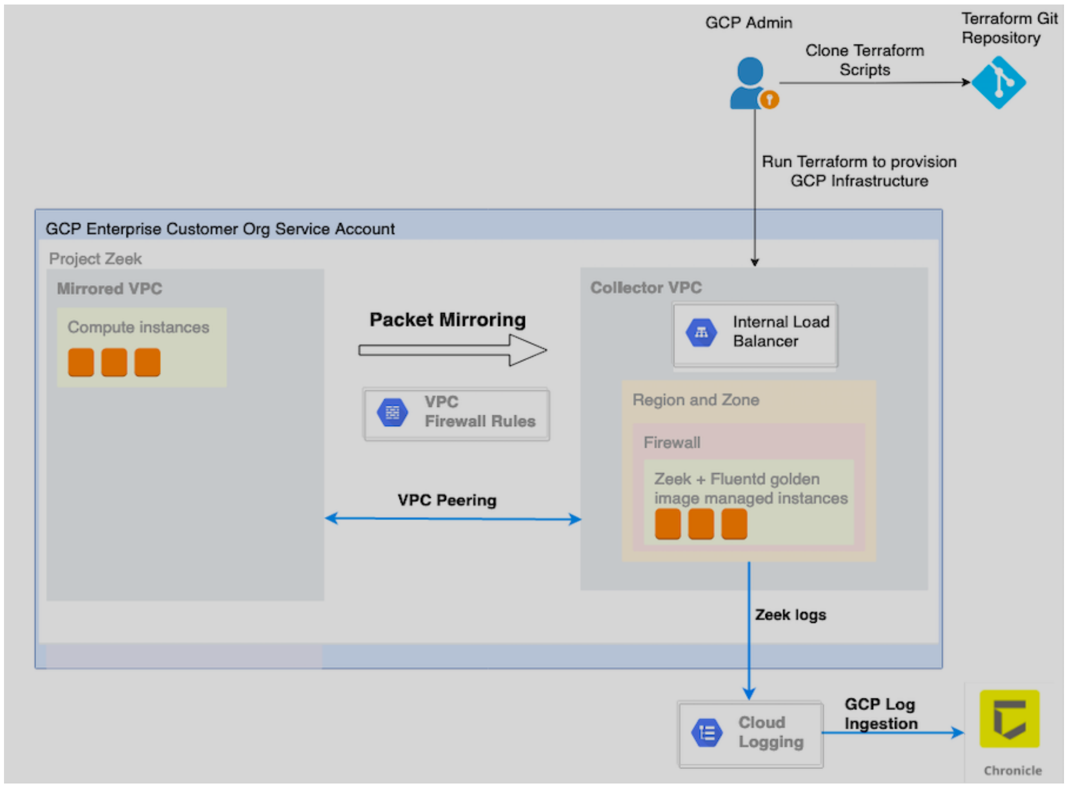
1. **Core Architecture**

   Zeek automation can be divided into two categories:

   - As all the customers will have the common configuration of Zeek & Fluentd. To have stability in the machine image and for the super fast infrastructure deployment, Packer will be used to create the preconfigured Zeek + Fluentd golden image. This golden image will be made public and will be available for all the Google Cloud projects. The below diagram shows the same.



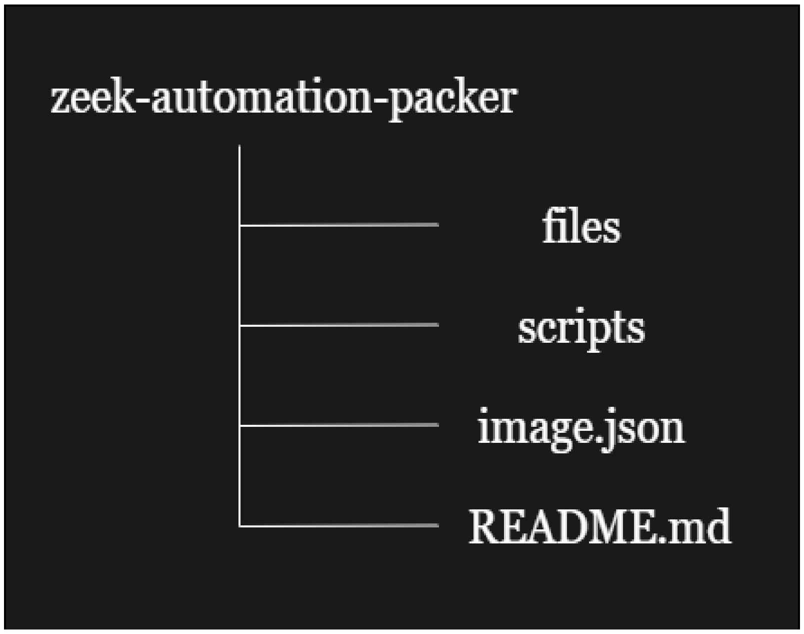
- A git repository consisting of the terraform scripts will be shared with the GCP admin belonging to the Google Chronicle Enterprise Customer's Organization. After the repository is cloned, the GCP admin can run the terraform scripts to provision the GCP infrastructure. As a sub-part of the provision, terraform script will use the above golden image to provision the managed instances in the Collector VPC. The below diagram shows the same.



Low-Level Design

This section describes the low-level design (directory structure) of the application.

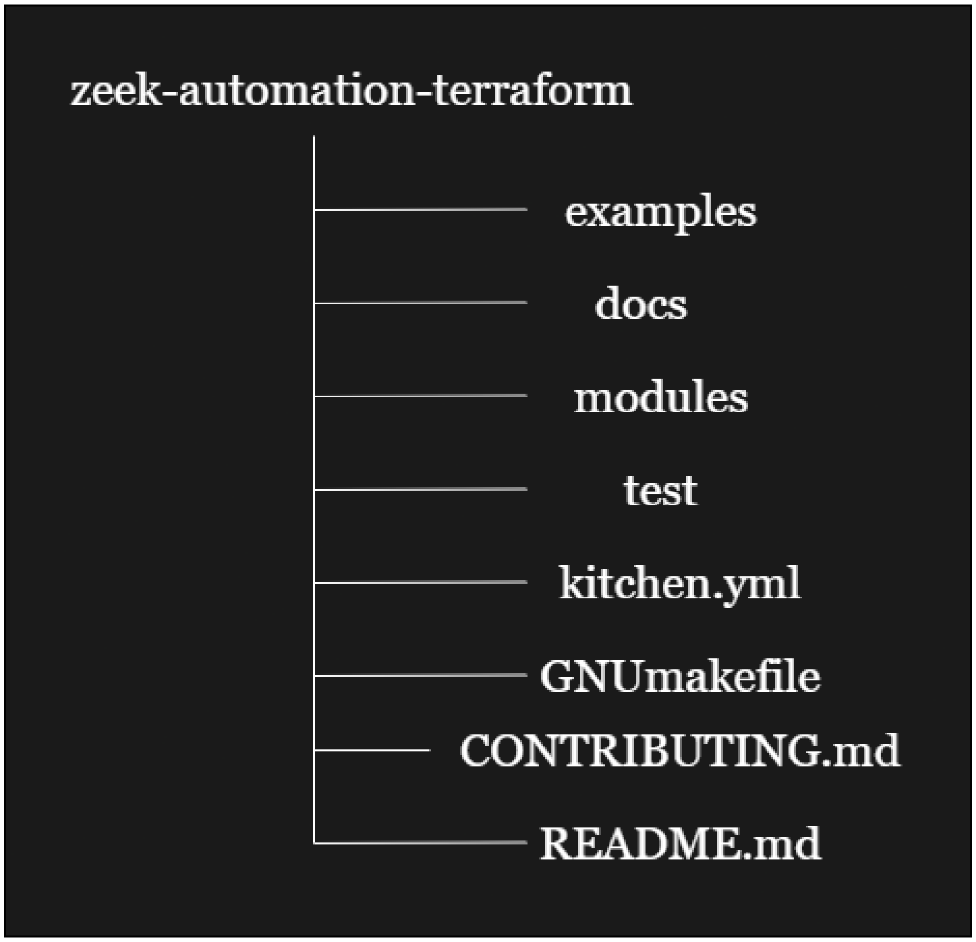
1. **Application Directory Structure**
    a. **Packer**



Directory Name	Description
zeek-automation-packer	This is a root directory which consists of all files, scripts and packer json file which is required for AMI creation
files	This directory includes all zeek and fluentd configuration related files.
scripts	This directory includes all zeek and fluentd configuration shell scripts.
image.json	This is the main packer image creation file.
README.md	It contains the structural and procedural flow of the zeek automation packer and technical requirements for the creation

	of packer AMI.

b. Terraform



Directory Name	Description
zeek-automation-terraform	This is a root directory which consists of all terraform files, modules and helpers.
examples	This directory includes the working example usage of the module.
modules	This directory includes a module which consists of all the resources required for infrastructure setup and each module has the README file.

test	This directory includes unit and integration testing files.
kitchen.yml	This file is basically the entry point of all tests.
GNUmakefile	This file represents the make commands which is used for document generation, integration testing, etc.
CONTRIBUTING.md	Markdown file showing how developers can contribute in this repo.
README.md	It contains the structural and procedural flow of the zeek automation terraform and technical requirements for the creation of whole infrastructure.

2. **Tools/Technologies Versions**
    - Packer: 1.6.6
    - Terraform: v14

References

- https://zeek.org
- https://www.packer.io/
- https://www.terraform.io/

Glossary

Keyword	**Description**
Fluentd	Open-source data collector for unified logging layer
Zeek	A network security monitoring tool used to produce network security telemetry data
GCP	A suite of cloud computing services
VPC	On-demand configurable pool of shared computing resources allocated within a public cloud environment, providing a certain level of isolation between the different organizations using the resources
Packet Mirroring	Packet Mirroring clones the traffic of specific instances in your Virtual Private Cloud (VPC) network and forwards it for examination. Packet Mirroring captures all traffic and packet data, including payloads and headers and is useful when you need to monitor and analyze your security status.
VPC peering	A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses.
Terraform	Infrastructure as a code software tool
Packer	Packer is an open-source tool for creating identical machine images for multiple platforms from a single source configuration.
Mirrored VPC	VPC that is being monitored
Collector VPC	VPC where the managed instances shall be provisioned by terraform to collect logs
Golden Image	A golden image is a pre-configured template for the users.