Global Information Assurance Certification Paper

Open-Source Endpoint Detection and Response with CIS Benchmarks, Osquery, Elastic Stack, and TheHive

GIAC (GCIH) Gold Certification

Author: Christopher Hurless, christopher.hurless@gmail.com
Advisor: *Tanya Baccam*

Accepted: *30 September 2020*

Abstract

There is a wealth of open-source tools available for information security. A characterization of the various open-source products will provide a means of fortifying endpoints and auditing those fortifications with an Endpoint Detection and Response (EDR) solution. High-quality security practices do not have to be expensive products, but they do need to hit several automation requirements to be effective. With this in mind, building robust, automated, EDR capability using open-source, community-driven tools that automate and standardize security responses is not only possible but practical.

Having a set of predefined control settings on an endpoint goes beyond malware detection. It sets the stage to ensure that an organization's endpoints are fortified from an attack before it happens. By implementing the Center for Internet Security (CIS) Desktop Benchmarks, organizations have a means of strengthening endpoints from attack. Adding Osquery allows them to have a tool for knowing when a machine has fallen out of a fortified state. Following the loss of fortification is the need to investigate the cause and return the device to its intended state which can be done using Elastic Stack and TheHive.

1. Introduction

There is a shortage of Information Security Analysts in the United States labor market (U.S. Bureau of Labor Statistics, 2020). Creating, standardizing, and automating security processes becomes a means of mitigating this shortcoming. Through the automation and standardization of incident responses, more professionals with a wider array of skill levels are able to handle a case simply because much of the guesswork associated with an incident can be removed, and the investigator can be sent down a specific, guided path that gives a known result. Since good processes reduce the skill requirements for incident handling, the impact of the available analysts is maximized. Additionally, organizations can make use of non-security-focused Information Technology staff who are accustomed to handling a wide array of technical problems and can follow sets of instructions that lead to the desired result of closure or escalation.

An important part of fortifying endpoints is knowing what an organization's endpoint security configuration should be and when the endpoint has deviated from this configuration. CIS Benchmarks and Osquery will be useful in this regard. CIS Benchmarks are defined as "consensus-based secure configuration guidelines which are applicable to a variety of operating systems, middleware and software applications, and network devices, designed to assess Member's network cybersecurity" (Center for Internet Security, 2020). Subsequently, the open-source Osquery tool is a means of checking and reporting on the status of the endpoint configuration to ensure it meets the guidelines set forth by the CIS Benchmarks. This is a compelling method because it includes both enforcement and constant auditing of the system state. This concept is also a deviation from the standard suite of endpoint protection tools that focus on known malware threats and behaviors and shifts the focus to ensuring an organization's endpoint is fortified to the degree the organization has declared necessary. This form of endpoint fortification does not exclude other malware-detecting products. It does provide the organization a means of control beyond what a standard endpoint security suite is capable of by providing the ability to respond to changes beyond the scope of most endpoint malware detection software. This form of fortification allows IT staff to audit individual endpoint settings that are important to the organization, for example, the five-minute screen lock settings required by HIPAA (Centers for Medicare and Medicaid Services, 2007). Many organizations will set some sort of enforcement policy for screen

lockout but then have no mechanism to audit and ensure that policy is continuously enforced across the organization or to alert when that setting has fallen out of the defined requirement. Rather, these organizations must trust that the declarative Group Policy will always ensure that the setting is used as intended.

The concept of continuously measuring the status of endpoints must go beyond simple antivirus and endpoint detection applications to something more robust. An organization must be able to identify a wide variety of changes defined by policies that are regularly and automatically audited to ensure that the settings are enforced. In a paper by Deloitte titled "Five Essential Steps to Improve Cybersecurity," two fundamental ideas are set forth to adopting and advancing a metered risk-based approach to cybersecurity. One of these steps, "Fortify your Organization," states that "Organizations should seek to move from a compliance-based approach to a risk-based approach. Resist the urge to follow the easy route of performing ad-hoc patching and leaning too heavily on 'Super Tuesday patching'." (Deloitte, 2016). In this same vein, organizations must move beyond just enforcing settings and include auditing to ensure the most robust and informed cybersecurity approach.

When looking at better endpoint configuration controls that are designed to fit organizations and notify when the configuration has deviated from the required setting. Luckily CIS offers extensive documentation on such controls. CIS benchmarks (https://www.cisecurity.org/cis-benchmarks/) are "best practices for the secure configuration of a target system" (Center for Internet Security, 2020). Identifying the configurations that are necessary for an organization, then creating audit and enforcement processes, is the cornerstone of ensuring endpoint fortifications are solid and can endure across the organization. In my previous paper, "Exploring Osquery, Fleet, and Elastic Stack as an Open-source Solution to Endpoint Detection and Response" (Hurless, 2019), I established the efficacy of Osquery for detecting and enumerating damage to an endpoint from unwanted software installs. This paper will show that open-source tools can also be used to establish a baseline endpoint configuration, evaluate if that configuration has been changed, and alert on those changes by sending information from Elastic Stack. Events will be forwarded through "Elastalert" to "TheHive" where automated processes can help security analysts quickly investigate and remediate the deviant configuration. Additionally, this solution provides more advanced means to "Prepare for the inevitable attack" (Deloitte, 2016) by helping codify the

incident management process and creating methods for testing different attack types and how the security department reacts.

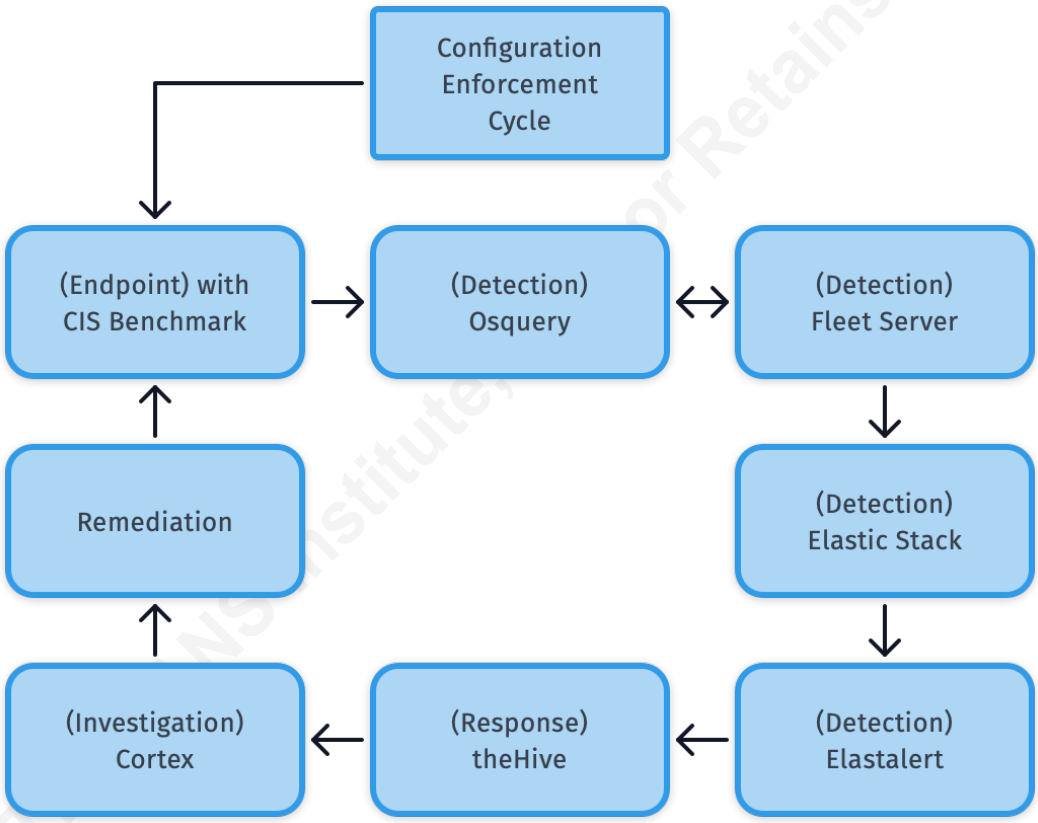

Figure 1. Detection and Response Cycle

2. Process and Technology Overview

From a high level this solution is designed to do three things. Primarily it will help to define a method for businesses to create configuration standards for endpoints using CIS Benchmarks. Following the commitment to CIS Benchmark settings a process follows where Osquery, Fleet, and Elastic stack will Audit, and Alert on changes to those settings. Finally, it will create a repeatable remediation process for endpoints that have deviated from the business configuration standard using TheHive.

2.1 (Endpoint) with CIS Benchmarks

CIS Benchmarks are sets of configuration recommendations and best practices for securing an endpoint. The recommendations are developed to help assist organizations in improving their security levels. CIS controls map to many established standards and regulatory frameworks, including the NIST Cybersecurity Framework (CSF) and NIST SP 800-53, the ISO 27000 series of standards, PCI DSS, HIPAA, and others (Microsoft, 2020).

2.2 (Detection) Osquery

Osquery is a security tool from Facebook, which provides a wealth of information about an endpoint using pure SQL commands. Finding information such as installed applications, running processes, loaded kernel modules, and open network connections are all practical uses of Osquery.

2.3 (Detection) Fleet Server

While listed as "detection" for its ability to create and schedule Osquery queries, Fleet can also be used extensively for investigations. Also, the queries run are recorded and logged for use by the Elastic Stack.

2.4 (Detection) Elastic Stack

All logged results from Osquery will end up in Elastic Stack where they can be searched and visualized. Elastic Stack consists of multiple independent applications that provide the foundation for the data visualization tool. Logstash, as the name suggests, is where logs are archived. Elasticsearch is the tool that indexes and arranges the logs while Kibana is the front end for interacting with the data visually. Also, a small log forwarding application, Beats, ensures that logs are transported from the host system to Logstash.

2.5 (Detection) Elastalert

Elastalert is an add-on for the Elastic Stack, which analyzes logs in real-time and forwards results to external entities via API, in this case, TheHive.

2.6 (Response) TheHive

TheHive will be the primary means for keeping accurate records, assigning tasks, and creating repeatable processes. It describes itself as: "a scalable 4-in-1, open-source, and free Security Incident Response Platform designed to make life easier for SOCs, CSIRTs, CERTs,

and any information security practitioner dealing with security incidents that need to be investigated and acted upon swiftly" (TheHive, 2020).

2.7 (Investigation) Cortex

Cortex is a powerful tool for investigation and analysis. Osquery results are capable of producing results as "observables" which can then be analyzed in TheHive. "When TheHive is used in conjunction with Cortex, security analysts and researchers can easily analyze hundreds of observables at once using more than 100 analyzers, contain an incident or eradicate malware thanks to Cortex responders" (TheHive, 2020).

2.8 Remediation

Remediation refers to the steps taken by analysts to resolve an incident and return an endpoint to its desired state. The remediation processes in this paper are designed to be repeatable using step by step instructions as defined within TheHive's incident handling structures.

3. The Lab Environment

Lab setup begins with a host computer using VMware Fusion Professional version 11.5.5. The hardware is a MacBook Pro (15-inch, 2017, 3.1Ghz Intel Core i7, 16GB 2133MHz LPDDR3) running MacOS Catalina version 10.15.6. This machine runs three virtualized operating systems or "guests" used for experimentation.

Hostname	IP	Role	OS	CPU	RAM
TheHive	192.168.13.181	TheHive, Cortex	Ubuntu 20.04	2	4096
Elastic	192.168.13.182	Elastic Stack, Elastalert, Kolide Fleet	Ubuntu 20.04	2	4096
MISP	192.168.13.179	MISP	Ubuntu 20.04	2	4096
Client02	192.168.13.178	Osquery	MacOS 10.15.6	2	4096

Table 1. Host Details

The host named "TheHive" runs both TheHive and Cortex. In a production environment, these would likely be hosted on two separate machines. The same is true of "Elastic" which is hosting Elastic Stack and Kolide. Not only would Kolide be hosted on a

different device, but it is also common to place Logstash, Elasticsearch, and Kibana on separate hosts as well.

4. Testing Methodology

"CIS Apple macOS Benchmark" (Center for Internet Security, 2020) configurations were set up on an endpoint running Apple OS 10.15.6. OSQuery was installed on the endpoint and connected to a Kolide/Fleet server. The logs of the OSQuery events were then forwarded to an Elastic Stack for real-time analysis of log events. When changes were noted in the OSQuery response, Elastalert sent details of the event to TheHive for automated analysis and canned response. To test the efficacy of this solution, unwanted software was installed on the endpoint, which modified settings that were being audited, kicking off the Endpoint Detection and Response cycle for remediation. For this test the MacOS Gatekeeper will be disabled on client02, and then a malware called "AppleJeus: Lazarus" will be installed. This research is a continuation of my previous paper, "Exploring Osquery, Fleet, and Elastic Stack as an Open-source solution to Endpoint Detection and Response", where I investigated Osquery's efficacy as an investigation tool (Hurless, 2019). With the addition of TheHive, the capabilities of investigation and remediation are intended to become standardized and available to any analyst regardless of immediate skill level.

4.1 Endpoint with CIS Benchmarks

As seen in Figure 1 the proposed "Detection and Response Cycle" begins with CIS Benchmarks, which are a part of the core of this endpoint fortification model. How the configuration is implemented into an organization's endpoints is beyond the scope of this paper. Still, it is assumed no endpoint will enter the fabric of the network until it has been configured to meet the organization's requirements. The goal here is to create endpoints that meet a set of security standards based on organization-specific criteria and ensure those configurations are maintained. If an endpoint has deviated from the standard configuration, this is an actionable item that could indicate a threat, a wrong user decision, or any unexpected change that needs to be understood. Looking at an example CIS Benchmark for MacOS Gatekeeper in Table 2 below the reasoning, or rationale for the configuration is followed by detailed steps for how this setting can be checked and changed. These step by step processes will later be converted into Osquery audits and then step by step processes for remediation in TheHive.

Rationale:

Disallowing unsigned software will reduce the risk of unauthorized or malicious applications from running on the system.

Audit:

Perform the following to ensure the system is configured as prescribed:

1. Run the following command in Terminal:
   $ sudo spctl --status

Ensure the above command outputs "assessments enabled." Remediation:

Perform the following to implement the prescribed state:

1. Open System Preferences
2. Select Security & Privacy
3. Select General
4. Select Allow applications downloaded from Mac App Store and identified developers

Alternatively, perform the following to ensure the system is configured as:

1. Run the following command in Terminal:
   $ sudo spctl --master-enable

Description:

Gatekeeper is Apple's white-listing control that restricts downloaded applications from launching. It functions as a control to limit applications from unverified sources from running without authorization.

Table 2. Example of a CIS Desktop Benchmark for Gatekeeper

4.2 Detecting Change with Osquery

If the recommendations of the CIS benchmarks are the “walls of the fortress,” Osquery makes up the “watchers on the wall.” Through the creation of a series of SQL-like queries, IT staff can monitor the status of desktops and log changes that are of interest to the organization. In this case, they are looking for modifications to CIS Desktop Benchmarks but

are in no way limited to only CIS Benchmarks. Osquery is a potent tool that should be expanded and explored as its implementation matures.

Following the example from Table 1, CIS Apple Benchmark 2.6.2 “Enable Gatekeeper” below is how a complimentary Osquery watcher search would appear in a terminal search.

```
osquery> SELECT assessments_enabled FROM Gatekeeper;
+--------------------+
| assessments_enabled |
+--------------------+
| 1                  |
+--------------------+
```

Table 3. Example of Osquery search

This specific SQL command reveals that Gatekeeper is enabled and that the CIS benchmark in question is in the expected state. The value status is recorded, and any change will be logged as a differential but also become the new baseline value from which change will be recorded. The full schema of osquery tables can be found here: https://osquery.io/schema/.

The schema offered by Osquery effectively watches settings that are in easy-to-access locations such as a PLIST file. It does start to break down; however, when specific tools such as pwpolicy are needed to check what a setting is. The same is true for Windows, where proprietary operating system tools are required to find the status of the setting.

4.2.1 Fleet Server

Moving beyond the Figure 1 diagram is the Fleet server: “Kolide Fleet is a beautiful, minimal, open-source web application for managing a fleet of hosts running osquery. Fleet gives you a place to store and iterate on Osquery queries” (Mike Arpaia, 2017). Below is how Osquery converts a scheduled query on Fleet.

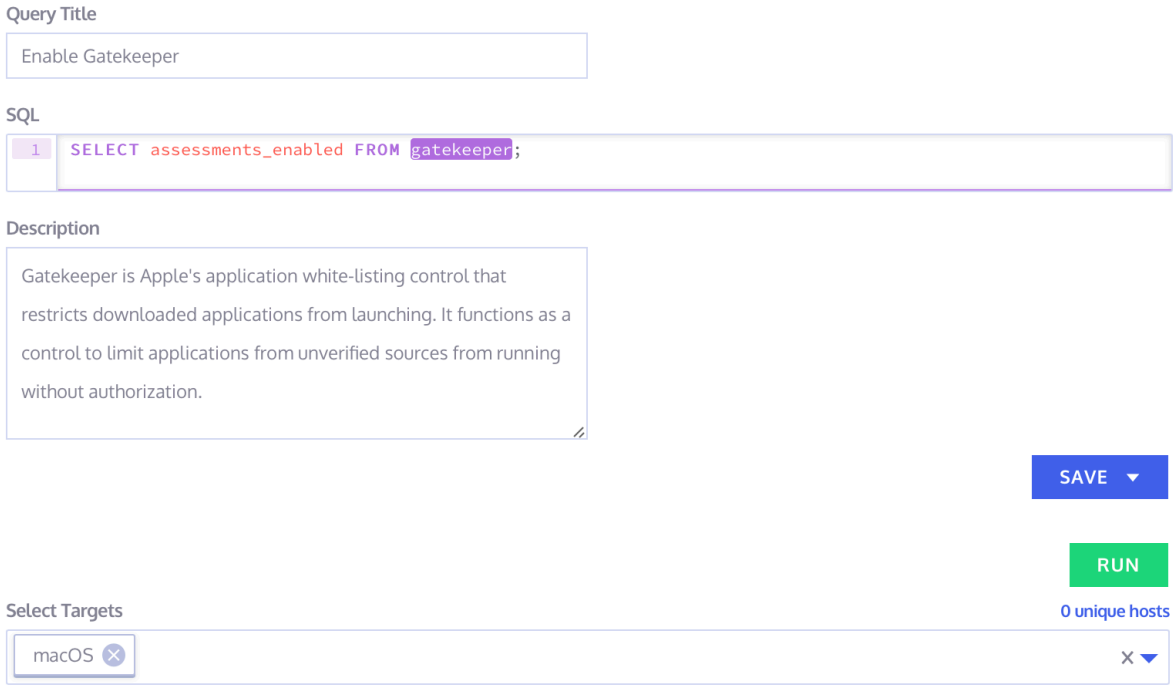

Figure 2. Osquery query as a fleet query pack

Figure 2 is the original Gatekeeper query, which now should be added to a query pack and set to run on a schedule.

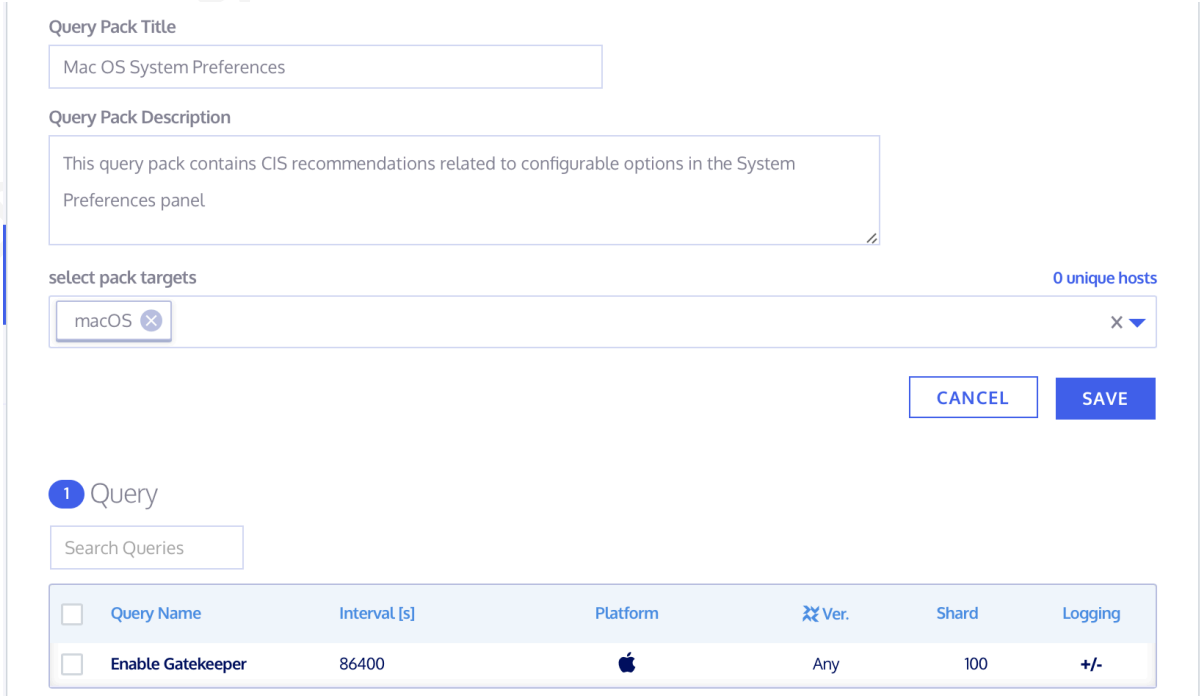

Figure 2.1 Osquery query as part of a query pack

Figure 2.1 shows the query pack, the selected target OS, the queries in the pack (in this case only the "Enable Gatekeeper" query), the interval at which the query is run (86,400

seconds equals one day), and the platform of the query (Apple) version of osquery that will run this query. Shard is the percentage of hosts that should run this query. The server load is reduced by lowering the shard from 100%. And finally, the logging type seen in figure 2.1 and represented as a +/- indicates a differential logging type, so only changes will be logged to the result file.

The Fleet GUI is powerful for visualizing and running Osquery queries however to populate fleet en masse, YAML files can be imported using the 'fleetctl' binary. The watcher for Gatekeeper would look similar to the following example in Table 4.

```
{
"scheduled_query": {
      "users_differential": {
            "query": "SELECT assessments_enabled FROM Gatekeeper;",
            "interval": 86400,
            "snapshot": true,
            "description": "Gatekeeper is Apple's application white-
            listing control that restricts downloaded applications from
            launching. It functions as a control to limit applications
            from unverified sources from running without authorization.",
            }
      }
}
```

Table 4. Example Fleet importable YAML

The first run of this query pack would create the following log entry to the results log. The critical point here is that "assessments_enabled" has a value of 1.

```
{
    "action": "added",
    "calendarTime": "Fri Jul 17 05:53:47 2020 UTC",
    "columns": {
        "assessments_enabled": "1"
    },
    "counter": 0,
    "decorations": {
        "host_uuid": "723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7",
        "hostname": "client02"
    },
```

```
    "epoch": 0,
    "hostIdentifier": "723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7",
    "name": "pack/Mac OS System Preferences/Enable Gatekeeper",
    "unixTime": 1594965227
}
```


Table 5. Log entry for Gatekeeper enabled query

Additional runs would create logs that show the query was run but would not return results as this query is looking for only a differential. This is simply the establishment of the baseline for this CIS benchmark. The marker from which future change will be noticed. The next step would be to turn off Gatekeeper and reference Table 5 for the result log.


```
{
    "action": "added",
    "calendarTime": "Fri Jul 17 06:00:25 2020 UTC",
    "columns": {
        "assessments_enabled": "0"
    },
    "counter": 7,
    "decorations": {
        "host_uuid": "723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7",
        "hostname": "client02"
    },
    "epoch": 0,
    "hostIdentifier": "723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7",
    "name": "pack/Mac OS System Preferences/Enable Gatekeeper",
    "unixTime": 1594965625
}
```


Table 6. Log entry for Gatekeeper disabled differential query

The “assessments_enabled” field has changed to 0, creating an entry in results.log. These logs are then ingested into Elastic Stack for the next stage of processing.

4.2.2 Considerations for Using Osquery with CIS Benchmarks

Osquery capabilities do not always map to CIS Benchmarks for several reasons. An excellent example on the MacOS side when checking password policies. CIS Benchmark 5.2.4 “Complex passwords must contain a Numeric Character” Osquery doesn’t currently

have the ability to check password complexity requirements on the Mac version. If the setting were in a non-binary plist file, the plist reader in Osquery could be used. In order to get the password policy value, open up a terminal, and run:

```
$ pwpolicy –getaccountpolicies | grep “Numeric”
```

There are several similar cases where Osquery was unable to be used to find a corresponding benchmark setting. One of particular interest is CIS Benchmark 6.3 “Disable the automatic run of safe files in Safari.” The method to check this setting is by running:

```
$ defaults read com.apple.Safari AutoSafeDownloads
```

However, by default in OSX 10.15, the terminal does not have direct access to the location of this plist. Terminal.app must have full-disk permissions via the “Security and Privacy” settings in System Preferences.

To correct these limitations, I have written a shim in Python that runs the native commands, such as ‘pwpolicy,’ and turns the output into plist files that can then be read by Osquery directly. With this shim in place, Osquery is now able to accurately report on almost every CIS benchmark for OSX. A few “unscored” benchmarks require GUI only to check and enforce, so these have been noted and left out. Additionally, I have created the configuration files needed to import into Fleet that would allow all of these settings to be monitored.

The shim and the Osquery/Fleet Configuration files are available on the following GitHub page at: https://github.com/christopherj525/osquery_cis_shim.

4.3 Alerting Change with Elastic Stack

The results from Osquery are moved via a log forwarder called “filebeat” into “Logstash” which is the Elastic Stacks log storage solution. Elasticsearch then traverses and indexes the logs. The logs are visually presented in Kibana, the web front end for the Elastic Stack solution as seen below in figure 3.

t	osquery.result.action	added
t	osquery.result.calendar_time	Fri Jul 17 06:03:16 2020 UTC
?	osquery.result.columns.assessments_enabled	1
?	osquery.result.counter	10
t	osquery.result.decorations.host_uuid	723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7
t	osquery.result.decorations.hostname	client02
?	osquery.result.epoch	0
t	osquery.result.host_identifier	723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7
t	osquery.result.name	pack/Mac OS System Preferences/Enable Gatekeeper
#	osquery.result.unix_time	1,594,965,796
t	prospector.type	log
t	read_timestamp	2020-07-20T05:08:15.034Z
t	source	/var/log/osquery/osqueryd.result.log

Figure 3 - Osquery result log visualized in Kibana

Here in Figure 3 is "osquery.result.columns.assessments_enabled" where the Gatekeeper status is set to 1, and the name of the pack and query that generated this result is under "osquery.result.name."

Next is the log result, which shows that Gatekeeper has been disabled. The event in Figure 5 is what will trigger an alert and move forward through the "Detection and Response" cycle. Here the result of "osquery.result.columns.assessments_enabled" has changed from 1 to 0, indicating it has been disabled, kicking off an Elastalert.

t	osquery.result.action	removed
t	osquery.result.calendar_time	Fri Jul 17 06:03:16 2020 UTC
?	osquery.result.columns.assessments_enabled	0
?	osquery.result.counter	10
t	osquery.result.decorations.host_uuid	723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7
t	osquery.result.decorations.hostname	client02
?	osquery.result.epoch	0
t	osquery.result.host_identifier	723A1A63-FED7-5BF0-A9CF-A6A3106DE8B7
t	osquery.result.name	pack/Mac OS System Preferences/Enable Gatekeeper
#	osquery.result.unix_time	1,594,965,796
t	prospector.type	log
t	read_timestamp	2020-07-20T05:08:15.034Z
t	source	/var/log/osquery/osqueryd.result.log

Figure 4. Osquery change result log visualized in kibana

4.3.1 Elastalert

"Elastalert is a simple framework for alerting on anomalies, spikes, or other patterns of interest from data in Elasticsearch" (Yelp, 2020) In short, it will forward log events, in this case to TheHive, in almost real-time based on rules and conditions met by log entries.

To follow the "Gatekeeper disabled" event onward to TheHive, an Elastalert will be created. The details of the alert are below in Table 7.

```
es_host: 127.0.0.1
es_port: 9200
name: Gatekeeper_disabled
type: frequency
index: filebeat-*
num_events: 1
timeframe:
    minutes: 1
filter:
- term:
    osquery.result.columns.assessments_enabled: 0
alert: hivealerter
hive_connection:
  hive_host: http://192.168.13.181
  hive_port: 9000
  hive_apikey: "/IGu/bG0HTzEDqua5U1v1OVhttjB3DLP"
hive_alert_config:
  type: 'external'
  source: 'elastalert'
  description: '{rule[name]}" on host
"{match[osquery][result][decorations][hostname]}" '
  severity: 3
  tags: ['{rule[name]}']
  tlp: 3
  status: 'New'
  follow: True
hive_observable_data_mapping:
  - hostname: "{match[osquery][result][decorations][hostname]}"
```

Table 7. Elastalert Gatekeeper example

Many essential values are shown in the Elastalert YAML text of figure 7 which maps values that will be passed on to TheHive and subsequently will become available in TheHive's alert preview.

Name: Gatekeeper_disable is a self-assigned name to describe the alert, which also acts as a filter tag giving the ability to see how many alerts are happening at a given time.

Index: filebeat-* Elastic can store a variety of logs and log sources; filebeat is the source for this. The name of the index can be changed to help define the host it came from, which would likely be more descriptive for a production environment.

Source: 'elastalert' If TheHive is receiving alerts from multiple sources, which it is fully capable of, this quickly identifies the alert as being triggered by the Elastalert system.

Description: '{rule[name]}' on host "{match[osquery][result][decorations][hostname]}" ' is a place to pass a description of the event on to TheHive alert. In this case, the description will read "Gatekeeper_disabled on host client02". This is a very malleable field that can contain a fair amount of text to help understand why this alert went off and what the next steps should be.

Severity: 3 Critical, High, Medium, Low: 4, 3, 2, 1 are the respective values for the severity levels.

The final construct that is important here is what data can be pulled directly from the logs and, also, what additional data an organization might want to append to an action. These "observables" are named in TheHive and are simply passed from Elastalert. In the case of the "hostname and "query-pack," these are references to the logs JSON entries. In the case of the "remediate" observable, a string from the CIS guideline for the remediation steps on this event was added.

Hive_observable_data_mapping: - hostname: "{match[osquery][result][decorations][hostname]}" This observable was included by default because any other event with the same attribute will be flagged in the incident. It can be used to map attributes for other automated checks in TheHive against its investigation tool cortex. As an example, if a query returns a field such as a file name, hash value, URL, or email address, those attributes can be forwarded as "observables" to a local MISP server, virus total, or any number of add-ons which are included with Cortex.

4.5 Responding to Change with TheHive

TheHive is described as "a scalable, open-source and free Security Incident Response Platform" (TheHive project, 2020), and represents the response tool in Figure 1 of the proposed "Detection and Response cycle." From there, TheHive alerts are noticed, investigations begin, and action is taken based on Osquery data and the remediation guidance provided by TheHive's workflow processes. Continuing with the Gatekeeper example, consider figure 5. This is an example of what the alert as it looks in TheHive:

Figure 5. Elastalert represented in TheHive

Next, Figure 6 below shows that a new external alert has been triggered, the log source is filebeat, and the rule name is Gatekeeper disabled. The severity level is 3 – High, and there is one observable attribute and then the date and time of the alert.

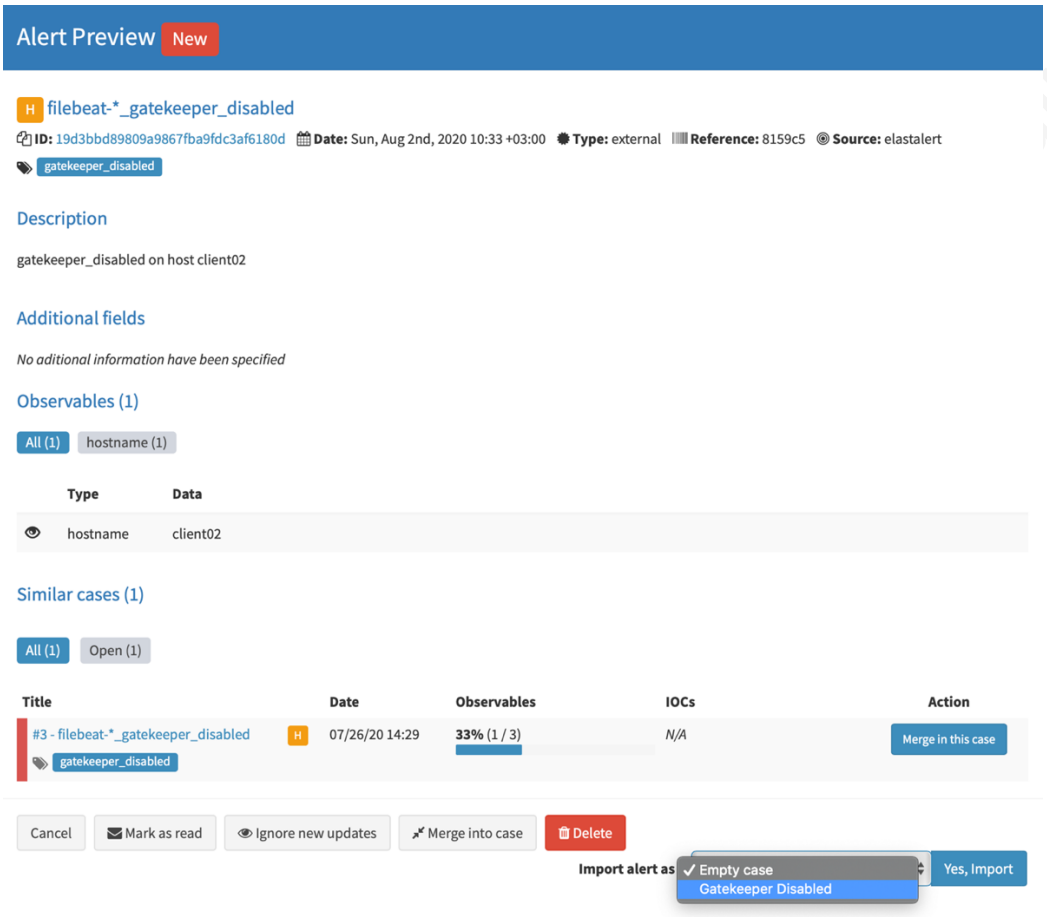

Figure 6. Elastalert represented in TheHive details

Figure 7 below shows how the different attributes from the original JSON log and the Elastalert from Table 1 map to the final alert seen here.

- index, rule name, type, source, and tags
- description – "gatekeeper_disable on host client02
- the observable data – hostname: client02

The alert can also be imported as a named case with a predefined process for investigation. Once imported, related cases that will correlate with each other based on observable and tagged criteria are visible. Figure 7 below shows how related cases are mentioned based on observables. Cases can also be merged.

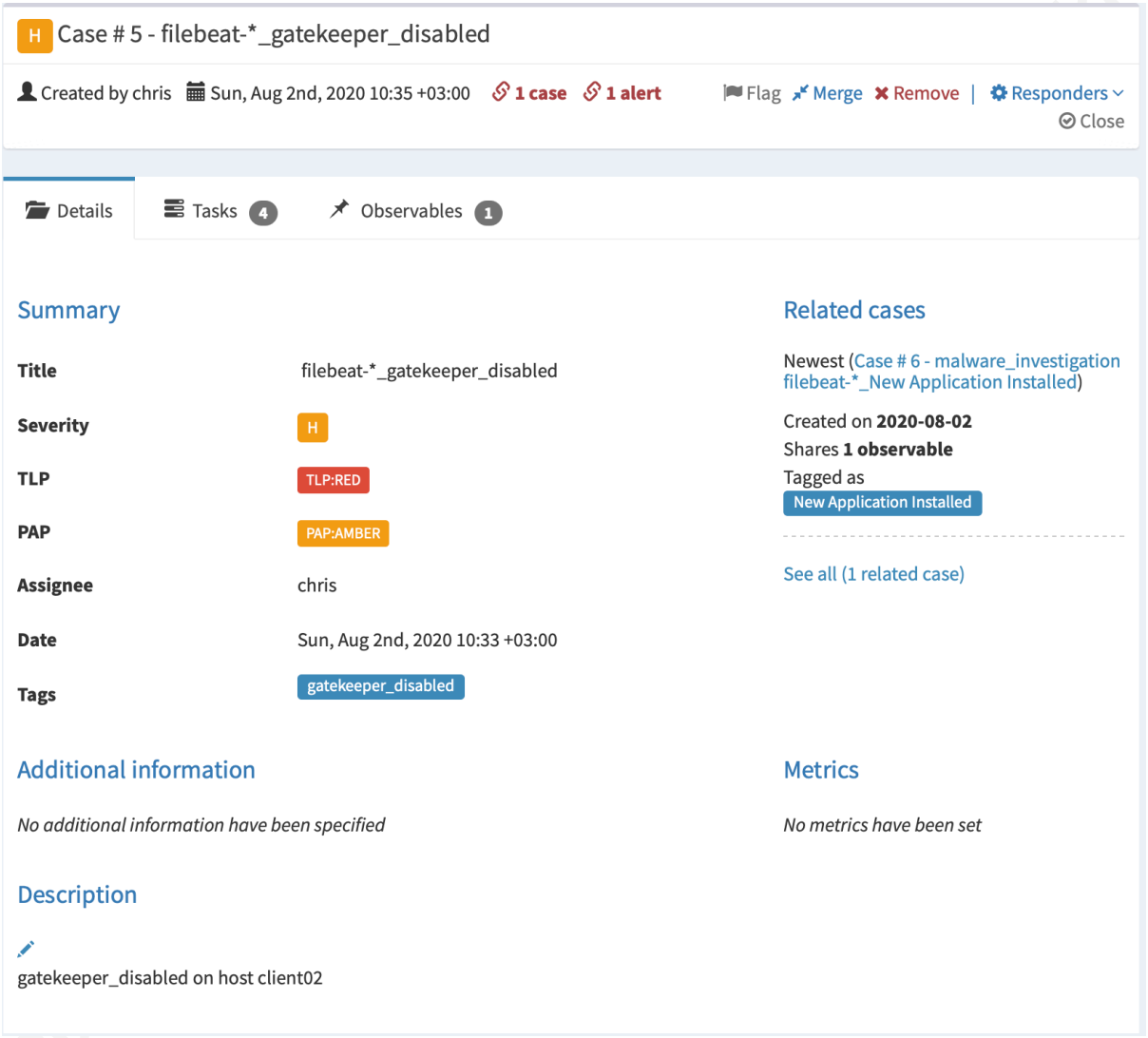

Figure 7. Elastalert Attributes in TheHive.

Following the opening of the case and assignment to a template, Figure 8 shows how the task list is established and can be assigned.

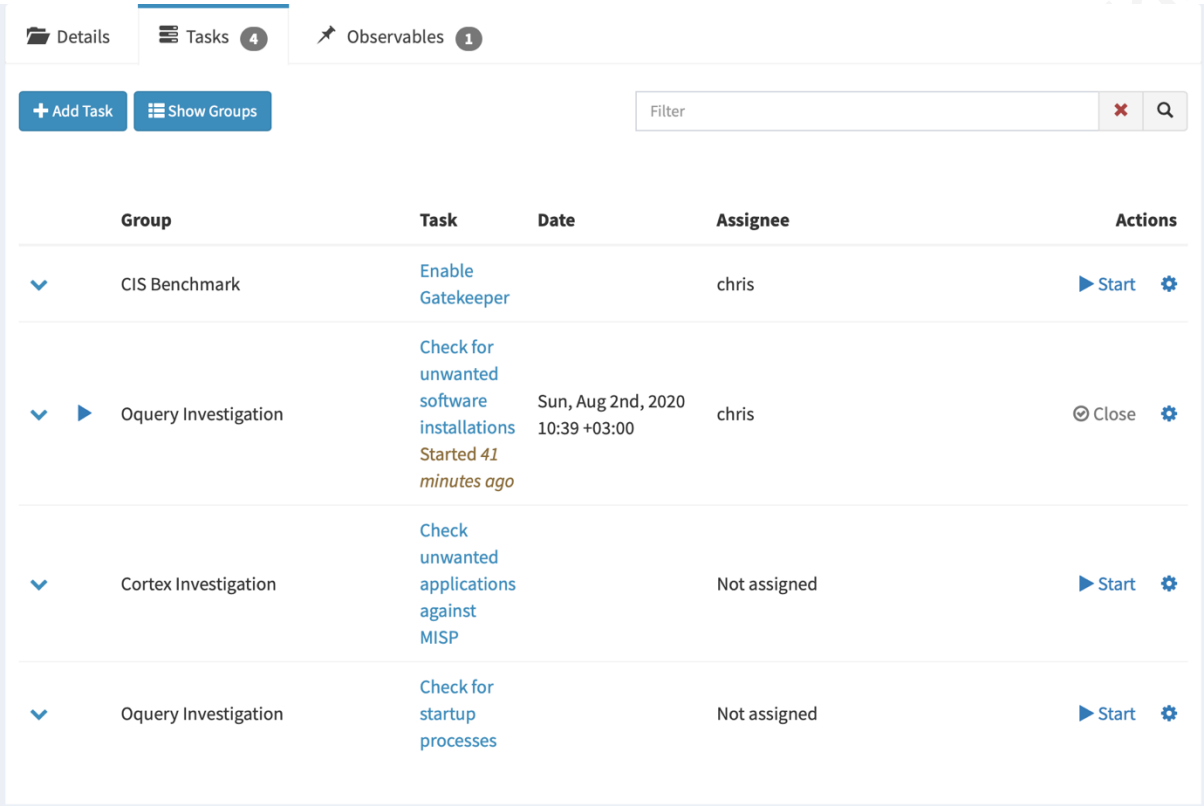

Figure 8. Alert task breakdown and assignments

An organization can choose how to describe the tasks themselves, and the assignee can add notes about the investigation as they see fit based on information obtained or actions taken. Figure 9 shows that a detailed process is being described and that the analyst determined that there is essential information to be shared.

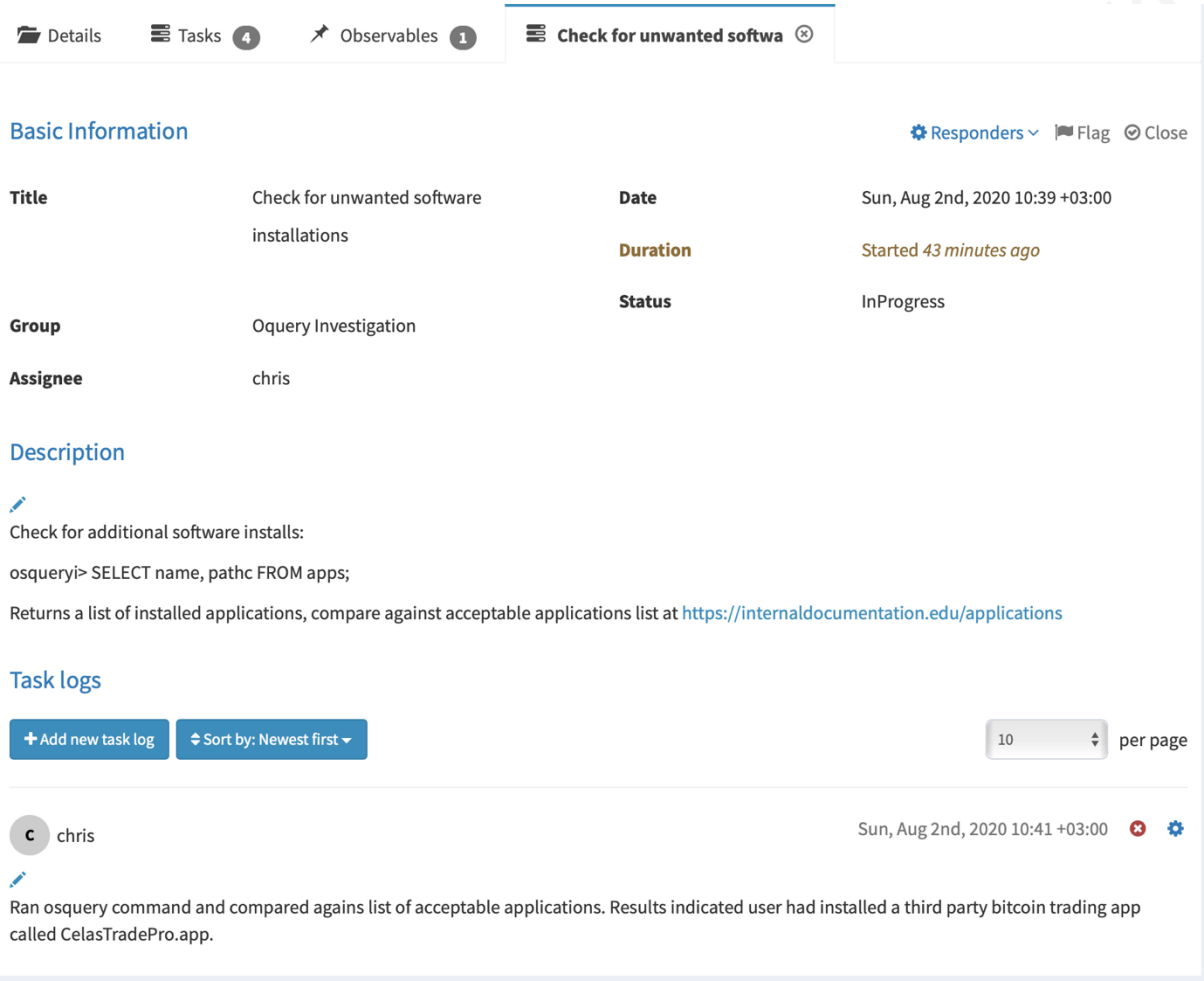

Figure 9. Example of analyst task completion.

In Figure 10, the analyst has found and uploaded a file, and its md5 has to be associated with the investigation. This file or hash value can now be forwarded to Cortex for analysis.

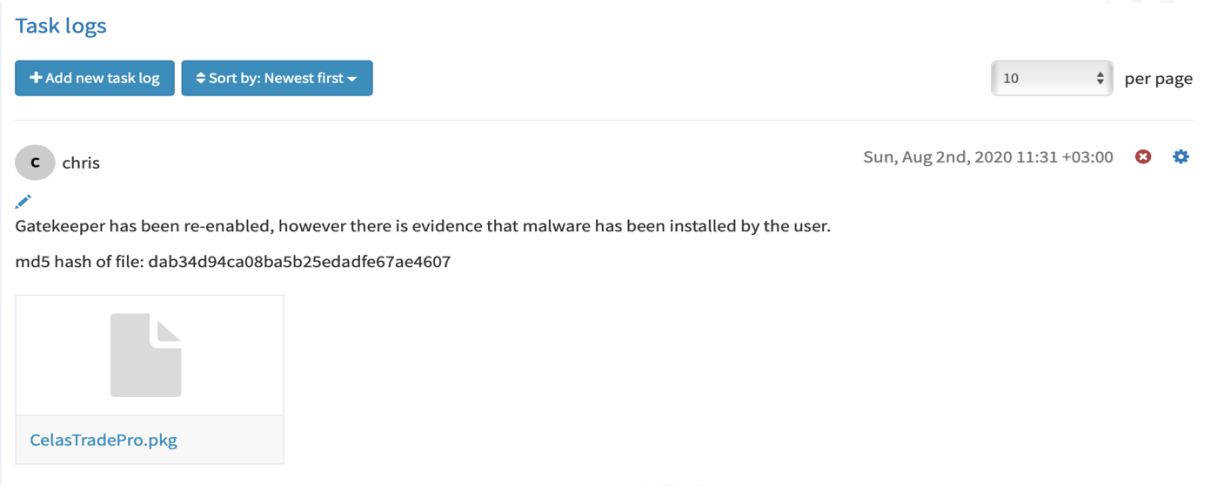

Figure 10. TheHive file upload

Figure 11 shows that analyzers have been configured for MISP and VirusTotal.

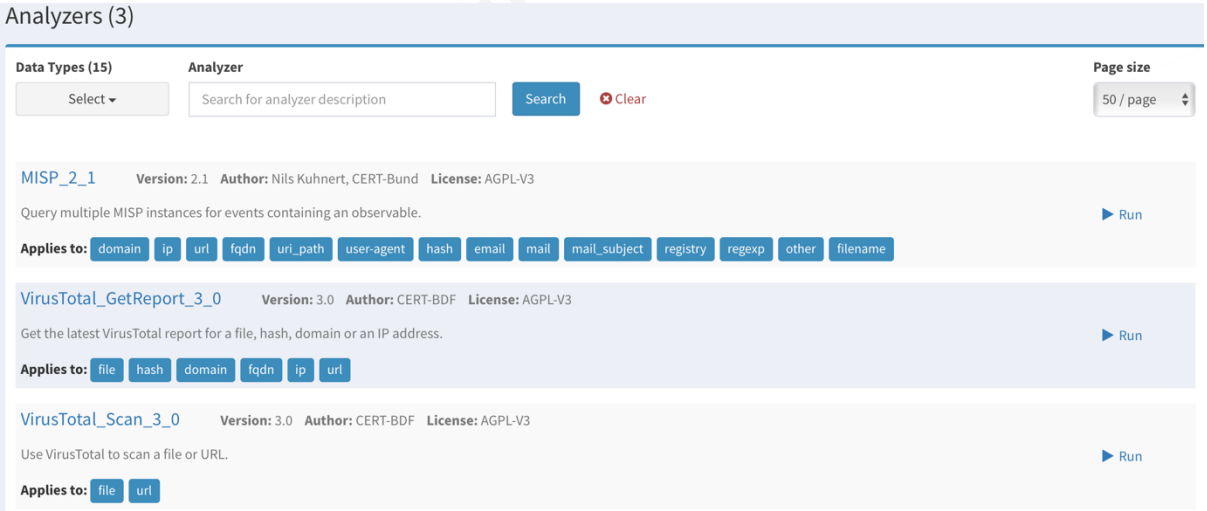

Figure 11. Cortex Analyzers

There are several ways to run analysis, all based on observable attributes: uploading the file directly or using the hash returns the following:

```
{
  "summary": {
    "taxonomies": [
      {
        "level": "malicious",
        "namespace": "VT",
```

```
        "predicate": "GetReport",
        "value": "29/59"
      }
    ]
  }
```

```
"FireEye": {
        "detected": true,
        "version": "32.31.0.0",
        "result": "Trojan.MAC.Lazarus.B",
        "update": "20200603"
      }
```

Based on this data the installation of unwanted software is confirmed, and the machine in question would likely be taken out of service for a full investigation.

As seen from these results, the Gatekeeper status, CIS Benchmark for MacOS 2.6.2, which was being monitored via Osquery, triggered an alert which was then forwarded to TheHive. Based on TheHive alert a new case is created and the tasks which require remediation can then be assigned and followed by analysts. The result of the analysis leads to a host that needs to be removed from service for detailed investigation.

This process improves the overall effort put into creating detailed remediation steps and advanced Osquery queries and Elastalert notifications. The CIS benchmarks are an important baseline for system security states but must be audited in such a way that we can monitor and understand change on an endpoint.

5. Analysis of Testing Results

The results show that tracking a change event from Osquery to TheHive is easy to create and follow. There is a lot of room for complexity, which could make this solution more robust as it is scaled to meet the needs of an organization. CIS Benchmarks, Osquery, Elastic Stack and TheHive become a compelling set of open-source tools that can enhance incident handling and incident investigation processes.

5.1 Expected Results

The ability set a system state and notify when that state changes, as well as give analysts simple, repeatable instructions for remediation, was an obvious outcome. Having

access to the desired state, alerting of change, and then, in TheHive, providing detailed remediation results based on CIS Benchmark documentation worked exceptionally well.

5.2 Unexpected Results

One unexpected result was that so many of the CIS benchmarks were unavailable for audit by Osquery. This dramatically weakens the solution as a means of ensuring CIS Benchmark standards are being audited. Creating and individualized approach using Python to bridge the gap between CIS Benchmarks and Osquery proved practical. This provided many additional insights into each of the benchmarks as each benchmark was deconstructed and its underlying system processes for accessing and manipulating the benchmark analyzed.

5.3 Value and discussion of results

The most significant value in this solution is both the cost and the ability to give analysts predefined and repeatable processes. There are many enhancements to TheHive and Osquery that could be implemented as the system matures in an organization. Additionally, the data forwarded from Elastalert can be automated further than what was presented here by using Cortex "Analyzers and Responders".

5.4 Beyond endpoints

Elastalert is not by any means limited to Osquery result logs. It can match any logged event and forward that to TheHive in real-time. Simply put, any device that has logs that can be forwarded can have Elastalert set up to evaluate and react to events.

5.5 Real-world value

A senior security analyst will spend a great deal of their time using and setting up this solution if it is done from scratch. I do not doubt that businesses with higher volumes of incidents would benefit from this solution if they do not have something in-house already. Depending on staffing levels and needs, the implementation of a solution like this could be of great value. Organizations may also find they have other tools that may fit into this solution to act in various roles of the remediation cycle. Osquery could be replaced with Wazuh or OSSEC or some other host intrusion detection system.

5.6 Future Research

There is a gap between what needs to be audited and what can be audited with third party-tools. Operating system vendors have their views and needs of security, which creates a rift between them and the cybersecurity industry. Moving beyond a solely declarative security structure to one that both declares and audits the assumptions of declarations is an excellent step to better ensuring that systems are in the state we want them to be. Much can be done to understand this gap on Windows, MacOS, and *nix operating systems.

6. Conclusion and Recommendations

Most Information Technology professionals would be capable of implementing this solution, and the benefit of the final product makes the process of investigations available to a broader group of individuals, especially those who may not have formal security training but can perform basic investigative tasks when good instruction is provided. The benefit of this is the ability to give staff real-life training of incident handling through well-documented processes and procedures.

While Osquery's ability to act as a watcher for many CIS Desktop Benchmarks is apparent, Osquery's inability to track settings that required OS-specific tools proved that there is a real gap in every security professionals ability to quickly find important security settings as they are obfuscated behind OS specific tools and commands. CIS solves this problem with its "CIS-CAT" tool (Center for Internet Security, 2020); however, this tool is not free or open-source. The shim that was written to bridge the gap between CIS Benchmarks and Osquery is a viable and straightforward solution that is designed to make OS system calls that are then translated into XML plist files that can be read by Osquery. Creating a similar set of commands to fill the void between CIS Benchmarks and Osquery should be very attainable using Python, bash, swift, or PowerShell should an organization be using a Windows solution.

Organizations looking to standardize and broaden their abilities for incident response and investigation will benefit from using CIS Benchmarks, Osquery, Elastic Stack, and TheHive as a foundation for creating sound and repeatable incident handling processes, provided they are prepared to develop some in-house methods for bridging the gap between the CIS Benchmark requirements and Osquery.

References

Center for Internet Security. (2020). CIS SecureSuite Membership Terms of Use. Retrieved from Center for Internet Security: https://www.cisecurity.org/cis-securesuite/cis-securesuite-membership-terms-of-use/

Center for Internet Security. (2020). CIS SecureSuite Membership Terms of Use. Retrieved from Center for Internet Security: https://www.cisecurity.org/cis-securesuite/cis-securesuite-membership-terms-of-use/

Center for Internet Security. (2020). CIS-CAT Pro. Retrieved from cisecurity: https://www.cisecurity.org/cybersecurity-tools/cis-cat-pro/

Center for Internet Security. (2020). Securing Apple OS. Retrieved from Center for Internet Security: https://www.cisecurity.org/benchmark/apple_os/

Centers for Medicare and Medicaid Services. (2007, 03). HIPAA Security Security Series. Retrieved from Centers for Medicare and Medicaid Services: https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/administrative/securityrule/techsafeguards.pdf

Chuvakin, A. (2015, 09 11). Security: Automate And/Or Die? Retrieved from Gartner Blogs: https://blogs.gartner.com/anton-chuvakin/2015/09/11/security-automate-andor-die/

Deloitte. (2016). Five Essential Steps to Improve Cybersecurity: Trekking towards a more secure, vigilant and resilient organisation. Retrieved from Deloitte: https://www2.deloitte.com/content/dam/Deloitte/nz/Documents/risk/cybersecurity-new-zealand-five-essential-steps.pdf

Facebook. (2020). Welcome to osquery. Retrieved from Read the Docs: https://osquery.readthedocs.io/en/latest/

Hurless, C. (2019). Exploring Osquery, Fleet, and Elastic Stack as an Open-source solution to Endpoint Detection and Response. Retrieved from SANS Reading Room: https://www.sans.org/reading-room/whitepapers/detection/exploring-osquery-fleet-elastic-stack-open-source-solution-endpoint-detection-response-39165

Microsoft. (2020). Center for Internet Security (CIS) Benchmarks. Retrieved from Microsoft: https://docs.microsoft.com/en-us/microsoft-365/compliance/offering-cis-benchmark?view=o365-worldwide

Mike Arpaia. (2017, 10 18). Kolide Fleet: An open-source osquery fleet manager. Retrieved from Kolide Blogs: https://blog.kolide.com/kolide-fleet-an-open-source-osquery-fleet-manager-26e8094fab

Rouse, M. (2016, 09 22). Definition: Elastic Stack. Retrieved from Tech Target: https://searchitoperations.techtarget.com/definition/Elastic-Stack

TheHive. (2020). TheHive-Project. Retrieved from GitHub: https://github.com/TheHive-Project/TheHive

TheHive project. (2020). Retrieved from TheHive project: https://thehive-project.org

U.S. Bureau of Labor Statistics . (2020, 04 10). Retrieved from Information Security Analysts : Occupational Outlook Handbook: https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm

Yelp. (2020, 04 16). ElastAlert - Easy & Flexible Alerting With Elasticsearch. Retrieved from Elastalert Read the Docs: https://elastalert.readthedocs.io/en/latest/elastalert.html