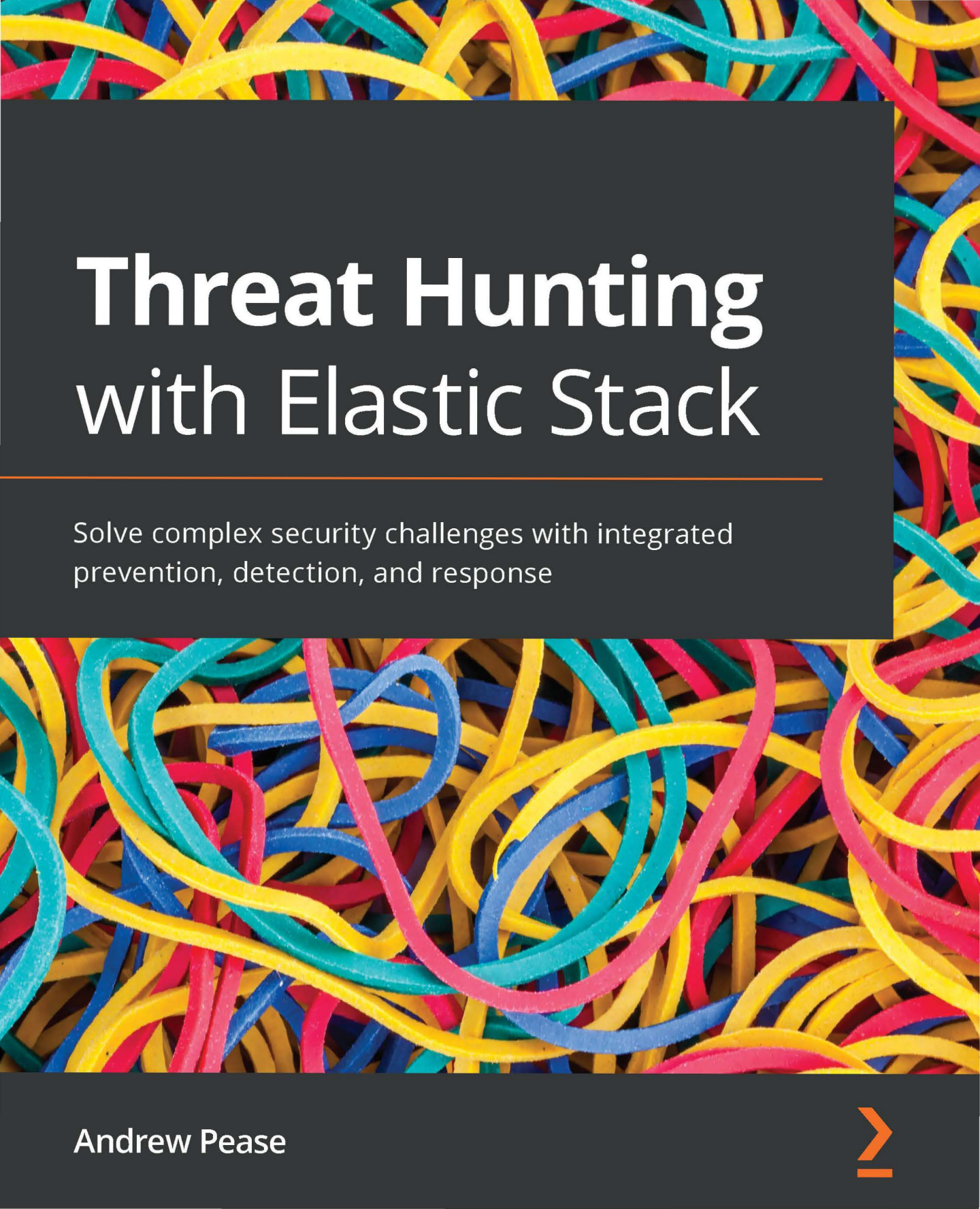
Threat Hunting
with Elastic Stack
Solve complex security challenges with integrated prevention, detection, and response
Andrew Pease

Threat Hunting with Elastic Stack

Solve complex security challenges with integrated prevention, detection, and response

Andrew Pease

Packt

BIRMINGHAM—MUMBAI

Threat Hunting with Elastic Stack

Group Product Manager: Wilson Dsouza
Publishing Product Manager: Yogesh Deokar
Senior Editor: Rahul Dsouza
Content Development Editor: Sayali Pingale
Technical Editor: Shruthi Shetty
Copy Editor: Safis Editing
Project Coordinator: Neil Dmello
Proofreader: Safis Editing
Indexer: Tejal Soni
Production Designer: Shankar Kalbhor

First published: July 2021

Production reference: 1210721

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

978-1-80107-378-3

www.packt.com

Contributors

About the author

Andrew Pease began his journey into information security in 2002. He has performed security monitoring, incident response, threat hunting, and intelligence analysis for various organizations from the United States Department of Defense, a biotechnology company, and co-founded a security services company called Perched, which was acquired by Elastic in 2019. Andrew is currently employed with Elastic as a Principal Security Research Engineer where he performs intelligence and analytics research to identify adversary activity on contested networks.

He has been using Elastic for network and endpoint-based threat hunting since 2013, He has developed training on security workloads using the Elastic Stack since 2017, and currently works with a team of brilliant engineers that develop detection logic for the Elastic Security App.

8
The Elastic Security App

We have spent a great amount of time leading up to this, the Elastic Security app. The Elastic Security app is the central point for all security-related data and information. This was formerly referred to as the Elastic SIEM (Security Information and Event Management) and is how we can explore specific host and network data, analyze security events, leverage the detection engine, manage cases, and dig deep into data with timelines.

In this chapter, you will learn how to use the Elastic Security app to identify abnormal endpoint and network traffic, perform tailored detections of those events, and create detection logic based on your analysis.

In this chapter, we'll go through the following topics:

- The Elastic Security app overview
- The detection engine
- Hosts
- Network
- Timelines
- Cases
- Administration

Technical requirements

In this chapter, you will need to have access to the following:

- The Elastic and Windows virtual machines built in *Chapter 4, Building Your Hunting Lab – Part 1*
- A modern web browser with a UI

Check out the following video to see the Code in Action:
`https://bit.ly/2UODWi6`

The Elastic Security app overview

The **Elastic Security app** is the central point for Elastic's security solution. It includes a security news feed, host and network data, detections, timelines, cases, and an abstracted view into the administration of the Elastic endpoint configuration.

To get to the **Elastic Security** app, click on the hamburger menu and select **Overview** under the **Security** heading. This landing page will show you the highlights of the events that are in the security app. From here we can jump into specific sections that show their relevant data:

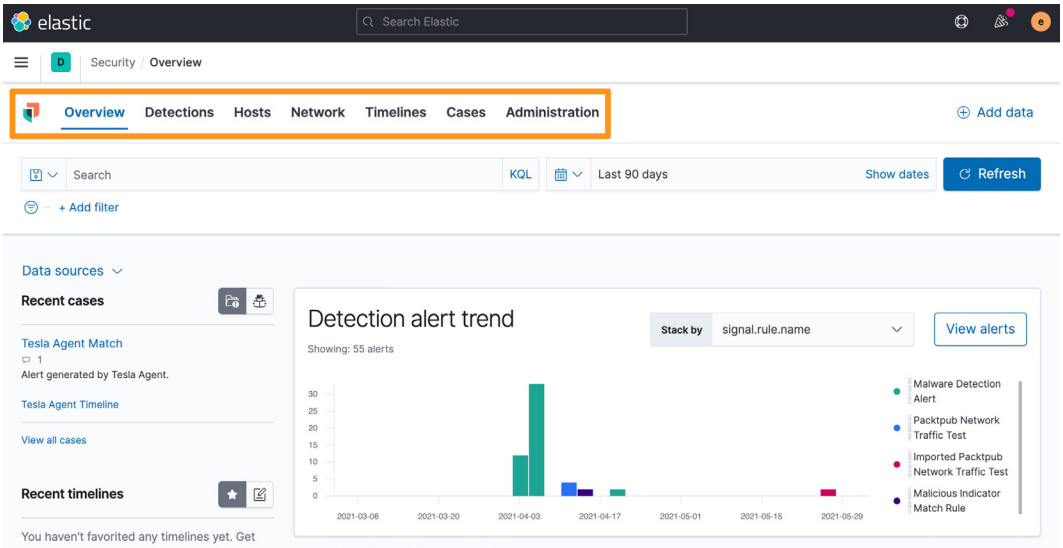

Figure 8.1 – Security app overview

You can scroll through this **Overview** section to get a high-level look at the different types of data that are reflected in the app. Most notably, at the bottom of the **Overview** page, there is a breakdown of the different datasets, separated by host and network, that we're sending into the Elastic Stack:

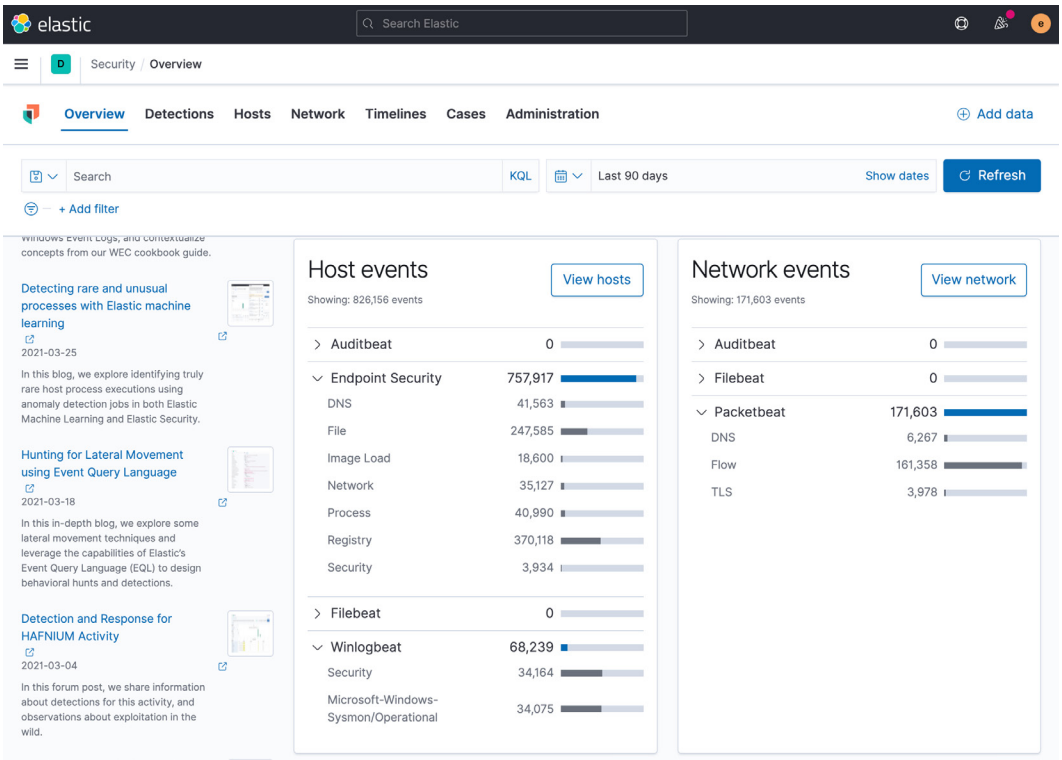

Figure 8.2 – Security app datasets

This **Overview** page allows us to see important information that is relevant across the entire Security app. To get additional information, we will use the section tabs at the top of the **Overview** page.

In this section, we learned how to get to the Elastic Security app **Overview** page. Next, we'll explore the detection engine.

The detection engine

The **Detections** section is used to investigate and create detection logic. Detection logic can be the results of a malware hit from a signature or behavior as well as potentially malicious activity. As of Elastic version 7.12, the detection engine has over 500 pre-built rules that are created by the Elastic Intelligence and Analytics Team and the Elastic community:

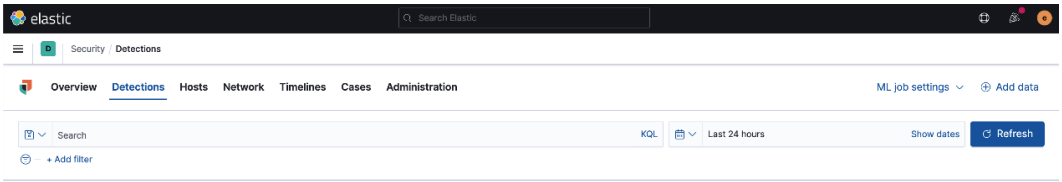

Figure 8.3 – Detection section

Keeping with the theme across the rest of Kibana, you can apply specific queries directly into the **Detections** section or apply a date picker selection to narrow any searches.

In this section, we began to explore the detection engine of the Elastic Security app. In the next section, we'll learn about managing the detection rules.

Managing detection rules

One of the most powerful features of the Elastic Security app is the detection rules. **Detection rules** are pre-configured queries that compare events from various data sources to identify non-signature-based malicious activity.

As an example, perhaps you want to know whether you have systems that are receiving **Remote Desktop Protocol** (**RDP**) connections from outside your network, whether someone is trying to brute force through **Secure Shell** (**SSH**), or someone is trying to export your Windows Registry Hive. These things could be malicious in your environment, but these are events that will not be detected by traditional anti-virus.

Elastic has released hundreds of open source rules for the detection engine (`https://www.elastic.co/blog/elastic-security-opens-public-detection-rules-repo`) and has made them all available on GitHub (`https://github.com/elastic/detection-rules`). As I mentioned before, there are 546 rules available for free. Not only are the rules available on GitHub, but they are also automatically loaded in Kibana. You may remember in *Chapter 5*, *Building Your Hunting Lab – Part 2*, we loaded all of the prebuilt rules:

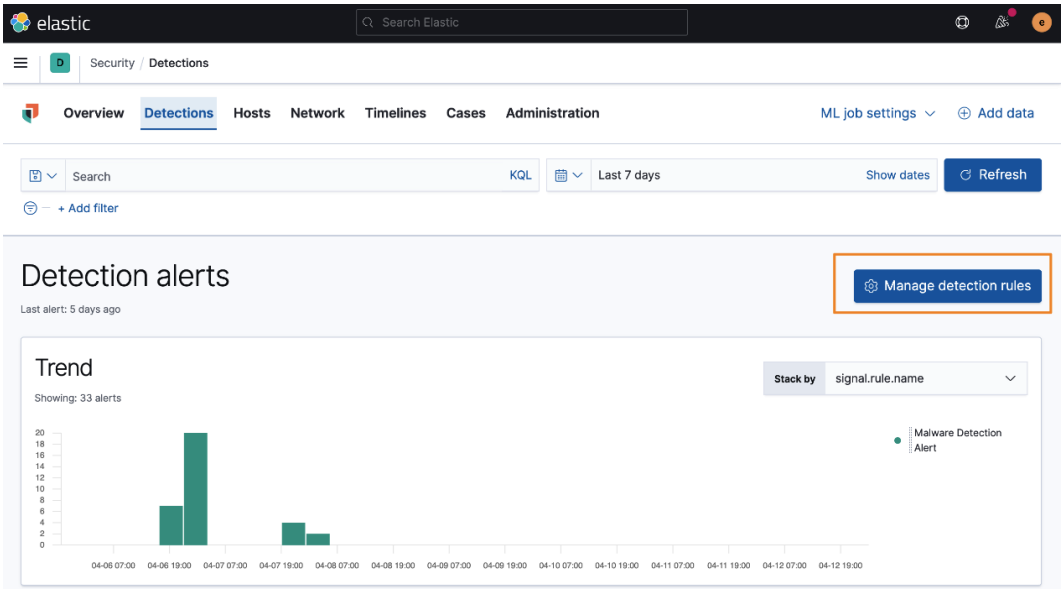

Figure 8.4 – Manage detection rules

If you click on the **Manage detection rules** button, it will open the **Detection Rules** management section.

From here, we can see the **Rules**, **Rules Monitoring**, and **Exception Lists** tabs.

Rules

From this tab, we can enable different rules, search for rules by their names or tags, or dig into the rules to learn more about them:

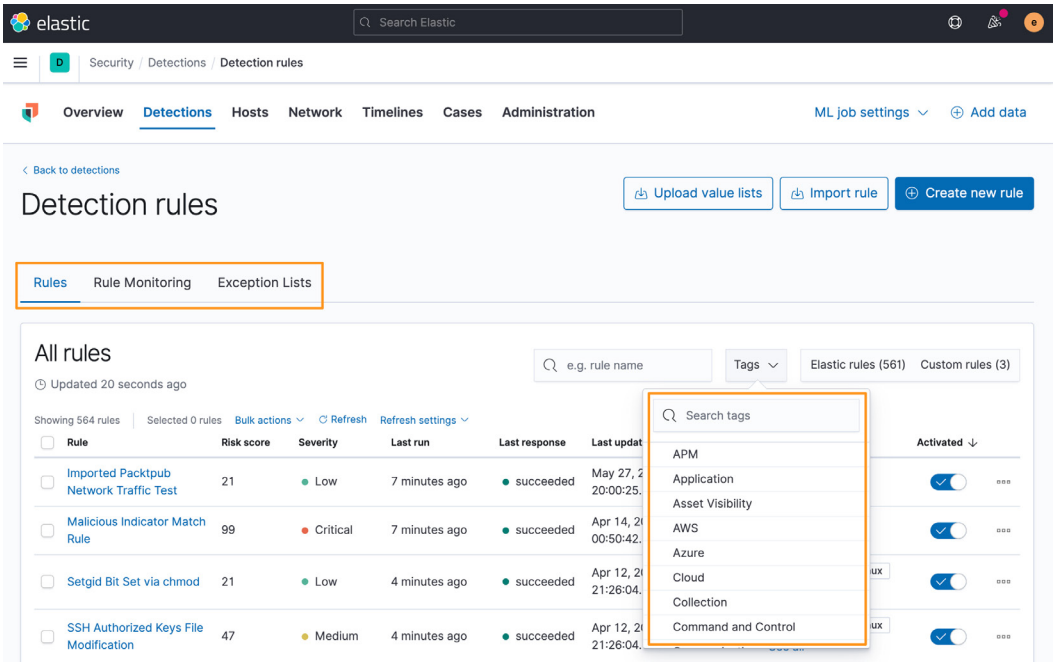

Figure 8.5 – Rules overview

Clicking on a rule will open the rule so that you can inspect the metadata about the rule, where the data must come from, what the query is, and so on:

> **Important note**
>
> You cannot modify the Elastic-provided rules, but you may make a duplicate and modify the duplicate if necessary.

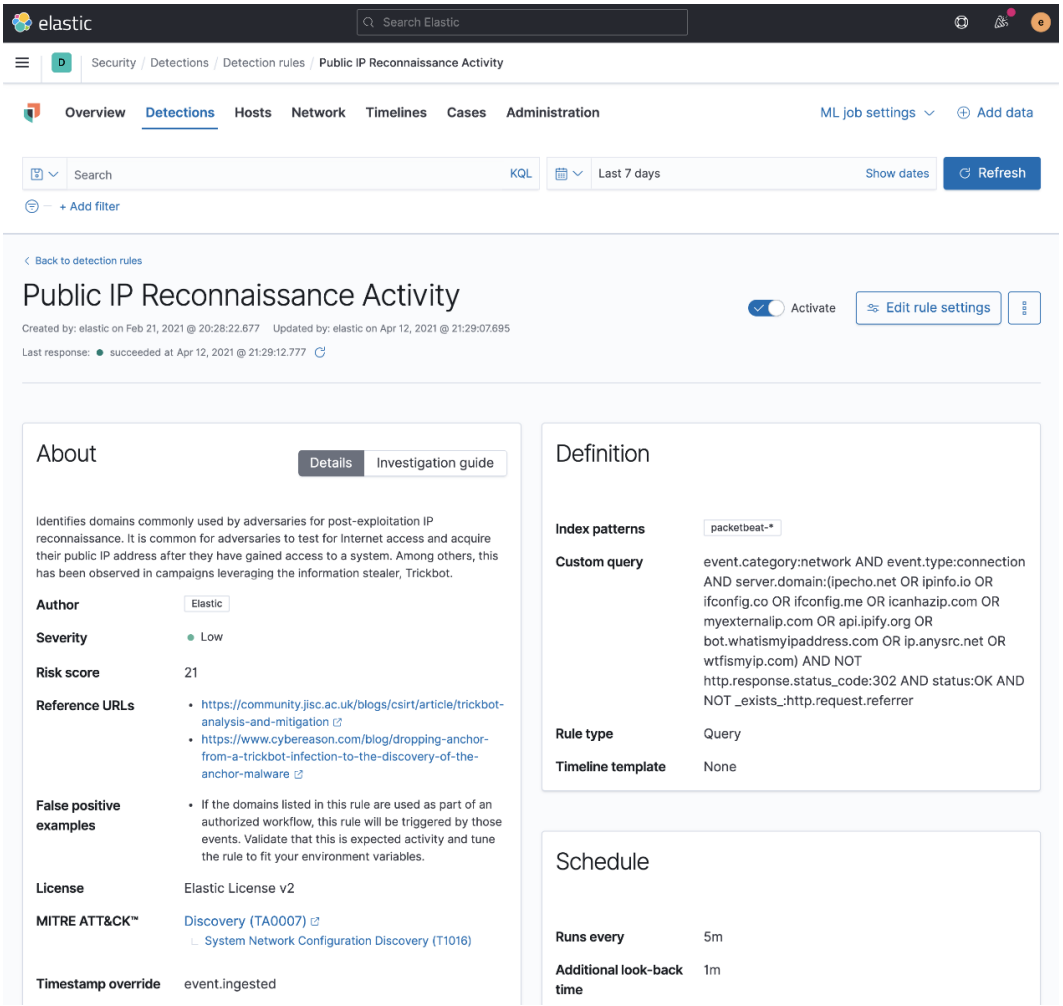

Figure 8.6 – Public IP Reconnaissance Activity network rule

This overview of the rules will help you determine what rules you want to enable and what rules don't make sense in your environment. As an example, if you aren't collecting cloud security rules, or Linux or macOS events, it doesn't make sense to enable those rules.

Rule monitoring

Clicking on the **Rule Monitoring** tab will give you a view of the amount of time the rules take to run:

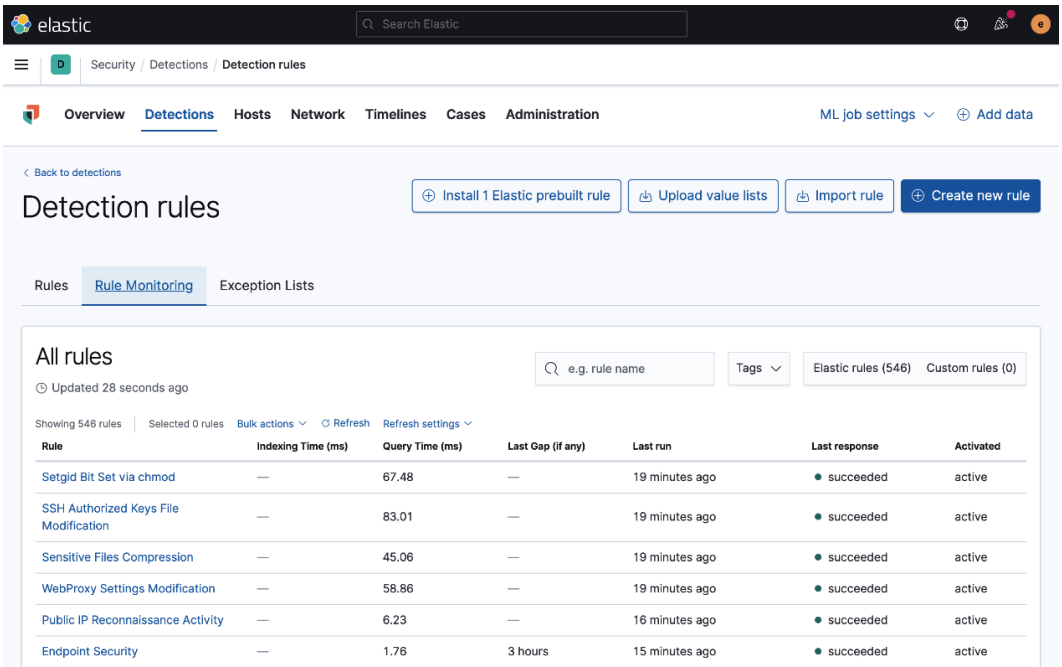

Figure 8.7 – Rule Monitoring

Rule Monitoring can be helpful if you're noticing a performance impact; you can look at what rules are taking the most time and decide if you need to increase the resources for your stack or if those rules are needed.

Exception Lists

The **Exception Lists** tab is where you can view any exceptions you've created for rules or the endpoint:

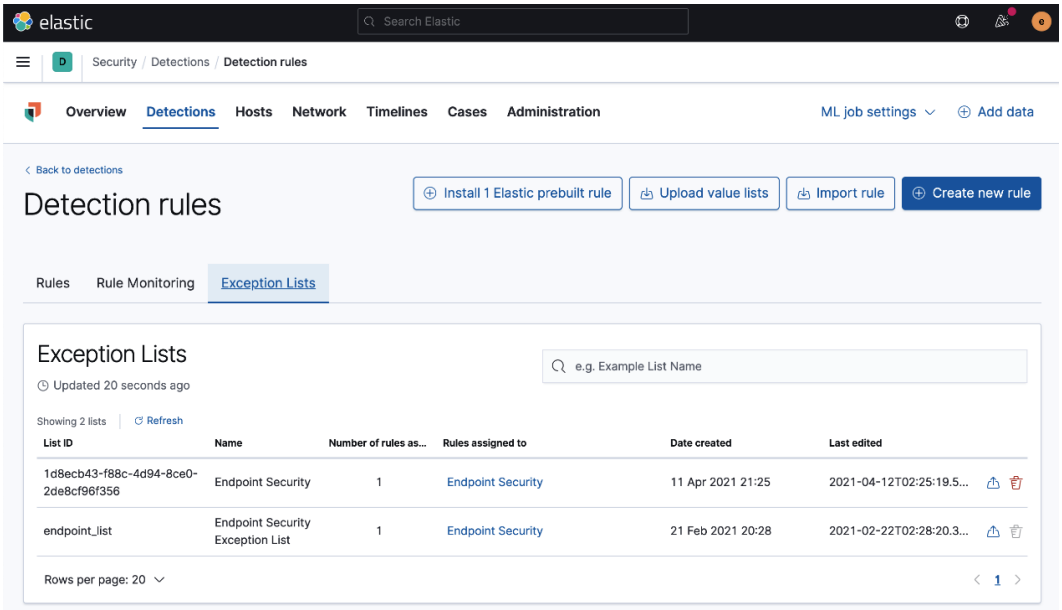

Figure 8.8 – Exception Lists

We'll talk more about the exception framework in the *Event actions* section, a bit further on in the chapter.

Creating a detection rule

I mentioned that Elastic provides 546 rules for you, but we can also create rules that fit a specific threat profile for our environment.

Rules can either be created using a Python module that Elastic provides and has made available, or be created and made available through Kibana.

To get started, click the blue **Create new rule** button:

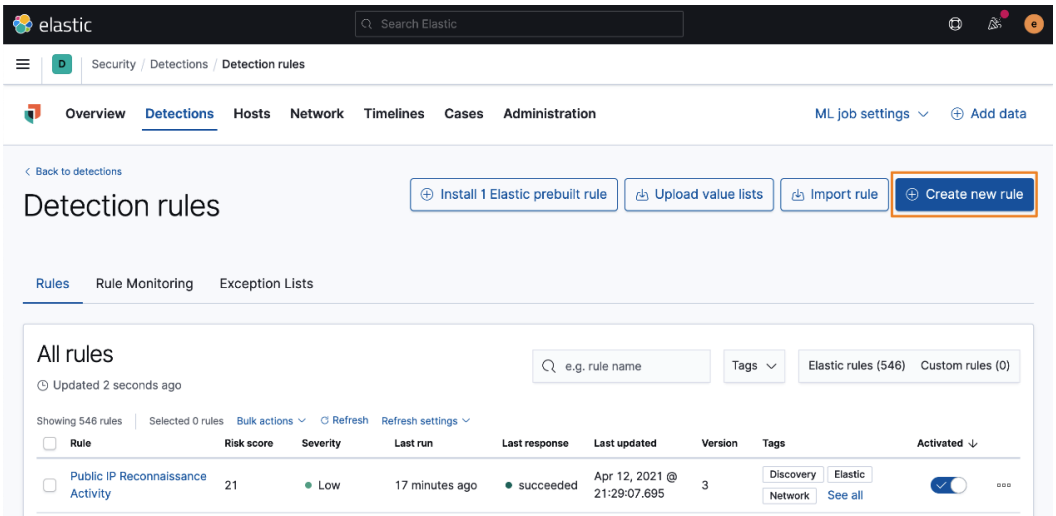

Figure 8.9 – Create new rule

There are five types of rules that you can create:

- A **Custom query** (KQL or Lucene rule)
- A **Machine Learning** rule
- A **Threshold** rule
- An **Event Correlation** rule
- An **Indicator Match** rule

Next, we'll walk through the creation of these rule types.

> **Important note**
>
> For the detection rules, the first section (**Define rule**) will change depending on the rule type that you're going to use, but the three follow-on sections will all be the same (**About rule**, **Schedule rule**, and **Rule actions**). We'll go all the way through the four sections for the **Custom query** type and just the first section for the other four rule types.

Custom query rule

By default, the rule type will be **Custom query**. This is how you'd create a KQL or Lucene query.

Below the rule type, you can select what index patterns your data will be in. By default, all of the possible index patterns are added. As we only have Endpoint, Packetbeat, and Winlogbeat data, we can safely remove the unused datasets.

Next, we can enter our query. As an example, I am looking for network connections to the domain `packtpub.com`:

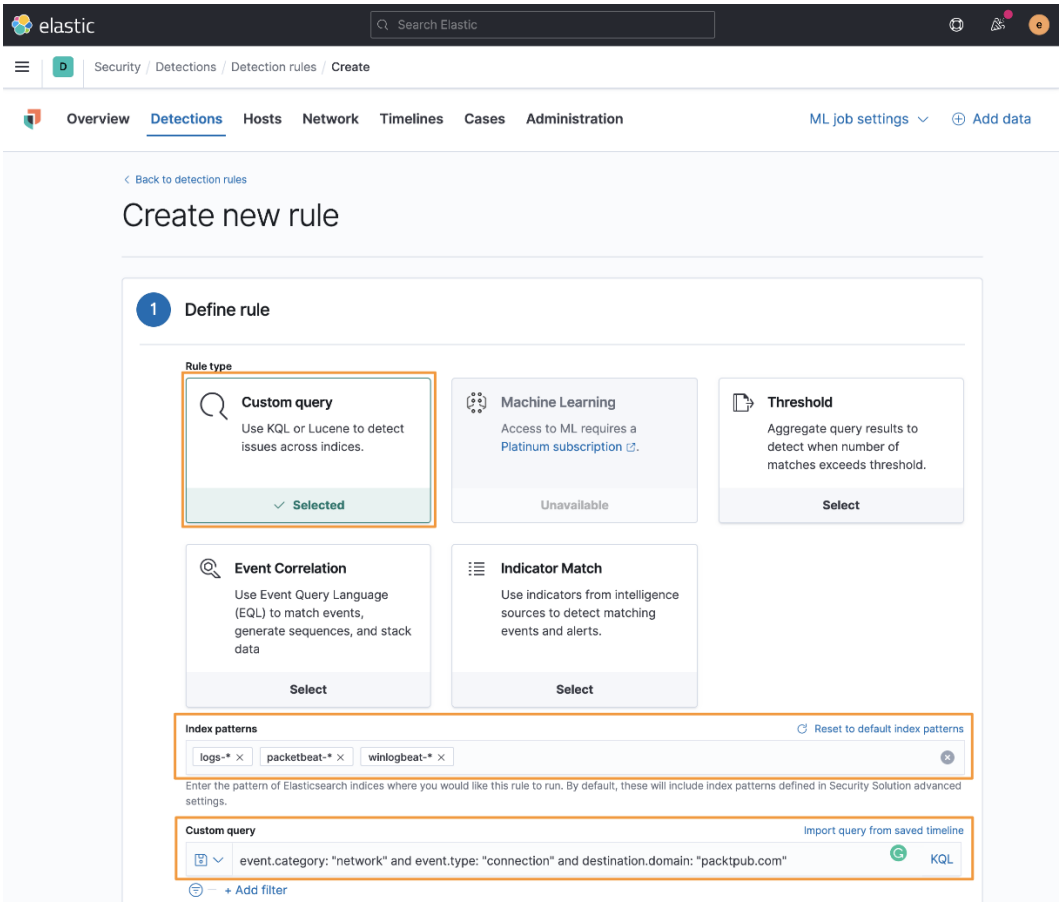

Figure 8.10 – Custom query

Here we can see what the first part of the rule will look like.

Elastic has added the **Preview results** feature so you can test to see whether your query is working as intended:

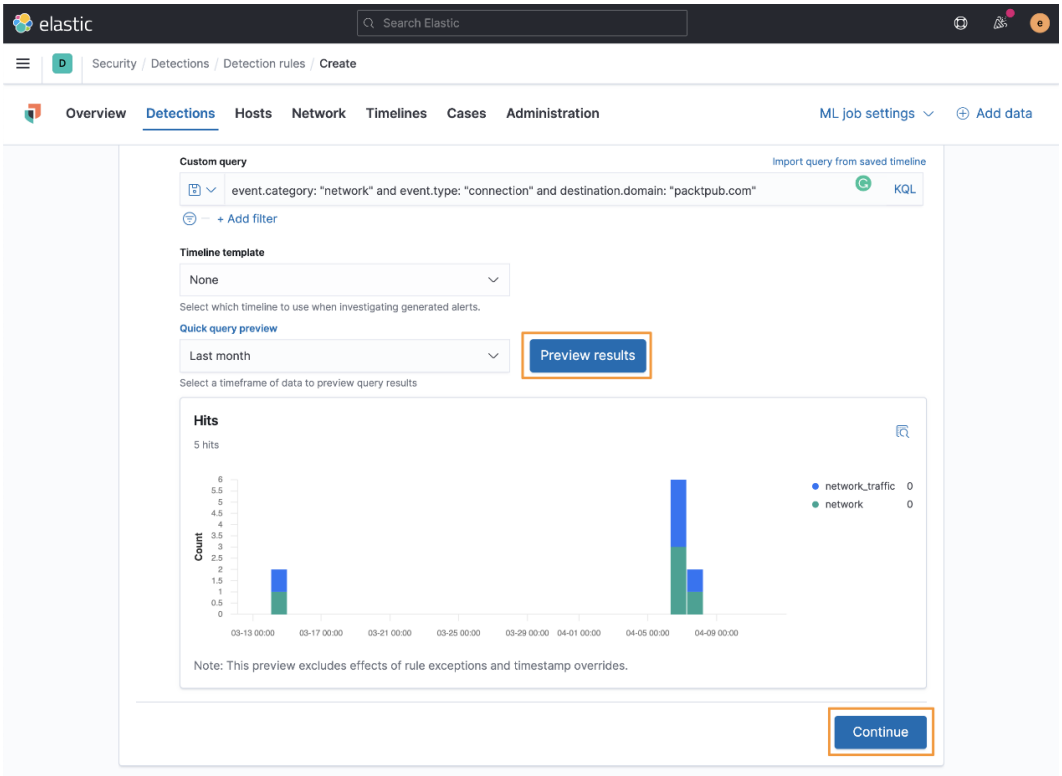

Figure 8.11 – Preview results

After we've set up the rule type and the data sources, written our query, and tested the results, we can click on **Continue** to move onto the next step.

In the second section, we can define the name, description, severity, and risk score, and add any organization tags:

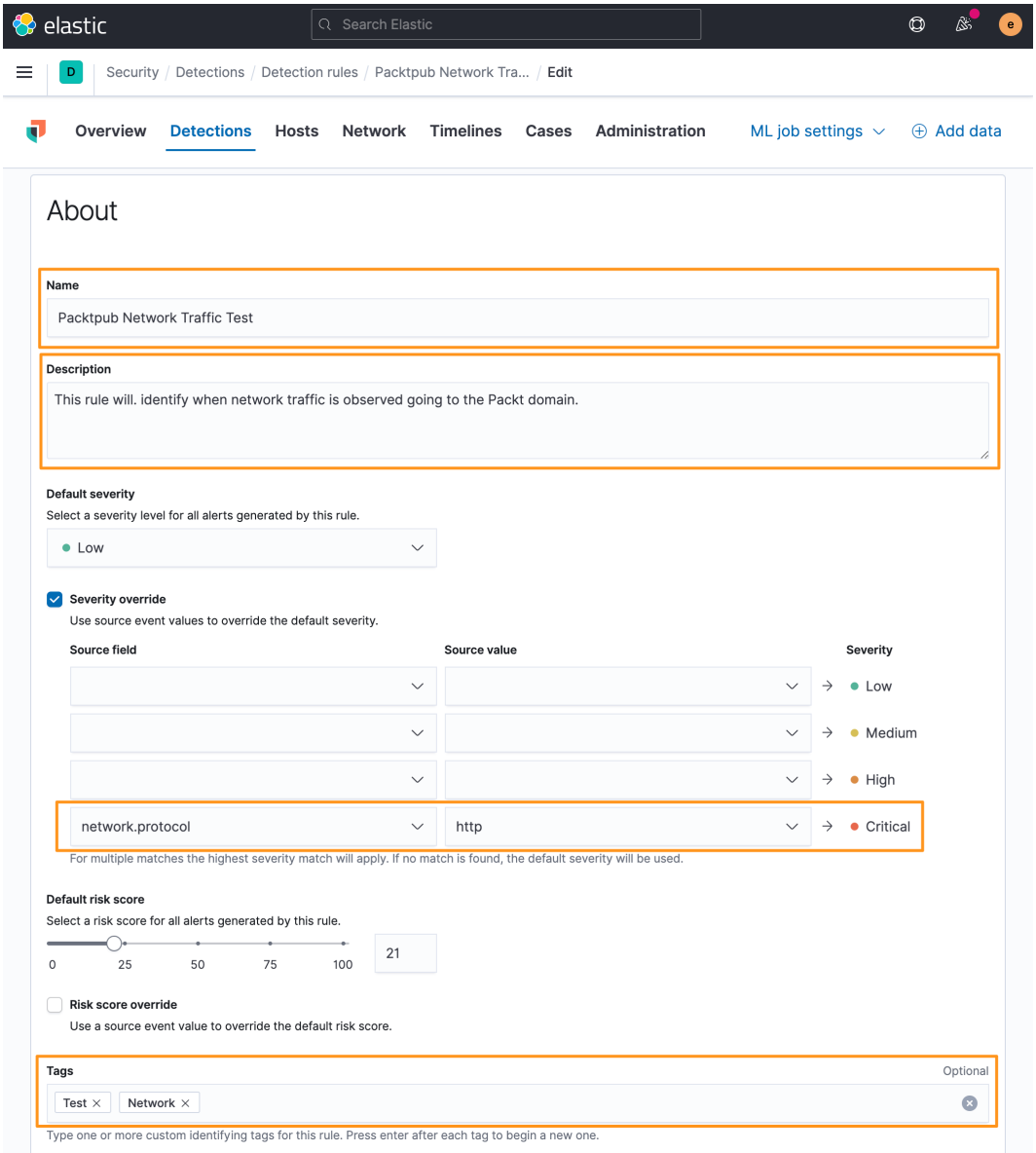

Figure 8.12 – Rule description

The default severity is **Low** and the risk score is **21**. As you change the severity, the risk score will automatically adjust. This can also be manually tuned if you have organizational policies that dictate a severity and risk matrix.

You can also override the defaults for severity and risk. In the preceding figure, I have changed the severity from **Low** to **Critical** if the network traffic is unencrypted over the HTTP protocol instead of TLS.

Clicking on the **Advanced** settings dropdown, we can add some additional information about the rule. These settings are optional:

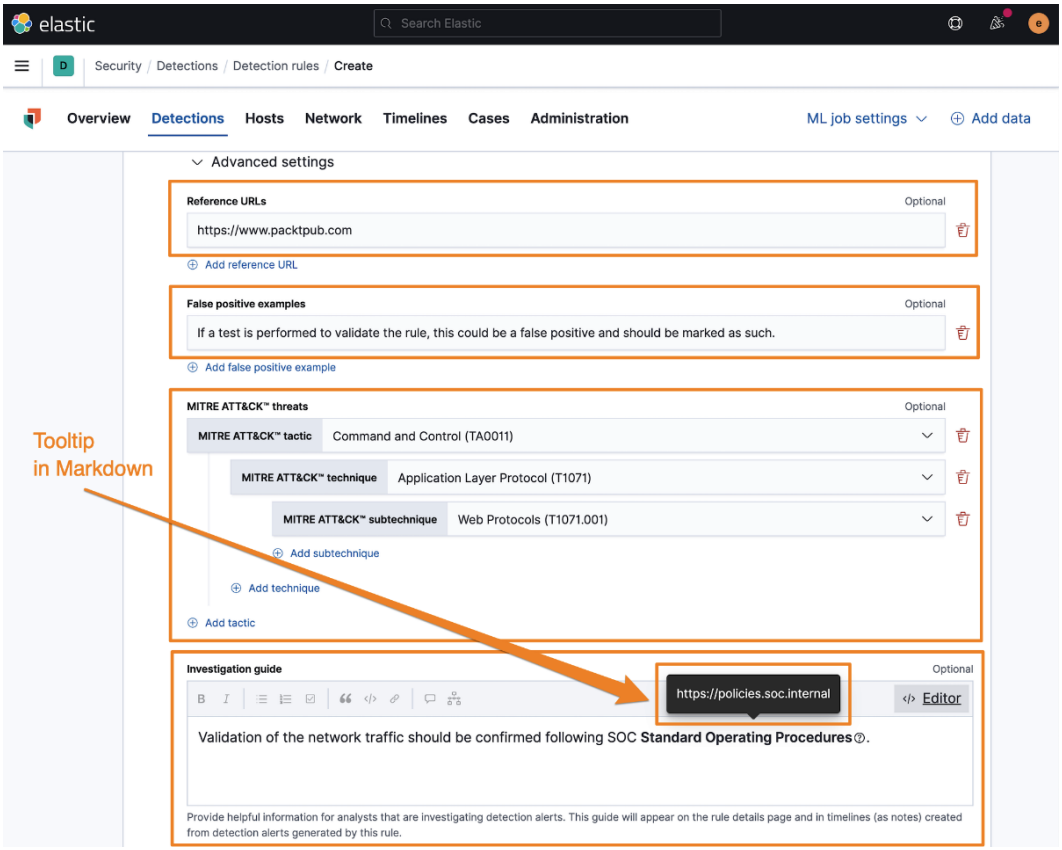

Figure 8.13 – Advanced settings

Here we can define any reference URLs that can help provide some context when performing an investigation. I like to put the source of the threat reports that may have led to the rule creation here.

We can provide some false positive examples.

We can add MITRE ATT&CK tactics, techniques, and subtechniques. The ATT&CK model was covered in *Chapter 1*, *Introduction to Cyber Threat Intelligence, Analytical Models, and Frameworks*.

We can also add investigative notes. This can be used to link to organization-specific documentation, points of contact, and so on. This guide renders GitHub-flavored Markdown (`https://github.github.com/gfm/`), which is helpful to use for inserting hyperlinks and tooltips.

GitHub-flavored Markdown can create hyperlinks using `[URL Text](url)` and tooltips with `!{tooltip[anchor text](helpful description)}`.

Moving on, we can populate the **Author** field and add the appropriate **License** details:

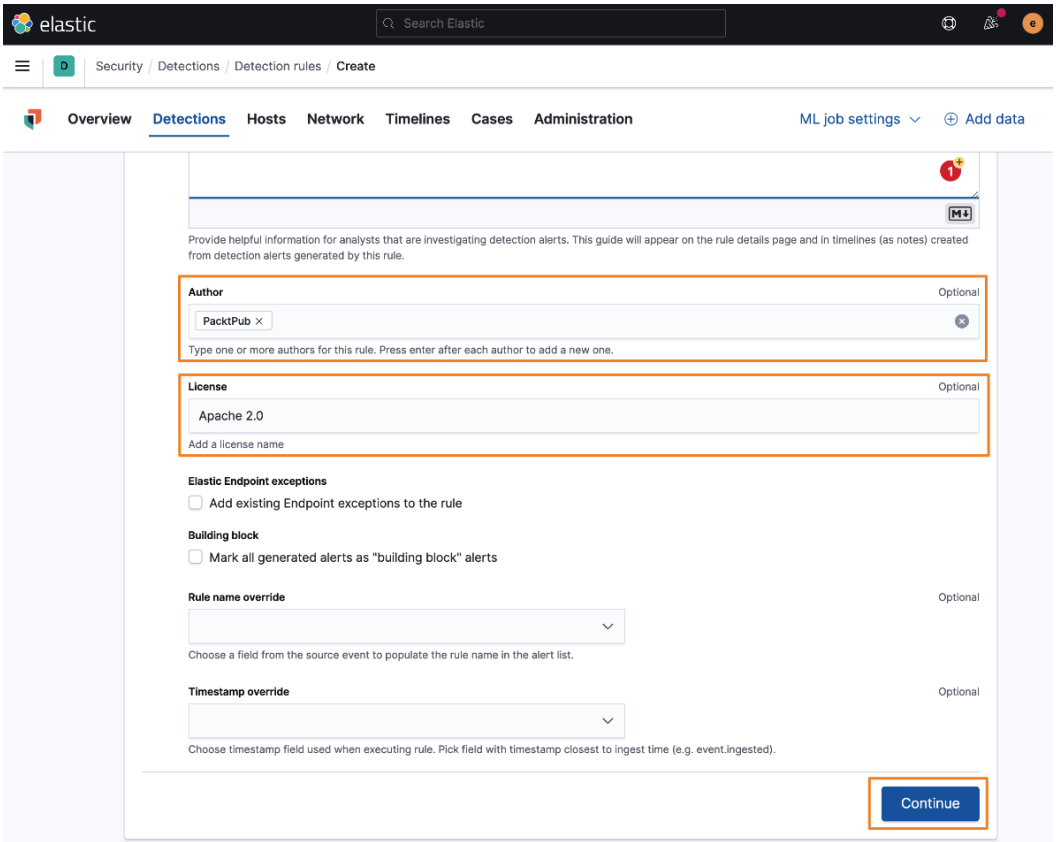

Figure 8.14 – Detection alerts trend sorting

We can also choose to apply any exceptions to this rule; we don't have any yet, but if we wanted to apply exceptions, we could check that box.

Building block rules are low-risk rules that we can create that will still write to the rules indices but not be displayed in the **Alerts** table in the main **Detections** view. This is helpful if you want to create a rule for context for other rules but not clutter up your view.

Rule name override will allow you to override the rule name we set before with the value of a field from the event. As an example, we could use the `destination.domain` field to name this rule `packtpub.com` when it is displayed in the **Alerts** table.

If we want to use a different timestamp than the default `@timestamp`, we can define that here.

Next, we can click **Continue** to move on.

We can define the schedule for the rule. This is how often the rule will run and how far back it should look. The lookback is to ensure there aren't specific events that happen to fall between the rule executions that could be missed.

The default is to run every 5 minutes with a 1-minute lookback. I prefer to change this to run every 9 minutes with a 1-minute lookback:

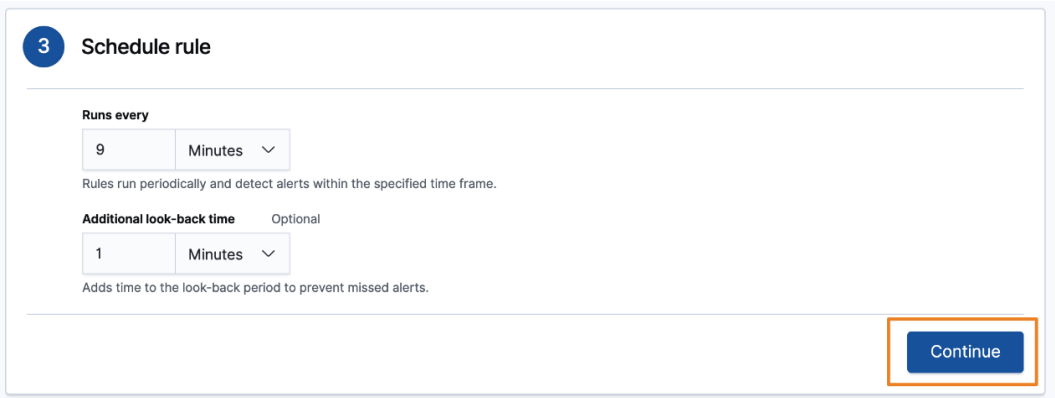

Figure 8.15 – Schedule the rule

After we have set the schedule and the lookback, we can click **Continue** to move on to **Rule actions**.

We can define how often the rule actions are performed, but they allow you to send notifications to third-party services. Actions to these external services are a paid feature.

Finally, we can create the rule. I always prefer to create the rule without activating it (meaning it will generate an event in the **Detections** section) so I can take one final look at the completed rule before I run it:

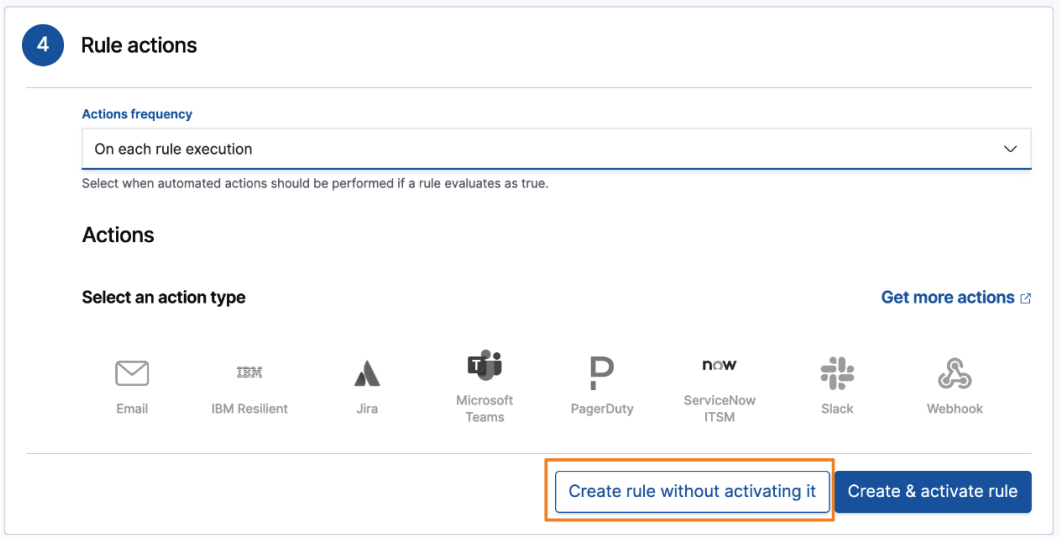

Figure 8.16 – Create the new rule

Once we've created the new rule, we'll land back on the **Detection rules** management page. We can click on **Custom rules** to view our new rule:

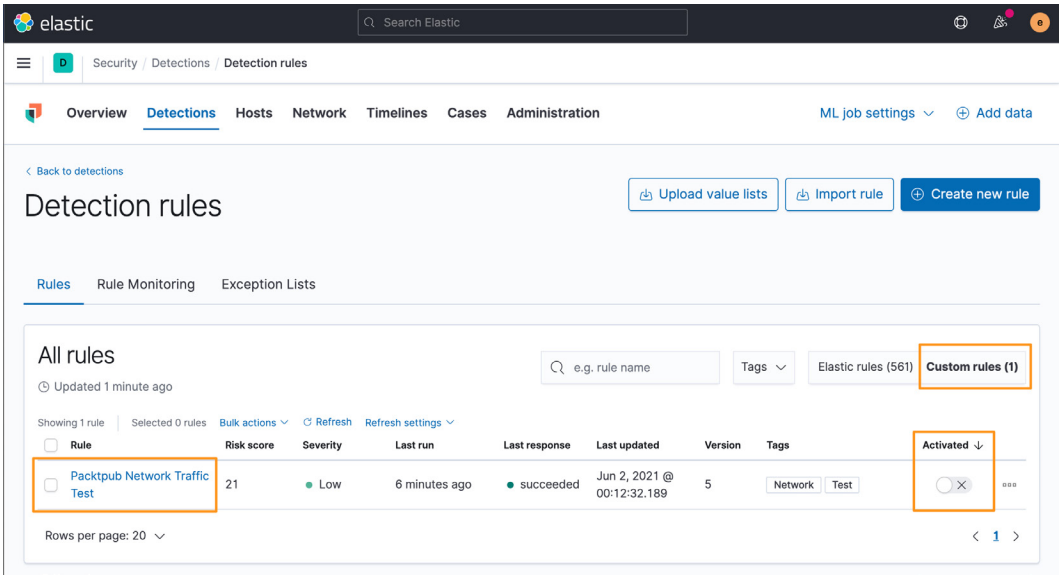

Figure 8.17 – View custom rules

Clicking on the rule name will open the rule and we can make a final check, make any necessary changes, and activate (or deactivate) it when you are ready to generate events with it:

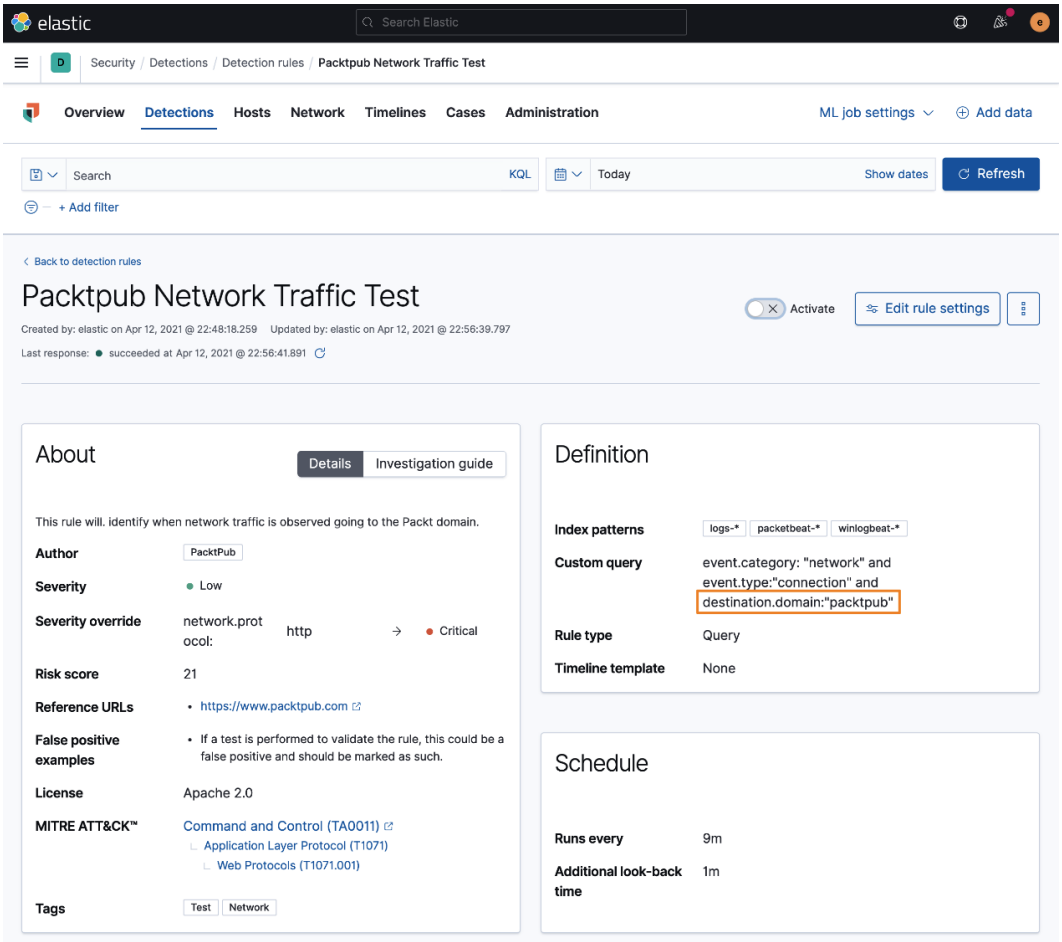

Figure 8.18 – Reviewing the new rule

As a real-world example, you notice in the preceding figure that the destination domain isn't right. I can click on the **Edit rule settings** button and adjust the domain to include the `.com` **Top-Level Domain** (**TLD**):

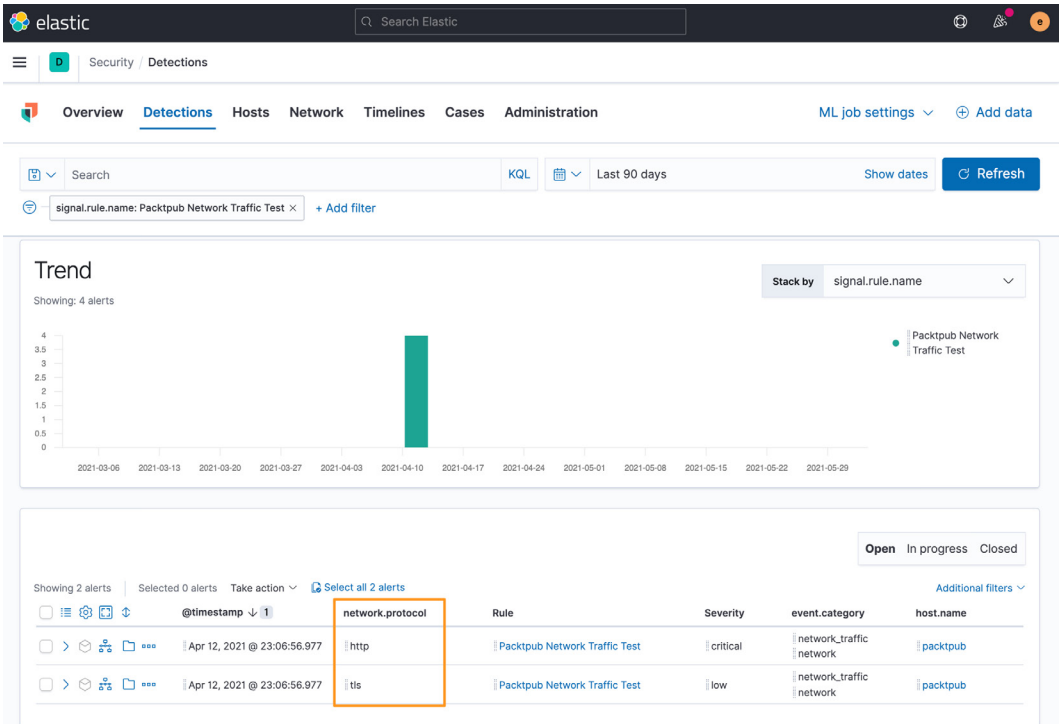

Figure 8.19 – Packtpub network rule execution

After adjusting my mistake, I can see an execution of the rule and we can even see the severity override we did for the HTTP connection.

We created a custom query rule with KQL; next we'll explore the machine learning rule.

Machine learning rule

Machine learning rules are available in the detection engine; however, as we discussed in *Chapter 7*, *Using Kibana to Explore and Visualize Data*, it is a licensed feature, so we'll not be able to explore them with our lab environment.

If you do add a license for your lab, you will be able to enable machine learning detection rules or create your own.

Next, we'll discuss threshold rules.

Threshold rules

Threshold rules are run against indices and then create an alert when the defined number of events occurs.

Using the **Group By** field in the **Define rule** section will create Boolean logic using `AND` for multiple fields that you define. You can also leave this blank and it will use the threshold value irrespective of any defined fields.

Additionally, you can set **Count field**; this will identify the number of unique values for a defined field.

In this example, I again used the same query as I did for the custom query previously to identify traffic to `packtpub.com` (`event.category:"network" and event.type:"connection" and destination.domain:"packtpub.com"`). As we only have one system, I opted to group by the `network.protocol` field with over two results and then count by the number of unique values for the `source.port` field. So, this rule will trigger when there are more than two network protocols and more than two source ports.

This rule would create alert rules only when the preceding criteria are met:

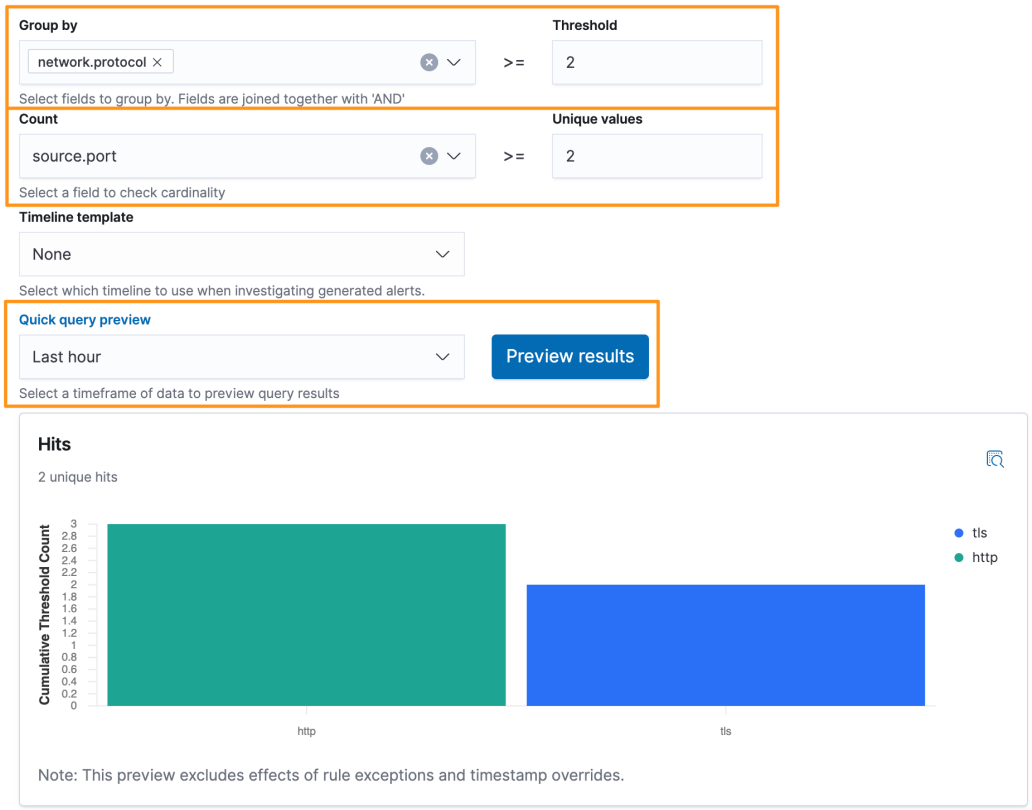

Figure 8.20 – Detection rule – threshold rule

Threshold rules are valuable when you are looking for things such as multiple processes calling the same domain or maybe user logins from more than one IP address.

In this section, we discussed creating a threshold detection rule; in the next section, we'll discuss creating a correlation rule with EQL.

Event correlation rule

As we discussed in *Chapter 7*, *Using Kibana to Explore and Visualize Data*, using the **Event Query Language** (**EQL**), we can create a rule to correlate multiple events together. This is extremely helpful when correlating process and network events together.

In this example, I am using `process.entity_id`, which is a unique identifier for a process, to connect the process and network event together. I am correlating events where the cURL process starts and makes a connection:

```
sequence by process.entity_id
     [process
         where event.type in ("start", "process_started")
          and process.name == "curl.exe"]
     [network
          where event.type == "connection"]
```

Here we can see how this event correlation rule looks in Kibana:

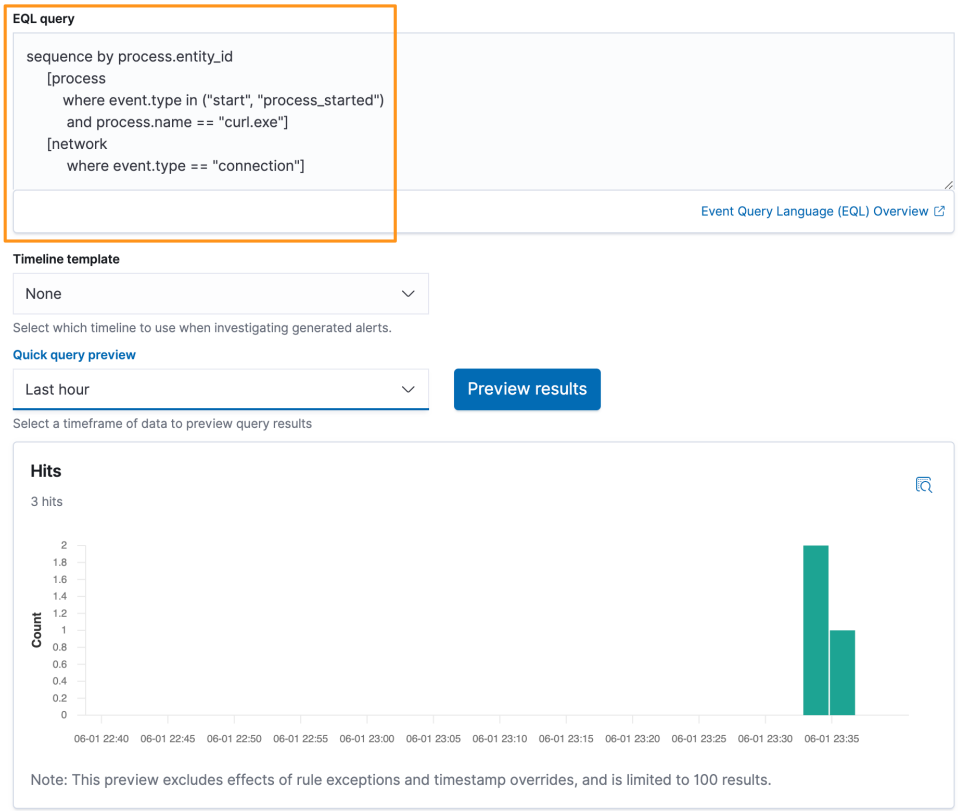

Figure 8.21 – Detection rule – correlation rule

This rule type can be used for any EQL rules or when you need to correlate multiple data types together.

Next, we'll explore the last rule type, the indicator match rule.

Indicator match rule

The indicator match rule is used to match local observations with indicators that are provided either by a previously ingested list (as we discussed using the Data Visualizer in *Chapter 7*, *Using Kibana to Explore and Visualize Data*) or from a threat feed.

In this section, we spent time creating and managing detection rules. Back on the main **Detection** page of the Security app, we'll continue down the page with the trend timeline.

The easiest way to do this will be to use the `Threat Intel Filebeat` module we set up in *Chapter 5*, *Building Your Hunting Lab – Part 2*.

The easiest way to generate some samples will be to identify a malicious domain and browse to it; after all, we're in a sandbox, right?

In the **Discover** app, go to your threat feed index pattern (`filebeat-*`). Let's apply a few filters so we can zero in on a good indicator to test:

```
event.dataset:threatintel.anomali and threatintel.indicator.
type:domain-name
```

This will narrow our view to data provided by Anomali and only domains. This search conforms with the threat ECS fieldset.

Let's add the `threatintel.indicator.domain` field as a column and pick any domain:

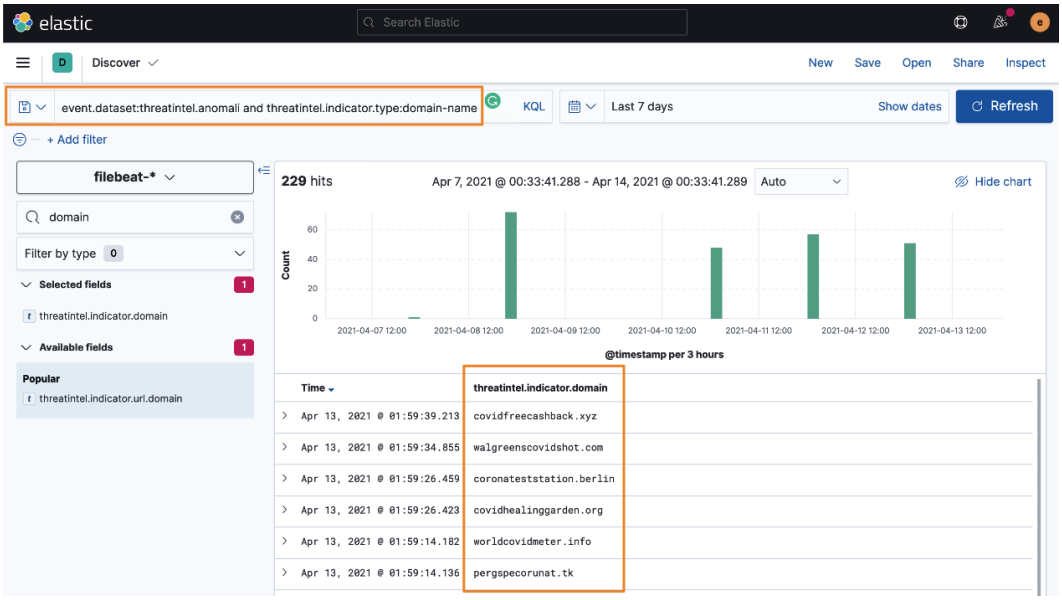

Figure 8.22 – Sample domain indicators

Now that we have the domain, let's go to our Windows box and use cURL to reach out and touch the domain:

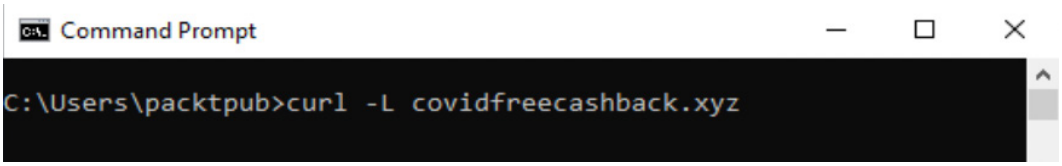

Figure 8.23 – Using cURL to generate indicator match traffic

Now that we've generated some data, let's go back to the **Detections** section of the Security app and create our indicator match rule.

Here, we'll use the same index patterns that we've been using.

We'll write a custom query for our local data. The default is `*:*`, meaning "match everything," but we're looking for domain traffic, so we can write a more specific rule to make it more performant:

```
event.category:"network" and event.type:"connection" and
destination.domain:*
```

Next up is the indicator index pattern; remember, we're using `filebeat-*`.

The indicator query can be tuned to require that the domain indicator exists, again, to make it more performant:

```
threatintel.indicator.type:"domain-name" and threatintel.
indicator.domain:*
```

Finally, we'll define what observation fields match what indicator fields. Here we're saying that the value of `destination.domain` must match `threatintel.indicator.domain`. You can extend this to include multiple fields to match, such as file hashes, IP addresses, and any fields that are present in both your local index patterns and your indicator data:

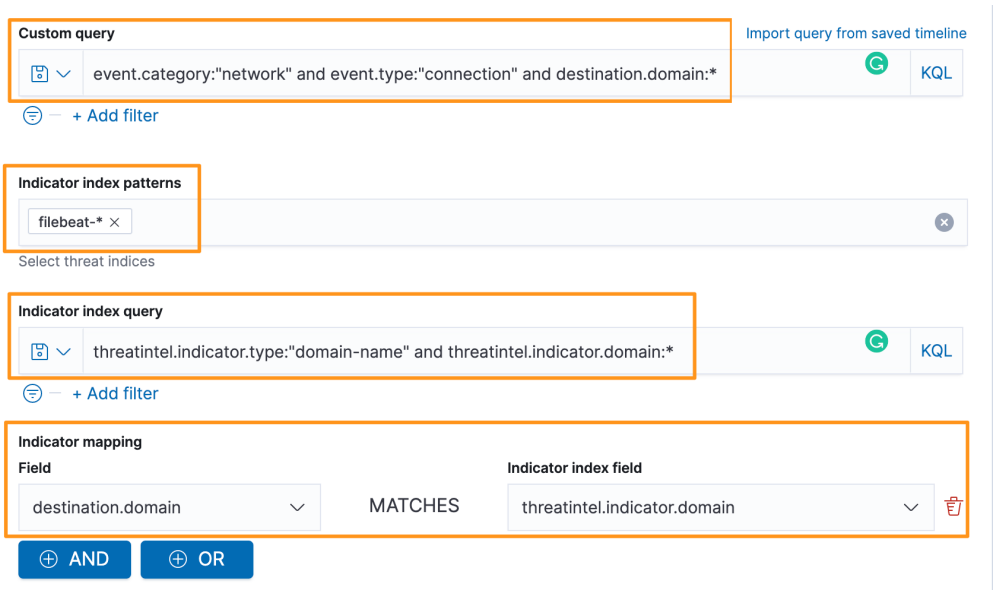

Figure 8.24 – Detection rule – indicator match

Running this rule, we can see that our generated malicious traffic generated a rule alert:

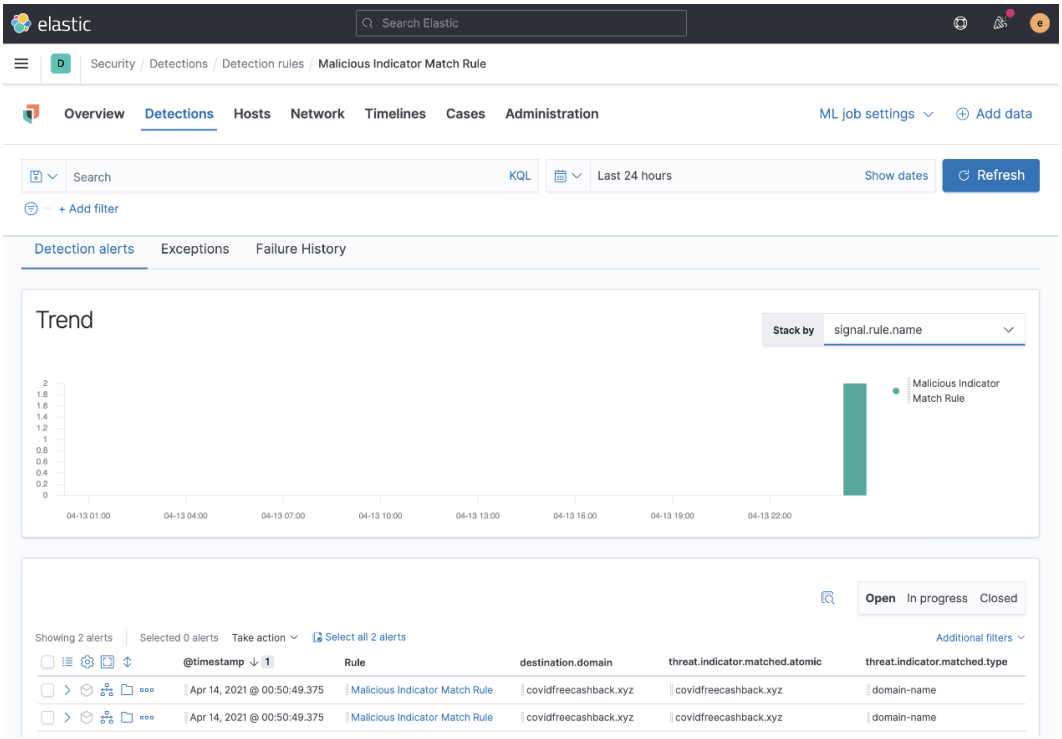

Figure 8.25 – Detection rule – indicator match alert

You may notice that there are fields that end in `.matched`. We're going to explore those in *Chapter 11*, *Enriching Data to Make Intelligence*, when we talk about indicator enrichment.

In this section, we created five detection rule types in the Security app. Next, we'll learn more about the **Detection alerts** page.

Trend timeline

Using the trend timeline, you can sort all events by their specific metadata. This is a view into all of the events and alerts that are generated by the detection engine. This is very helpful in identifying priority, risk, and criticality based on the detection rule settings. We'll discuss that more as we create our own rules.

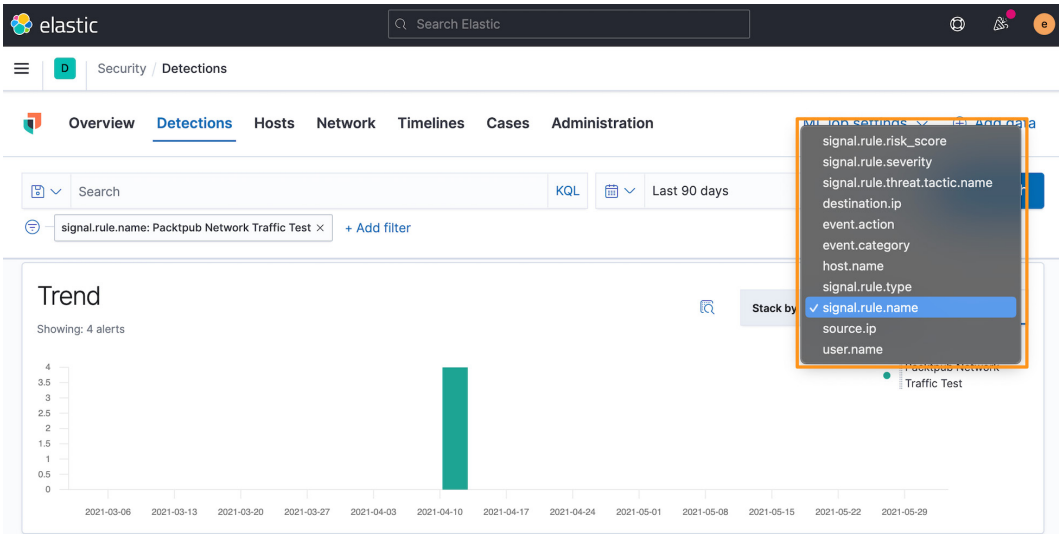

Figure 8.26 – Detection alerts trend sorting

If we continue down the page, we'll see more detailed information for each event that has occurred. While there is a lot in this single visualization, it is laid out in a way that makes it fairly intuitive.

From this visualization, we can customize the columns that are in our view, adjust the renderers, move to full screen, and sort the events:

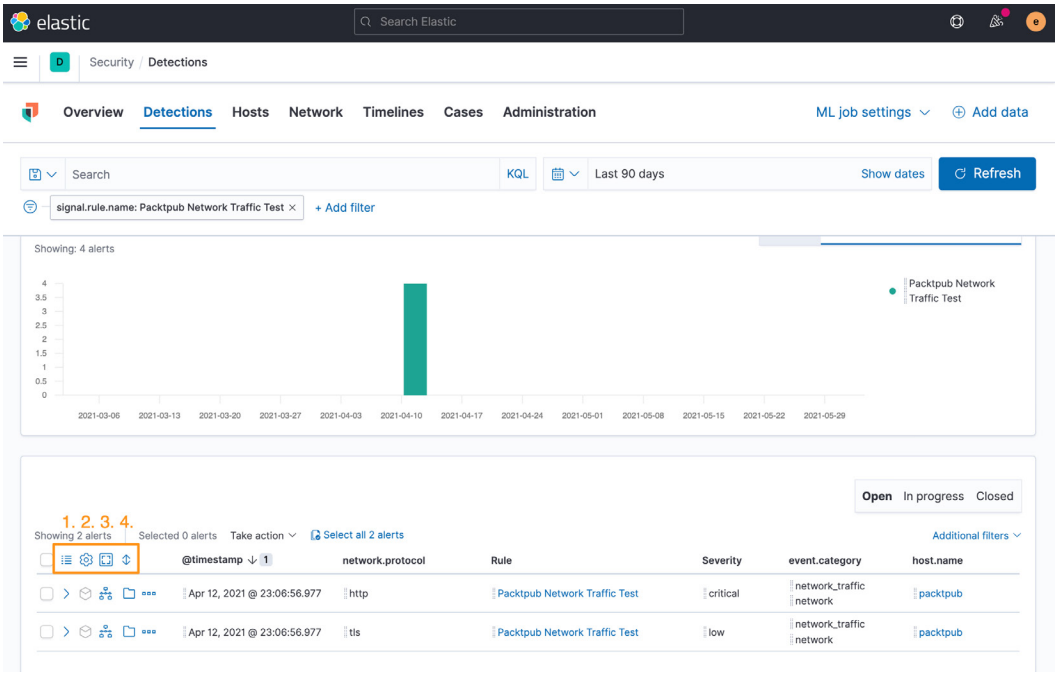

Figure 8.27 – Detection event organization

In the preceding screenshot, you can see four buttons (the icons labeled **1** through **4**):

1. **Customize Columns**
2. **Customize Event Renderers**
3. **Full Screen**
4. **Sort**

Clicking on **Customize Columns** will allow you to select the different information that you'd like to show in the event details. I usually change this to include just the timestamp, rule name, module, category, host name, user name, filename, and destination domain. This allows me to get a quick look at the important information. There is plenty more to look at, but this provides you with the basics.

To change the columns, click on **Customize Columns** and then you can type the field names in the query and simply put a checkmark next to the ones you want to add:

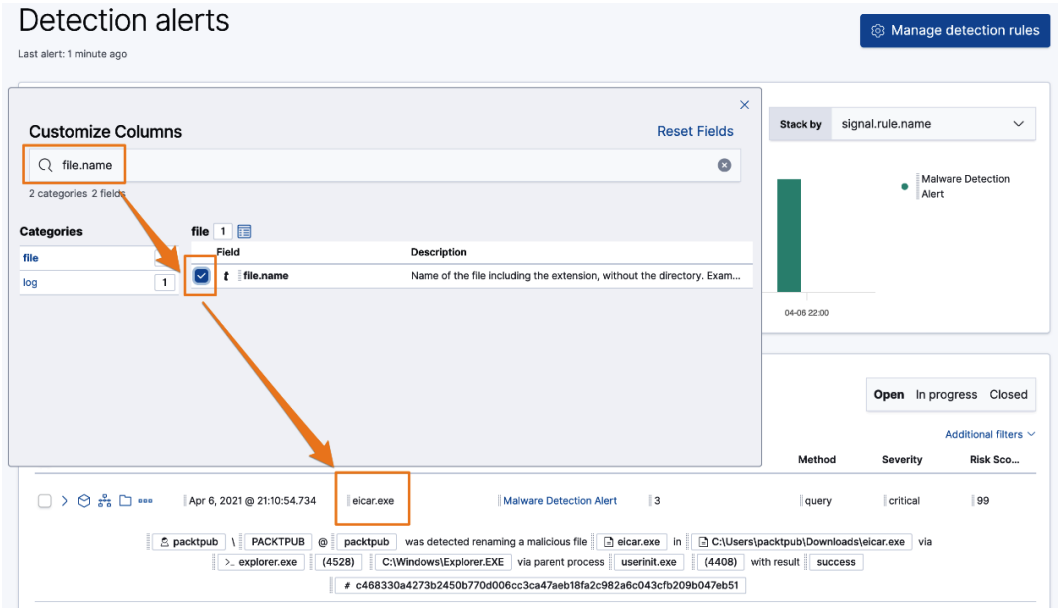

Figure 8.28 – Adding columns to the detection event view

You can also search for any columns you want to remove, but I find it's faster to just click the **x** next to the column name:

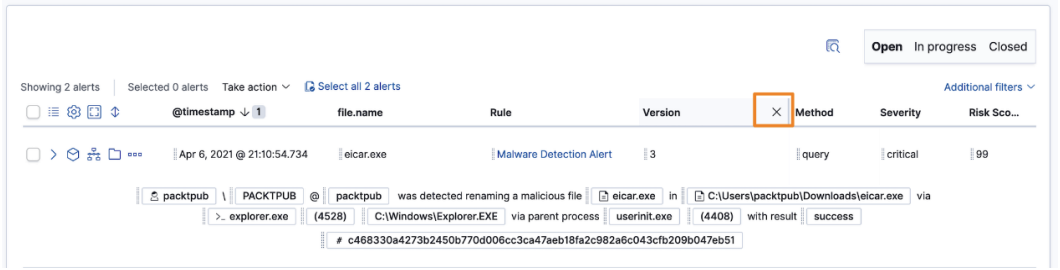

Figure 8.29 – Removing columns from the detection event view

You can also click and drag columns around to reorganize them. This is helpful for your specific analytical process – organizing data in a logical way that makes sense to you.

Next to the **Customize Columns** button, there is the **Customize Event Renders** button. Adjusting the renderers allows you to change your view to a specific context. By default, we're just looking at the basic information about the events. If we click on **Customize Event Renderers**, we can select **Alerts**:

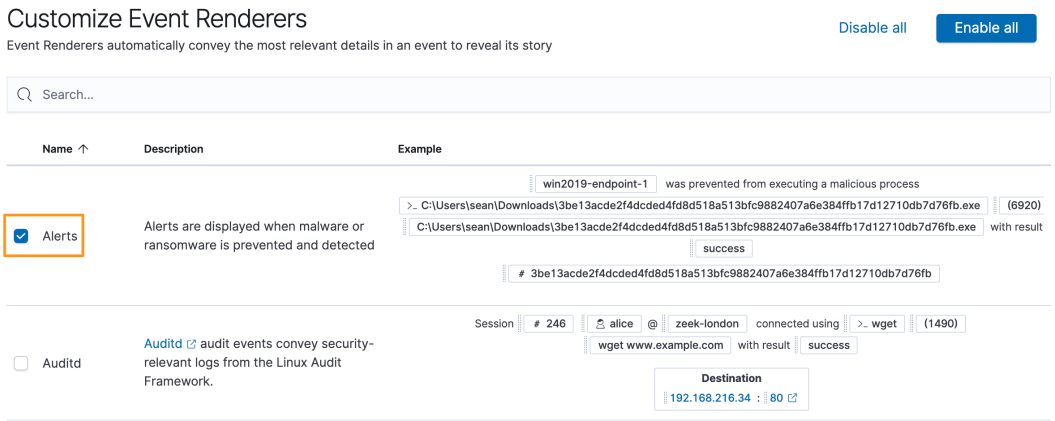

Figure 8.30 – Customize event renderers

Selecting **Alerts** organizes our events into a context of security events instead of a generic view.

You can view the event details in full screen mode by clicking the **Full Screen** button.

Finally, you can sort your events by the default of `@timestamp` or you can select a different field and go either ascending or descending:

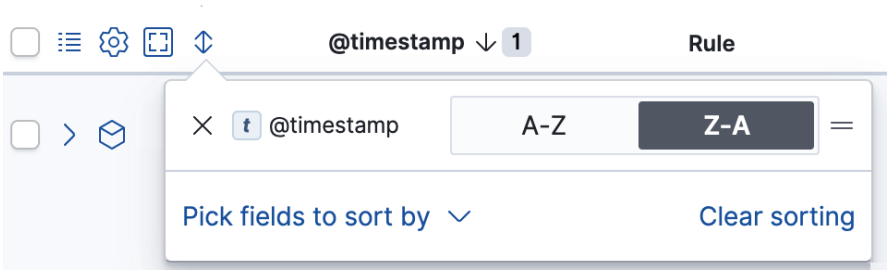

Figure 8.31 – Customize event renderers

Now that we've organized our data into an alert-centric context, let's explore the individual event detail options.

Immediately below the **View customization** section are five specific event detail options:

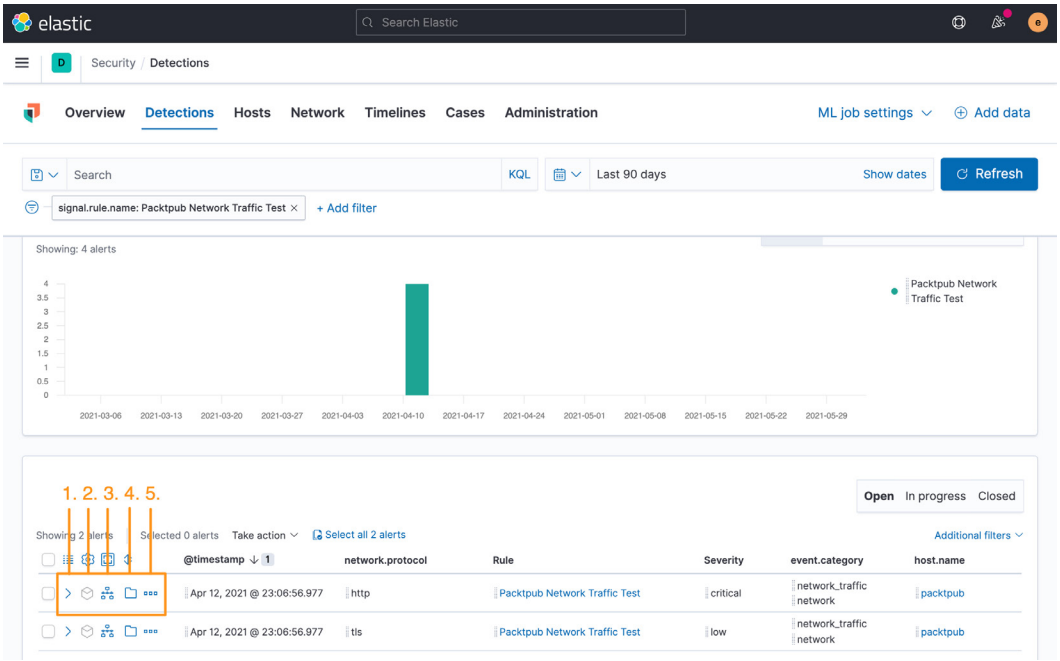

Figure 8.32 – Detection event details

In the preceding screenshot, there are five buttons (the icons labeled **1** through **5**):

1. **Event Details**
2. **Resolver**
3. **Add to Timeline**
4. **Add to Case**
5. **Event Actions**

Next, we'll walk through each of these options.

Event Details

Clicking on the **Event Details** icon will expand a slide-out pane on the right side of the screen with three different views: **Summary**, **Table**, and **JSON View**.

The **Summary** view will show you the high-level basics of the event, such as when it occurred, what rule was triggered, the severity, the risk score, the host name, and the user name. This provides basic information:

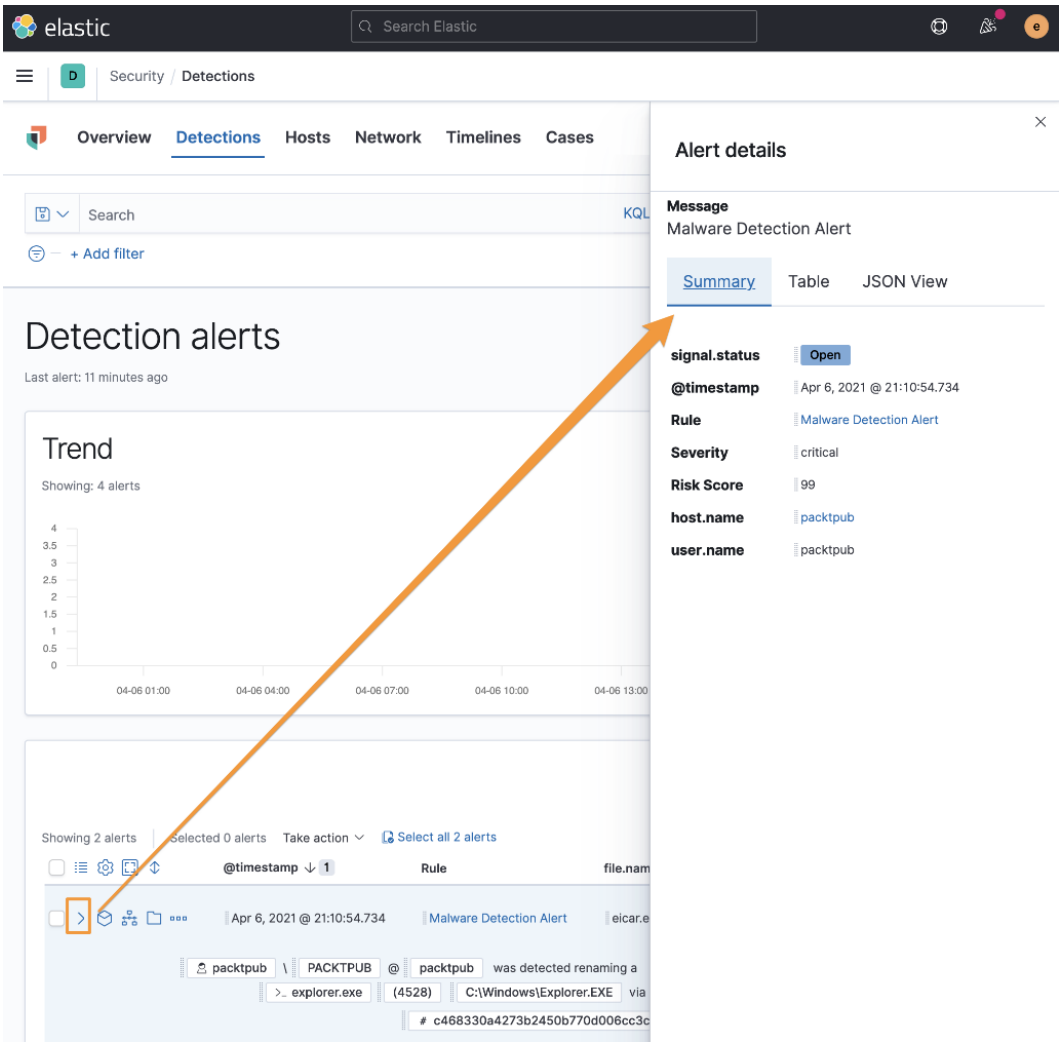

Figure 8.33 – Detection event details – Summary

The **Table** view shows you very granular data organized into a table similar to how an event would look in **Discover**. Like the **Customize Columns** menu we discussed earlier, you can use this view to add and remove columns:

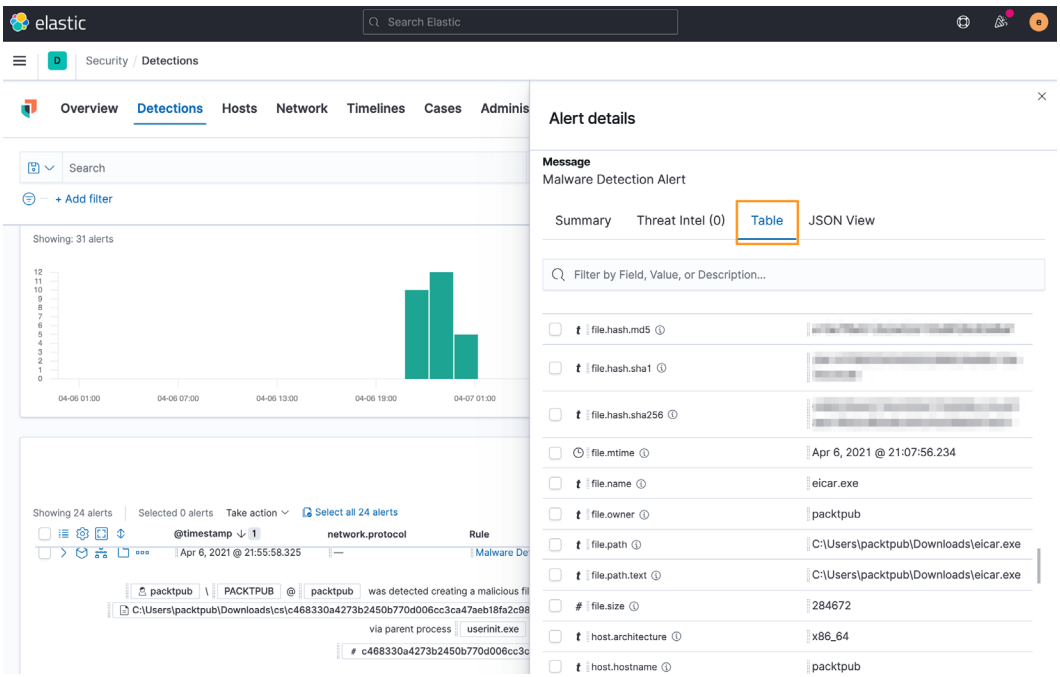

Figure 8.34 – Detection event details – Table

Finally, you can look at the raw data in JSON format. This is helpful if you have specific questions about how the data is structured. While looking at data this way can be busy, it can be helpful to look at similar events. As an example, it can be helpful to look at all the `file` or `process` information in one view:

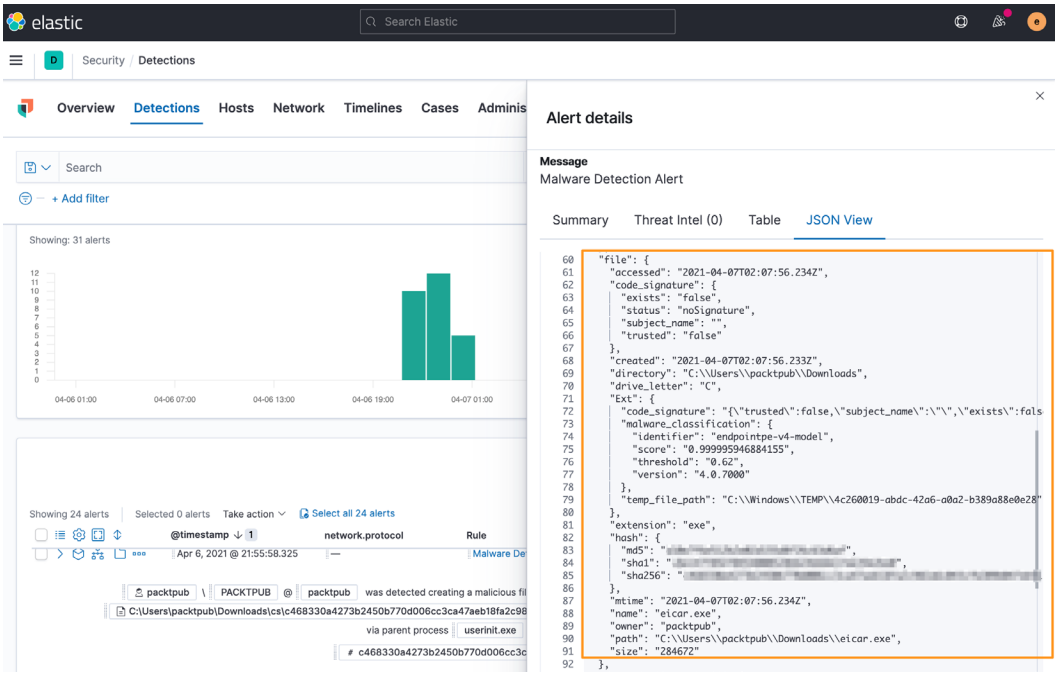

Figure 8.35 – Detection event details – JSON

Viewing the event details is often the first step in responding to events. Being able to move quickly between a summary to details in just a few clicks is very powerful, especially when you don't have to switch screens, portals, contexts, or views.

Next, we'll discuss the **Resolver** view, which is a valuable visualization to track events and their relationships.

Resolver

The **Resolver** view provides a tree-type view of file and network events. What makes **Resolver** particularly powerful is that it connects file and network events. From any security event that was generated by the Elastic Agent, there will be a **Resolver** icon that can be clicked to open the event in the **Resolver** view.

To highlight the utility of **Resolver**, I created a snapshot of my Windows VM, downloaded a malware sample of the Tesla Agent (a popular information stealer and remote access tool), and detonated it on the Windows system. As a reminder, we've configured the Elastic Agent Security Integration to only detect, not prevent.

I am going to obscure the identifying marks of the malware (hashes, network connections, and so on) because this is live malware that could absolutely cause damage. Additionally, adversary-controlled infrastructure may not be owned by them and I don't want to expose innocent victims if they are being used without their knowledge.

Executing the malware, we should take note that the filename is `tesla.exe`. This will help focus our search:

Figure 8.36 – Malware event

Remembering the file we're tracking is called `tesla.exe`, we can look for `tesla.exe`. If we click on the **Resolver** icon and then the **4 file** button, we are shown some additional details:

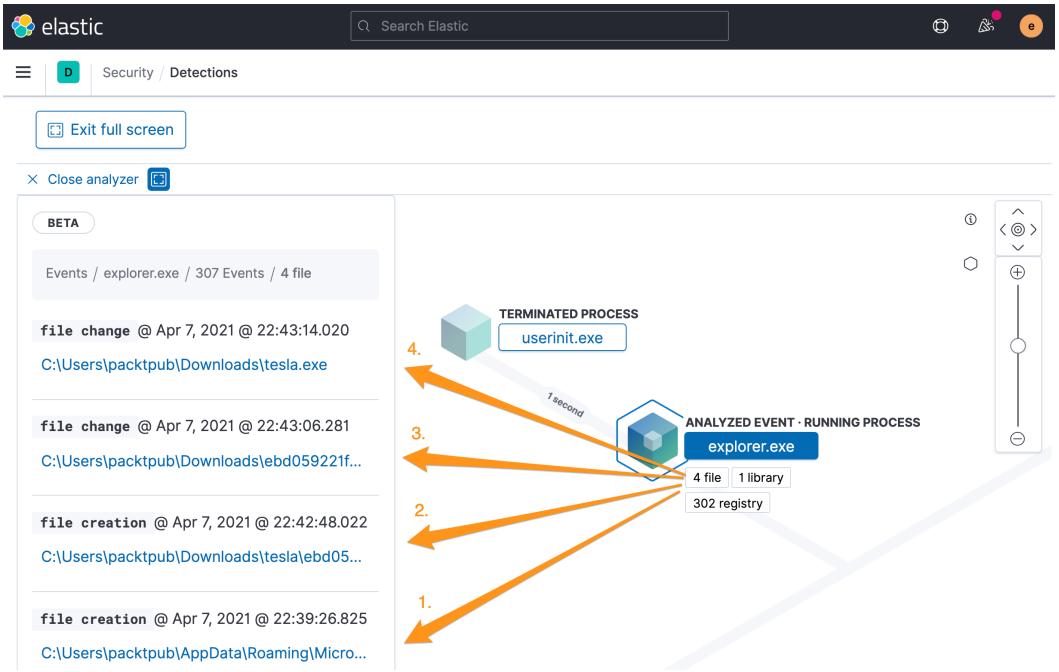

Figure 8.37 – Malware event – Resolver view

Resolver shows us some analysis information and even a few artifacts of me staging the malware! There are labels **1** through **4** in the screenshot:

1. This is a screenshot of the desktop wallpaper that was stored.
2. This was me staging the malware by unzipping it from an encrypted archive. This isn't part of our analysis, but it's great that the agent tracked and associated this with the activity. This would be useful telemetry in tracking an actual event.
3. This was me renaming the malware `tesla.exe` and moving it from the default archive folder. This isn't part of our analysis, but it's great that the agent tracked and associated this with the activity. This would be useful telemetry in tracking an actual event.
4. Here is the actual file that was detonated, `tesla.exe`. We knew that already, but we can see how it's recorded by the Agent.

We'll spend more time deep-diving into `tesla.exe` in *Chapter 9*, *Using Kibana to Explore and Visualize Data*, but for continuity, we'll continue to use this event for our examples throughout this chapter.

Next, we'll explore how to add events to a timeline.

Adding to a timeline

We had a brief introduction to timelines in the EQL section of *Chapter 7*, *Using Kibana to Explore and Visualize Data*. We'll discuss using timelines to natively build EQL queries later in this chapter in the *Timelines* section, but from within the detection engine, we can create a timeline using the document identification value or even drag a specific event field into a timeline.

The easiest way to create a timeline from an event is to click on the **Investigate in timeline** button next to the event, and this will create a new timeline using the event document ID. The document ID field is named `_id` and it is a unique value that is assigned when an event is indexed:

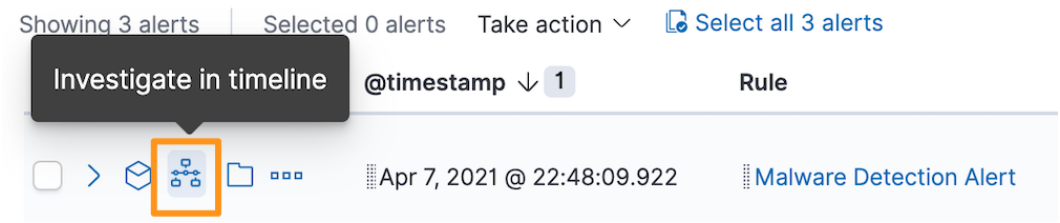

Figure 8.38 – Malware event – add an event to timeline

From here we can see this single event has been added to a timeline:

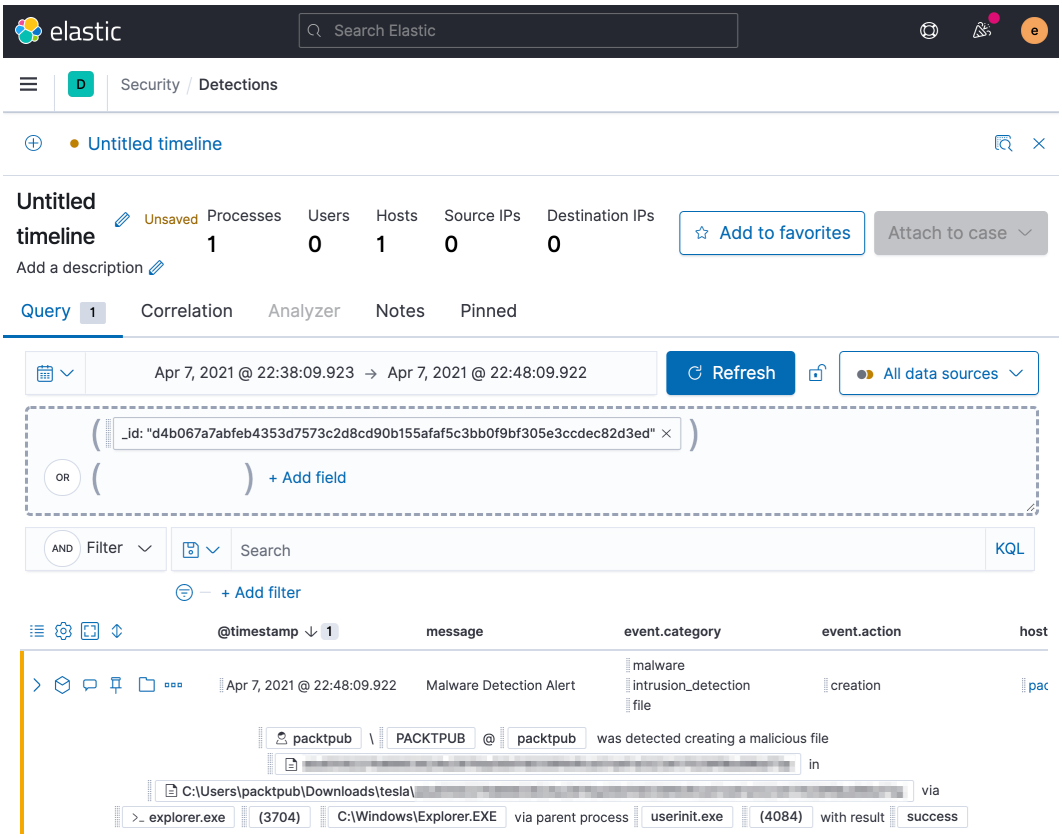

Figure 8.39 – Malware event added to timeline

Additionally, we can also click and drag fields onto the timeline slide-out to add it to a timeline:

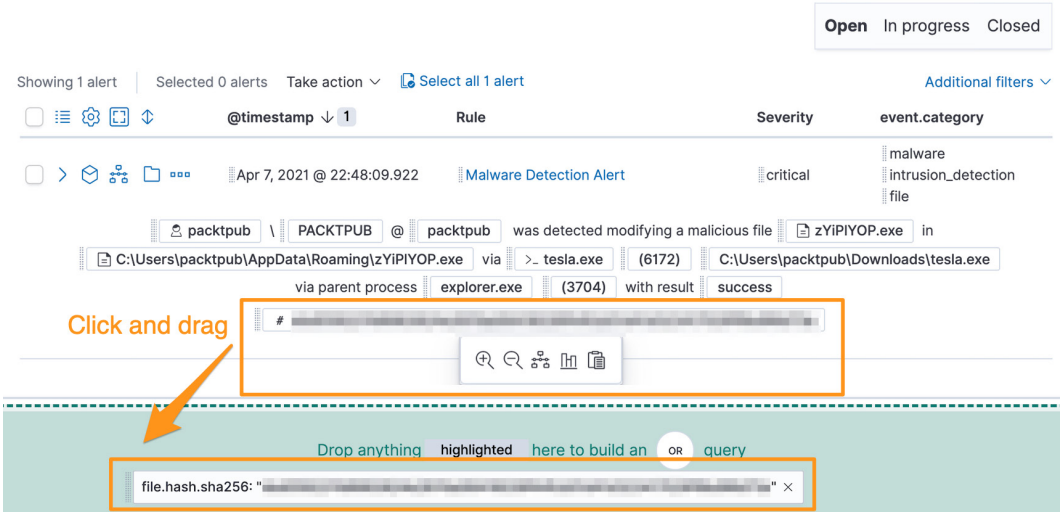

Figure 8.40 – Malware event – click and drag event to create timeline

Naming and providing a description for the timeline is helpful when you're tracking multiple events:

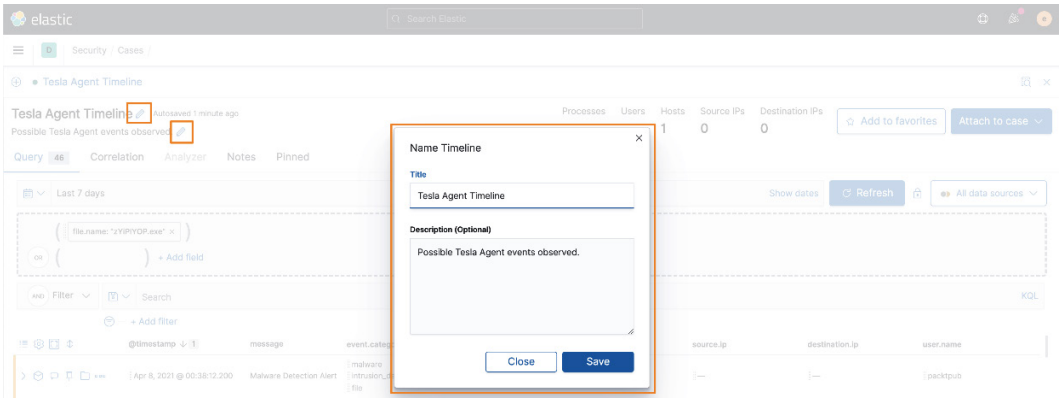

Figure 8.41 – Malware event – naming a timeline

We can also add this directly to an existing or new case (which we'll talk about in the next section):

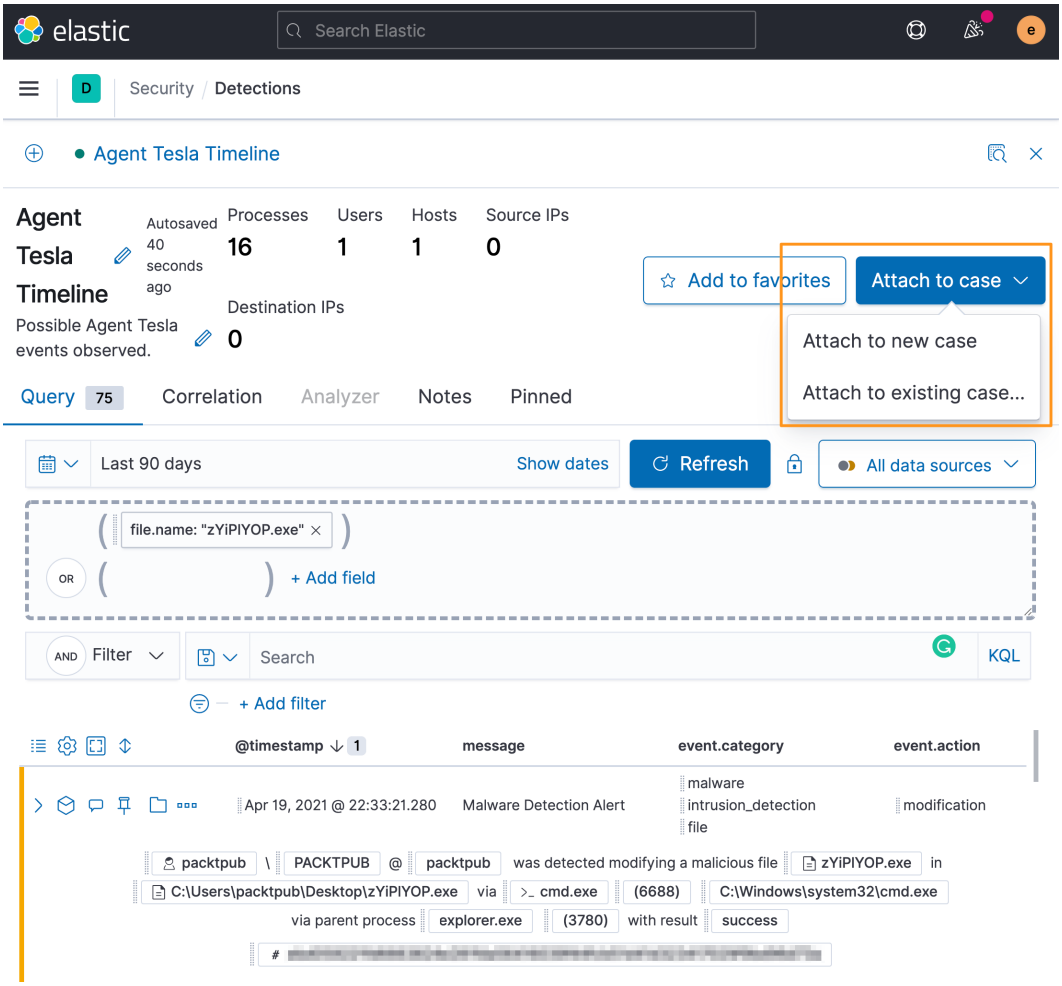

Figure 8.42 – Malware event – adding a timeline to a case

Here we learned the two ways that we can create timelines from the **Detection alerts** page. Next, we'll discuss how to create a case.

Adding to a case

Cases is a developing feature in Kibana. Currently, it is a case management capability to save links to data and make notes.

We'll work more with cases later in this chapter, but from the **Events** page, you can click the **Cases** button and add an event to a new or existing case:

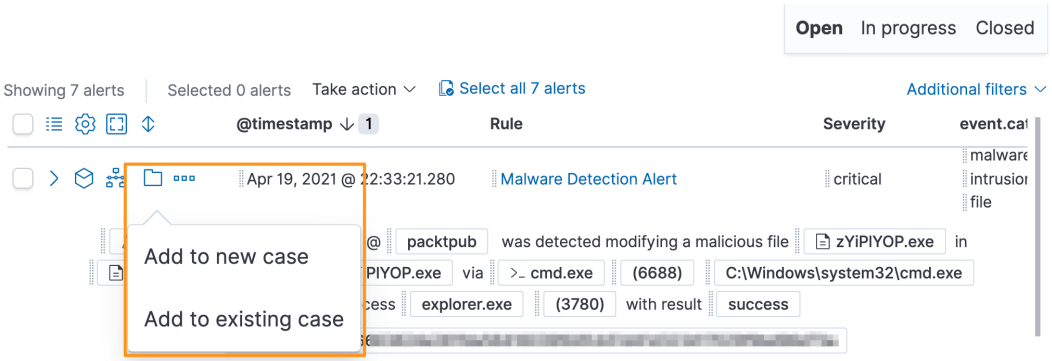

Figure 8.43 – Malware event – adding an event to a case

Here we showed how to add an event to a case. Next, we'll explore how to modify the status of, or even make exceptions to, an event.

Event actions

Now that we've looked at the different things we can do from an analysis perspective with an event, there are some administrative actions that we can take on an event.

We can mark an event as **In progress** or **Closed**. This simply helps with the event response organization so that multiple analysts can work on the alerts without stepping on each other's toes. When you mark an event as **In progress** or **Closed**, it is filtered from the default **Open** view. To find out who is working on an event, it must be in a case:

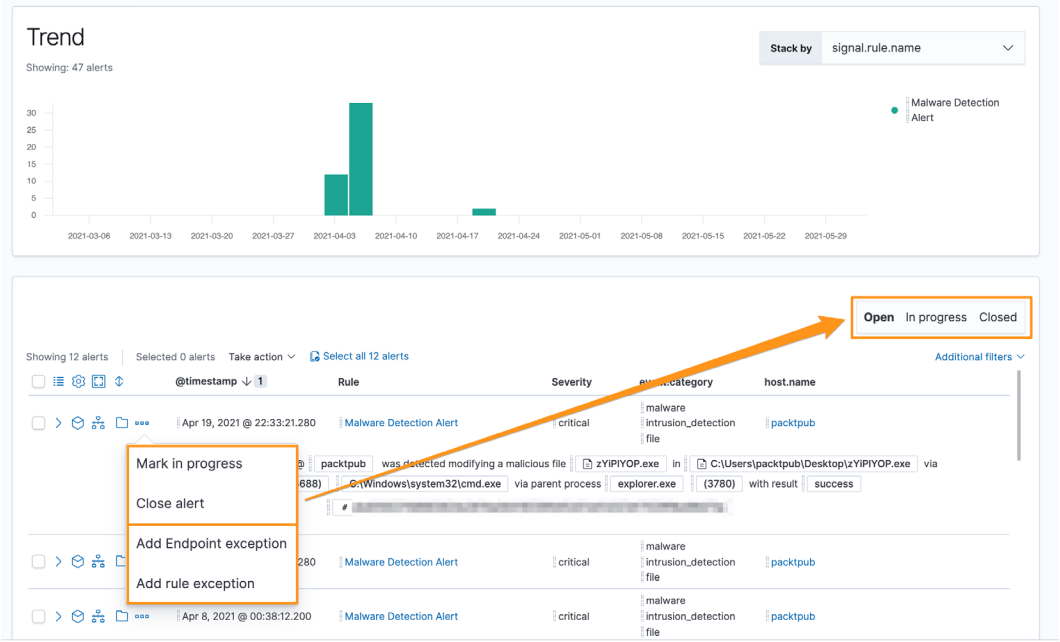

Figure 8.44 – Malware event – changing event StatusMalware event – changing event status

Additionally, we can also create either a rule or an endpoint exception. Both of these exception types prevent alerts from being generated when their conditions are met.

The difference between a rule exception and an endpoint exception is that the endpoint exception is evaluated on the endpoint and the rule exception is evaluated in the detection engine. This difference is extremely important if you are taking preventative measures on the endpoint, where you'd want the exception evaluated on the endpoint or the prevention would still occur.

When creating either exception type, you can choose to close the current alert as well as any other alerts that match the criteria. Endpoint exceptions can remove an endpoint from quarantine if the exception criteria are met.

Clicking on the *three dots* next to the alert allows you to create an exception of either type:

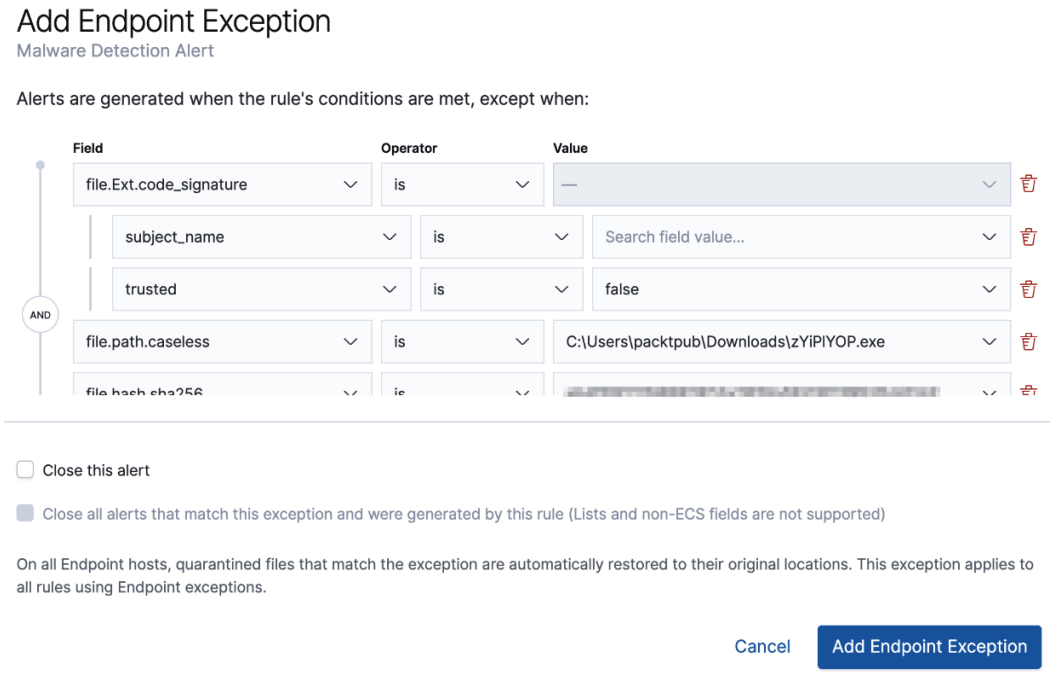

Figure 8.45 – Malware event – Add Endpoint Exception

Here we are creating an endpoint exception using the information that would prevent a malware detection alert from being generated if the preceding criteria are met:

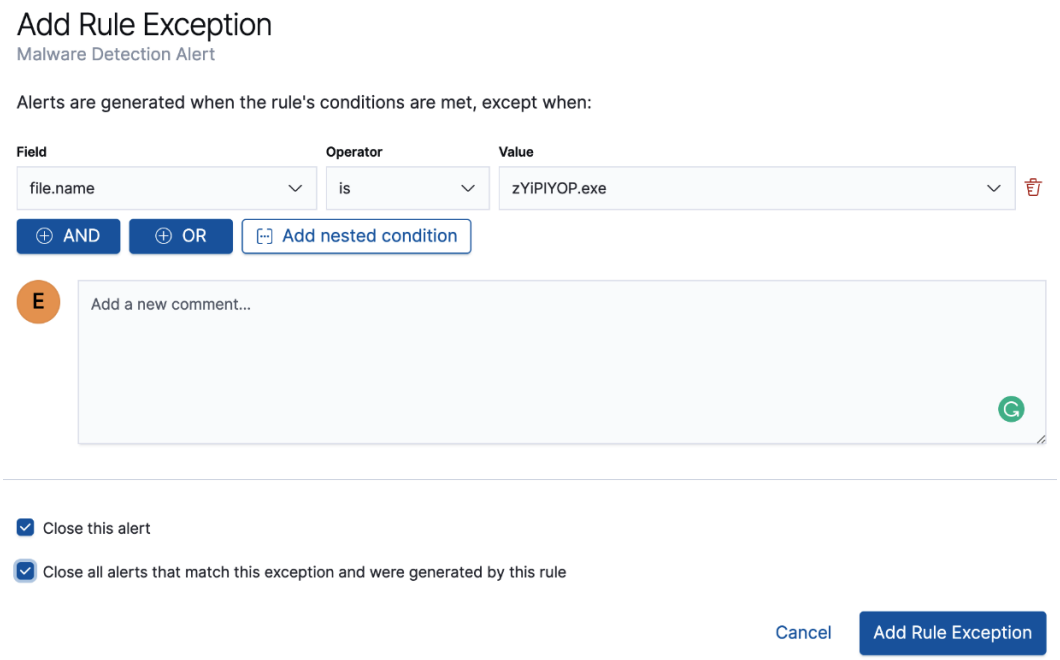

Figure 8.46 – Malware event – Add Rule Exception

We can also make a rule exception to not create an alert when the filename is `zYiPlYOP.exe`.

When making exceptions, we should look back at the *Pyramid of Pain* that we discussed in *Chapter 2*, *Hunting Concepts, Methodologies, and Techniques*. Remember that the higher up in the pyramid we go, the harder it is for an adversary to adapt. Using that as a critical thinking point, creating a rule based on a filename would be an exception that could easily be circumvented, so ensure you're making exceptions that have a high level of specificity, such as a file hash, code-signing information, or process, network, or registry associations.

The exception framework is a powerful tool in adapting the security solution to your environment. Many files or events could be considered malicious in some environments but benign in yours.

The detections engine is a tremendous part of the Elastic Security solution. We discussed alerts, individual events, organizing, the **Resolver** interface, creating timelines and cases, basic alert management, and creating both endpoint and rule exceptions.

Next, we'll be moving onto the **Hosts** tab to focus on host-specific information.

Hosts

The **Hosts** section of the Security solution allows you to get a high-level view of the endpoints that are reporting into your stack. This can be helpful to get ecosystem-wide metrics about your environment, such as the number of hosts, operating systems, authentication statistics, and so on.

Our lab environment will likely be sparsely populated with data because we only have one host (our victim machine). Looking at a larger analysis environment, we can see how this view can provide an overview of your hosts:

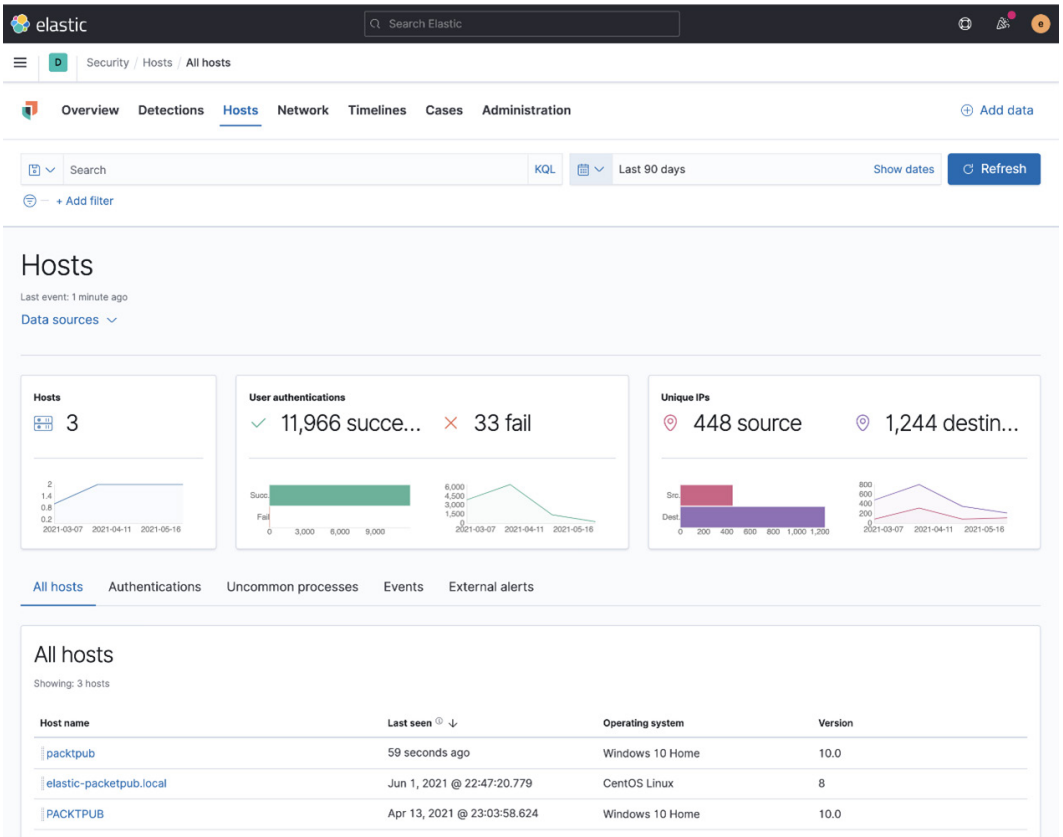

Figure 8.47 – Hosts overview

When we built our lab in *Chapter 4*, *Building Your Hunting Lab – Part 1*, we configured our victim to use the Elastic Agent, Packetbeat, and Winlogbeat. We can see those data sources reflected in the **Hosts** section. If you want to remove specific data sources, you can do that in **DATA SOURCES SELECTION**:

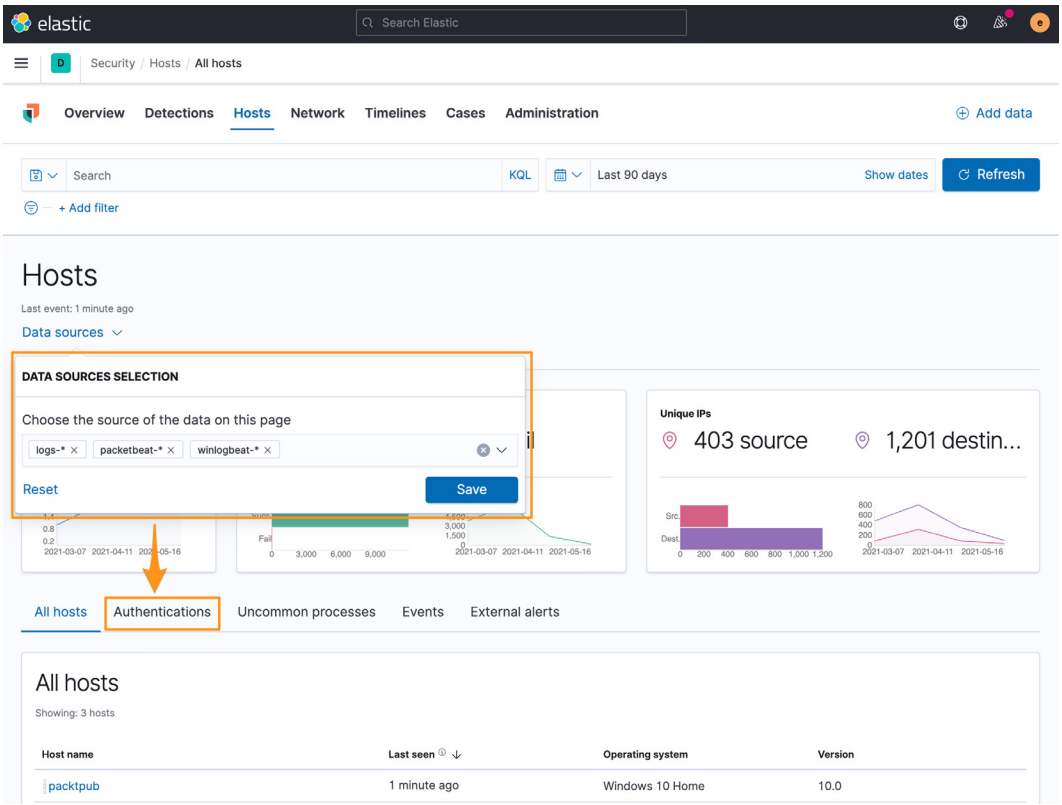

Figure 8.48 – DATA SOURCES SELECTION

Now that we've reviewed the different data source options, we can click on the **Authentications** tab to view an overview of the authentication events:

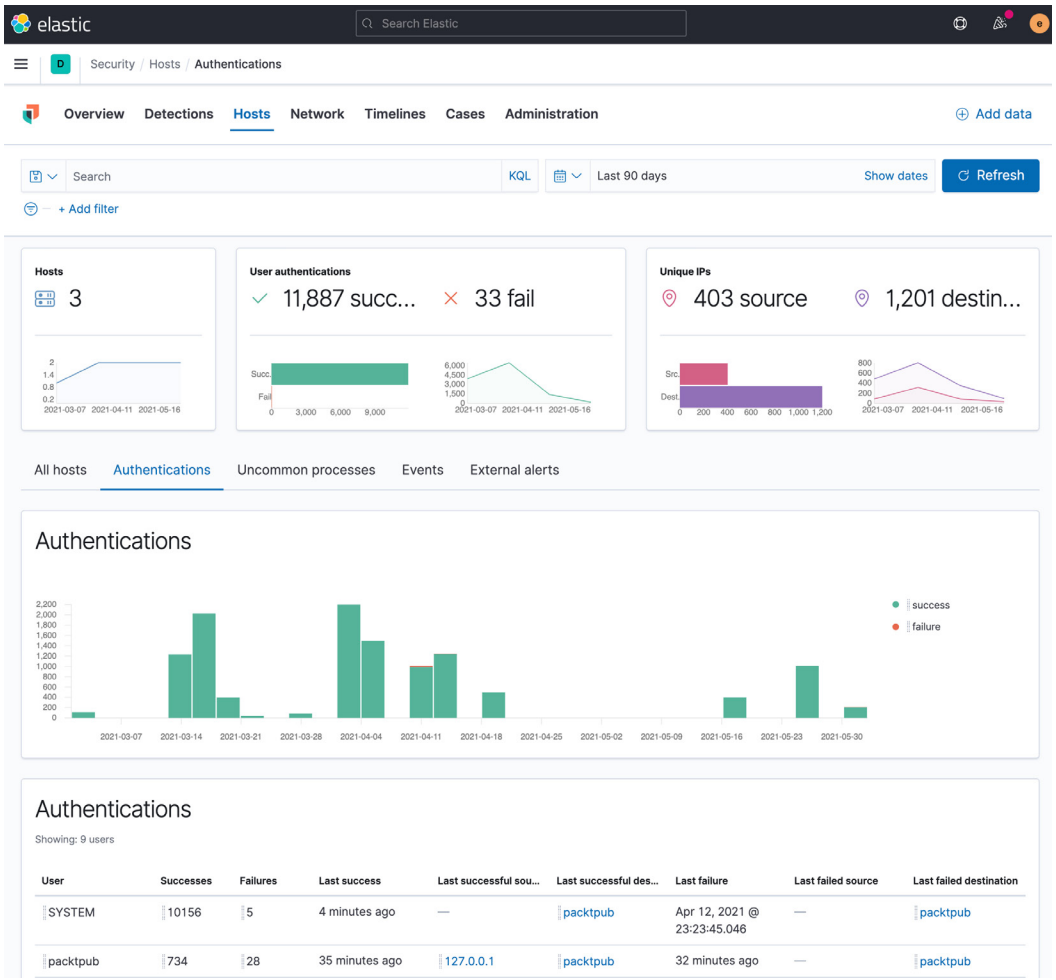

Figure 8.49 – Authentications

We can see additional information about authentications from our environment.

Just like on the **Detections** tab, if we want to know more about the failures that I generated, I can simply click and drag the failures down to the **Timeline** slide-out at the bottom of the screen, or click the **Timeline** button, to investigate more about these events:

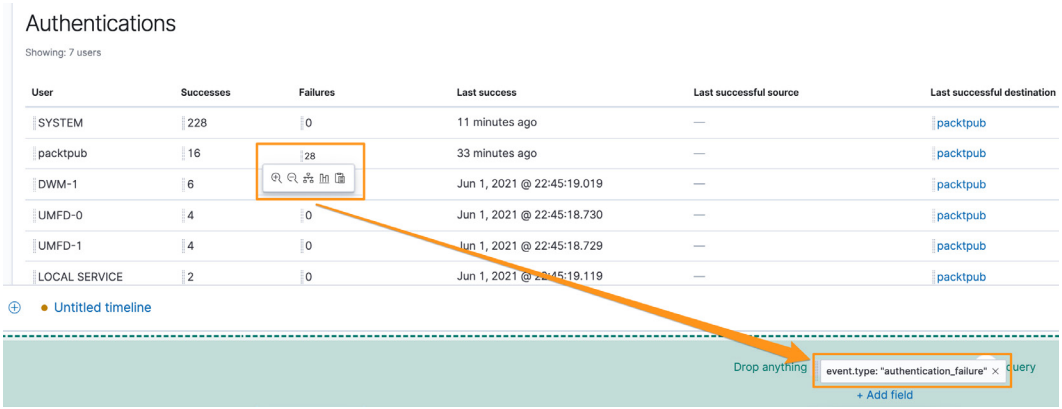

Figure 8.50 – Drag event to timeline

Moving on from **Authentications**, we can click on the **Uncommon processes** tab. This will show us processes that are occurring the least amount of times on the least amount of hosts. Our lab has just one host, so this will have a lot of processes.

Next, we can click on the **Events** tab. This will have a tremendous amount of data, from endpoint events to network events. This can be very valuable, but it should be a place that we search when we have an idea of what we are looking for. As an example, if we search for a previously identified suspicious process, we can do that here and greatly narrow down our aperture:

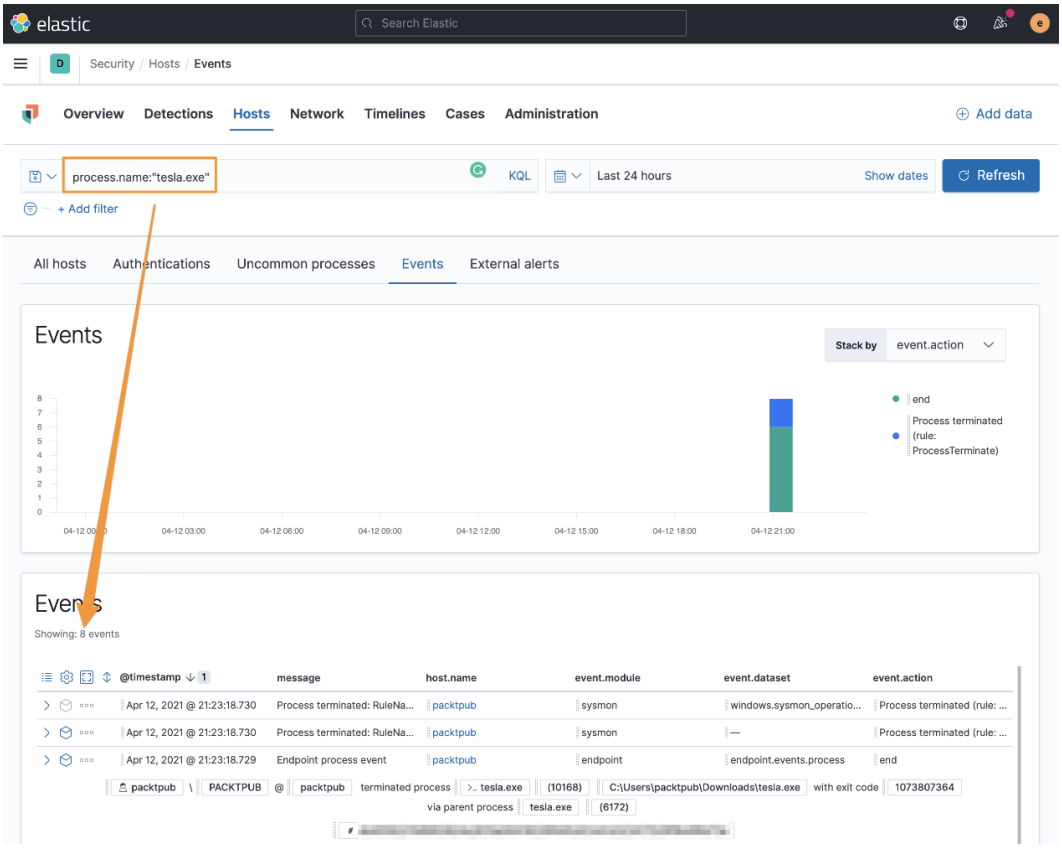

Figure 8.51 – Searching for tesla.exe

The **External alerts** tab will include endpoint alerts that are generated by third parties that are sending data into the Elastic Stack using ECS. Examples could be `osquery` (`https://osquery.io`), Tanium (`https://www.tanium.com`), and others. For our lab environment, we don't have any third-party sources.

Narrowing down the events that are displayed makes this a helpful view; again, as with all things in the Security solution, we can drag events into the timeline or analyze them in **Resolver**.

The **Hosts** section allows you to focus on just host-specific data. While you can use this hosts data to identify network events, having a narrow view while analyzing large amounts of data is helpful to identify abnormal events.

Next, we'll discuss analyzing network-specific events on the **Network** tab of the Security solution.

Network

Clicking on the **Network** tab will take you to an overview of the network data that is provided by the endpoints sending data into our Elastic Stack.

Similar to the **Hosts** section, there are protocol sections to allow you to review the more common network protocols, such as DNS, HTTP, and TLS. **Flows** are display data that doesn't fall into a parsed protocol, but is still recorded by Packetbeat.

Also, like the **Hosts** tab, you'll notice an **External alerts** section. This is where third-party network security solutions would report observations, such as Zeek or Suricata:

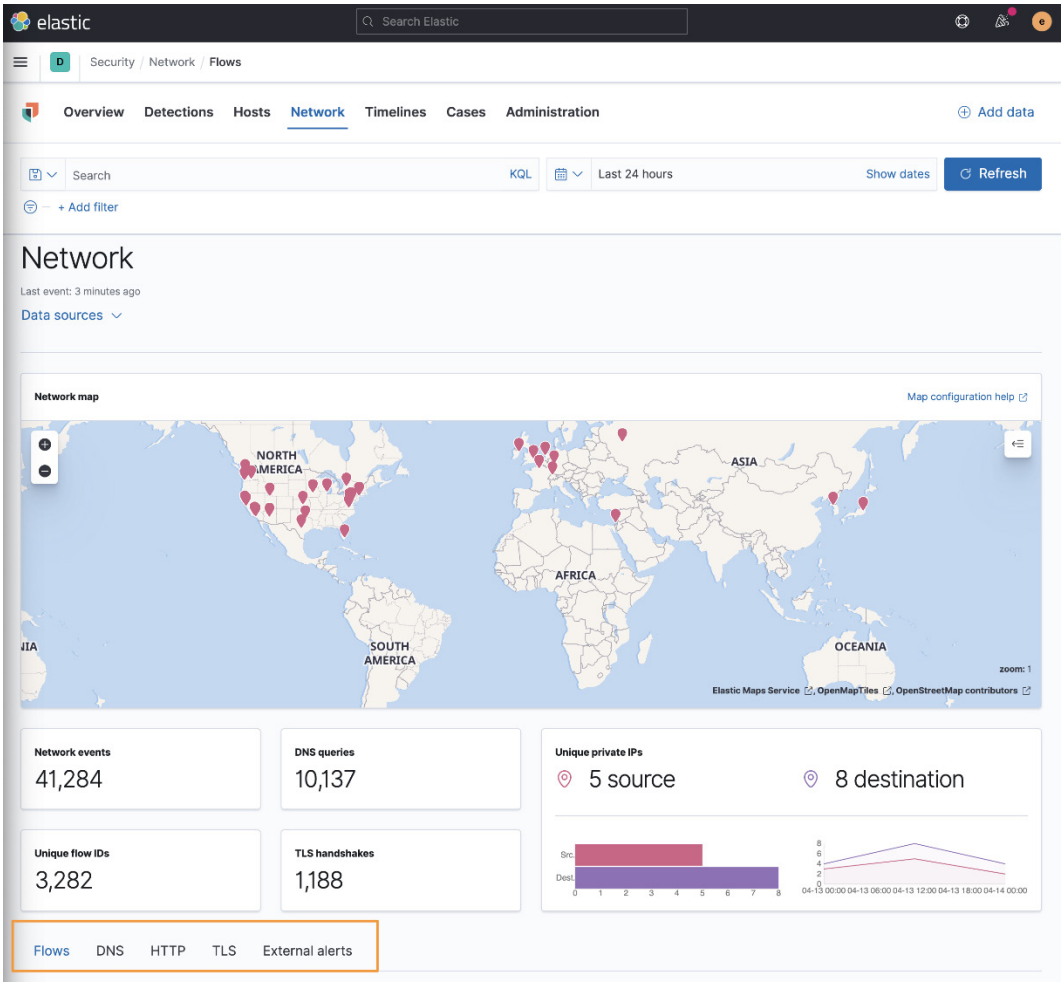

Figure 8.52 – Network overview of the Security solution

In this section, we introduced you to the **Network** section of the Security solution. Next, we'll explore **Timelines**, which is a powerful searching feature from within the Security solution.

Timelines

In the **Detection alerts** section earlier in the chapter, we discussed how to add events to the **Timelines** section as a query, either from the **Alerts** window or from the **Timelines** section by dragging fields onto the query panel.

There is another section in Timelines, and that is where you can write EQL queries. This is a huge benefit because the only other places that you can use the powerful EQL queries are against the Elasticsearch API or correlation detection rules.

Creating a very simple query to correlate events from the endpoint that show the cURL process starting a malicious destination domain we used in the indicator match rule:

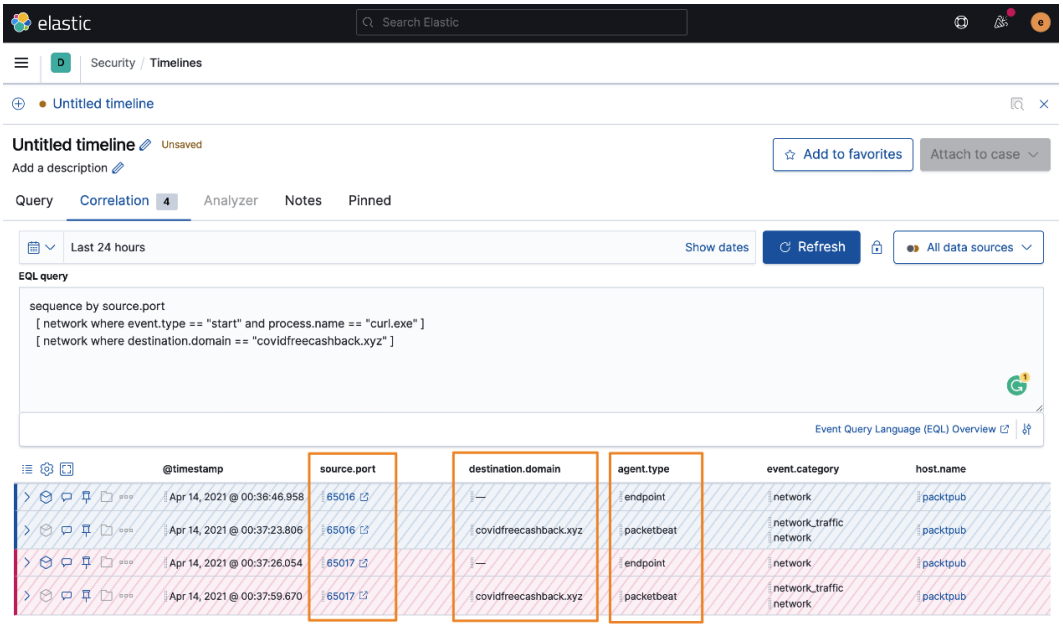

Figure 8.53 – Correlating endpoint and Packetbeat data together

The events are color-coded to visually associate them together. The blue endpoint events go with the blue Packetbeat data, and the same goes for the red events. You can see that the `sequence by` syntax for the `source.port` is reflected in source ports of `65016` and `65017`.

In this section, we covered timelines, which are a powerful tool used to query our security data using EQL for advanced queries.

Next, we'll discuss the **Cases** tool, which is used for basic event tracking.

Cases

The Elastic cases feature is used to manage basic workflow and processes for observed events. This is not a full-blown case management solution; it is basic, with the intention that third-party connections are used for a proper case-management solution.

Cases can be created from the **Alerts** section by clicking on the folder icon, from a timeline, or from the **Cases** tab:

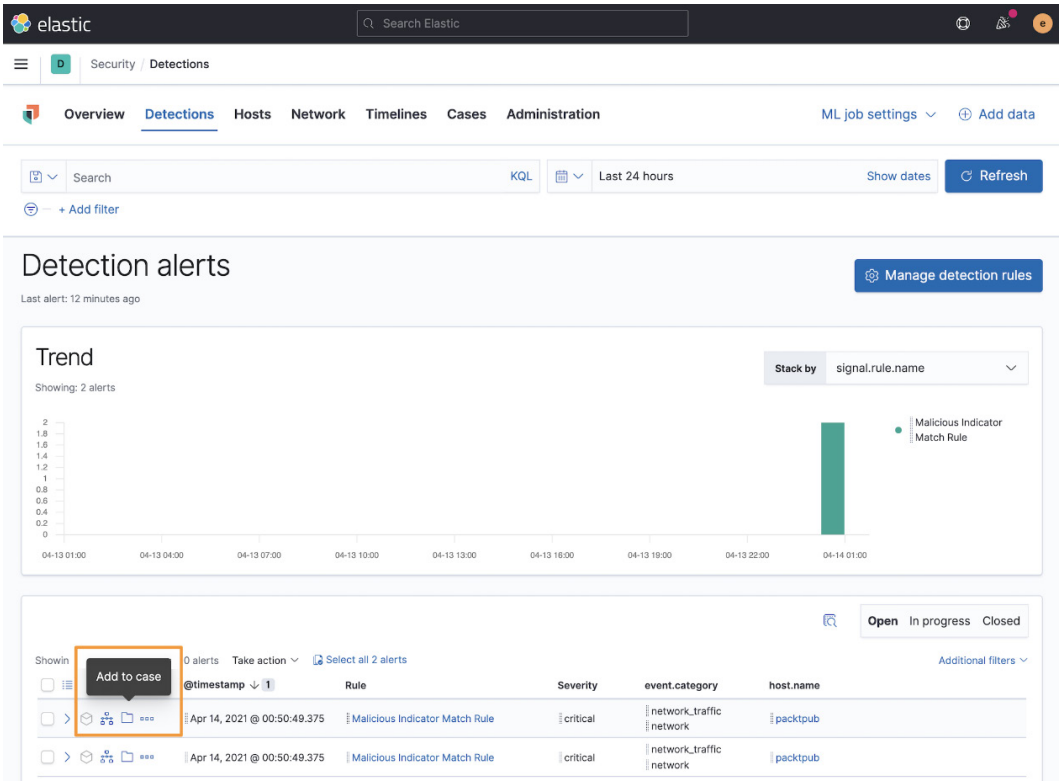

Figure 8.54 – Create cases from the Alerts page

Cases can also have templates added to them that aid in the investigation of events:

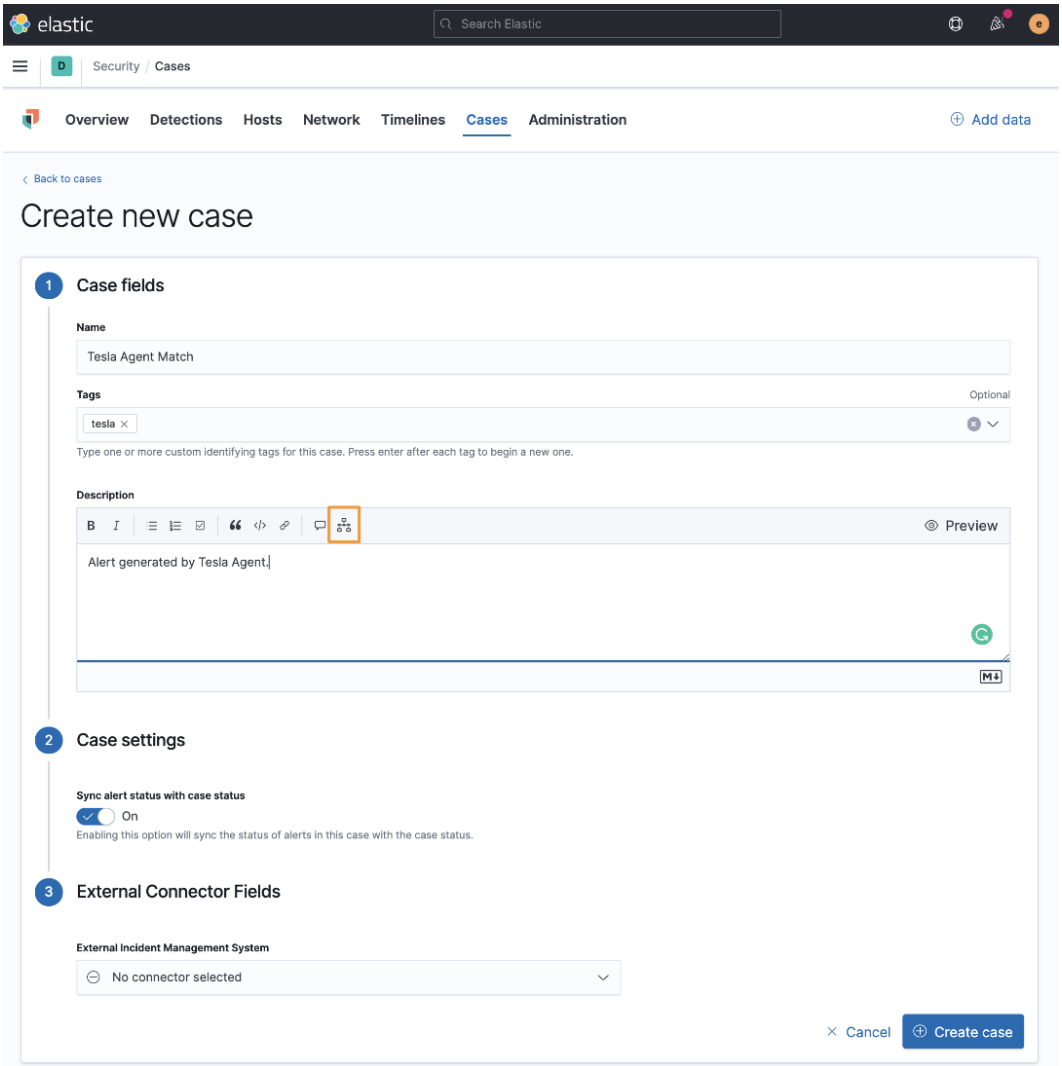

Figure 8.55 – Cases with timeline icon

Clicking on the timeline icon will open a window that will allow you to select any available timeline:

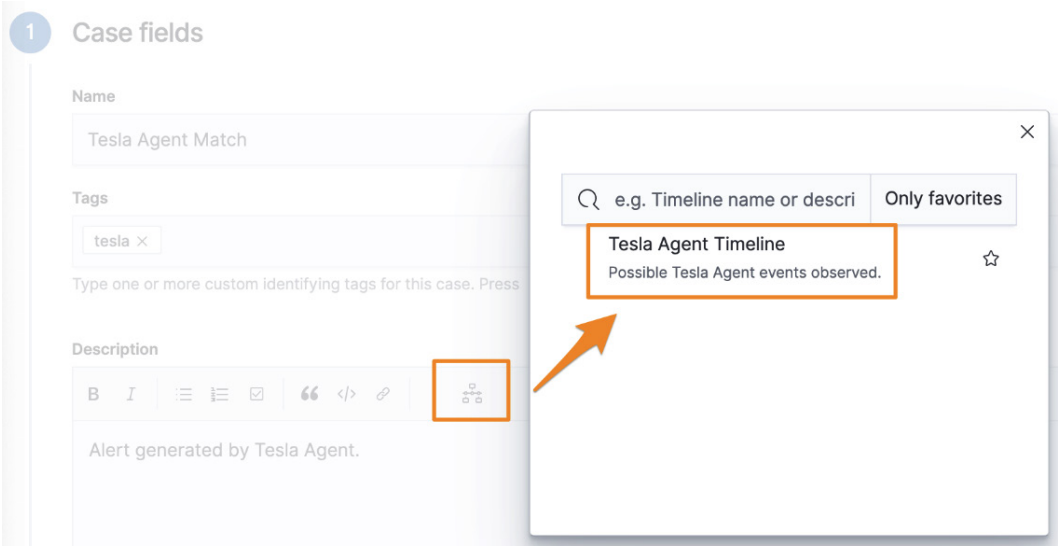

Figure 8.56 – Adding a timeline to a case

We can add the timeline we created for the previously observed Agent Tesla infection. This adds the timeline as a Markdown hyperlink.

Once the case is created, we can make basic annotations and notes during our investigation. All of the comments render Markdown. Once you've completed your investigation, you can close the case from here:

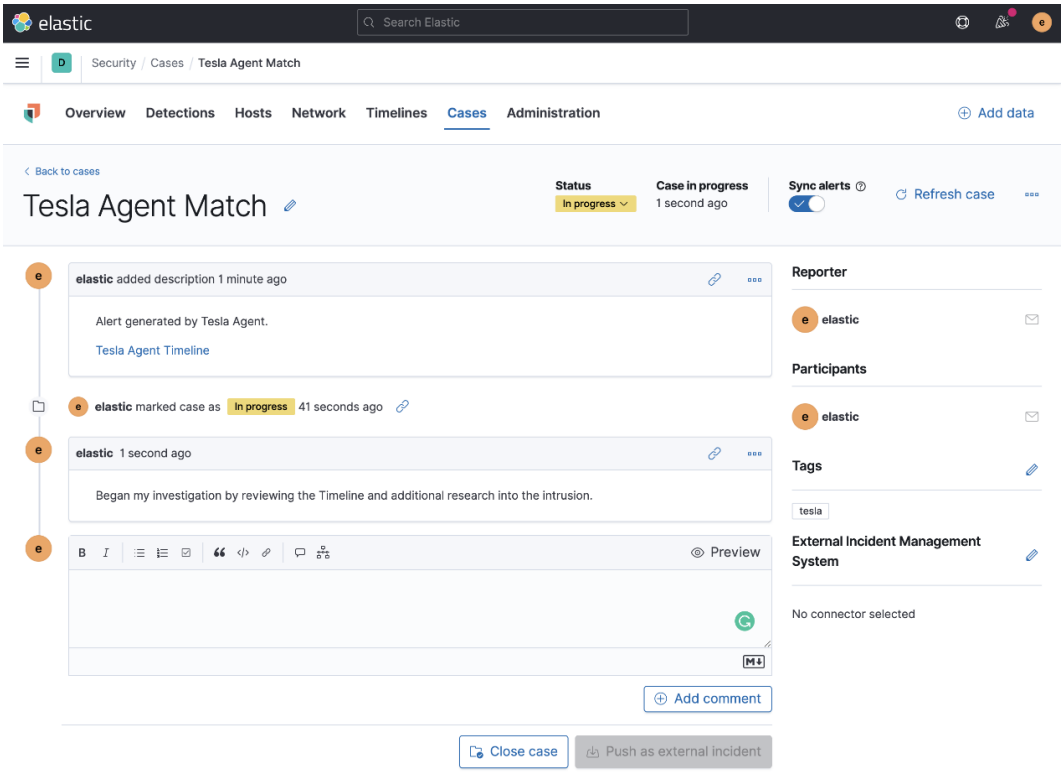

Figure 8.57 – Responding to an event using a case

Using cases, we can manage basic responses to identified events from within the Security solution.

Next, we'll review the **Administration** section of the Security solution.

Administration

The **Administration** tab allows us to review the status of all of the endpoints that are reporting to our Elastic Stack.

Additionally, we can add trusted applications that we don't want to generate alerts from. Great examples could be legacy anti-virus, asset management tools, or vulnerability scanners:

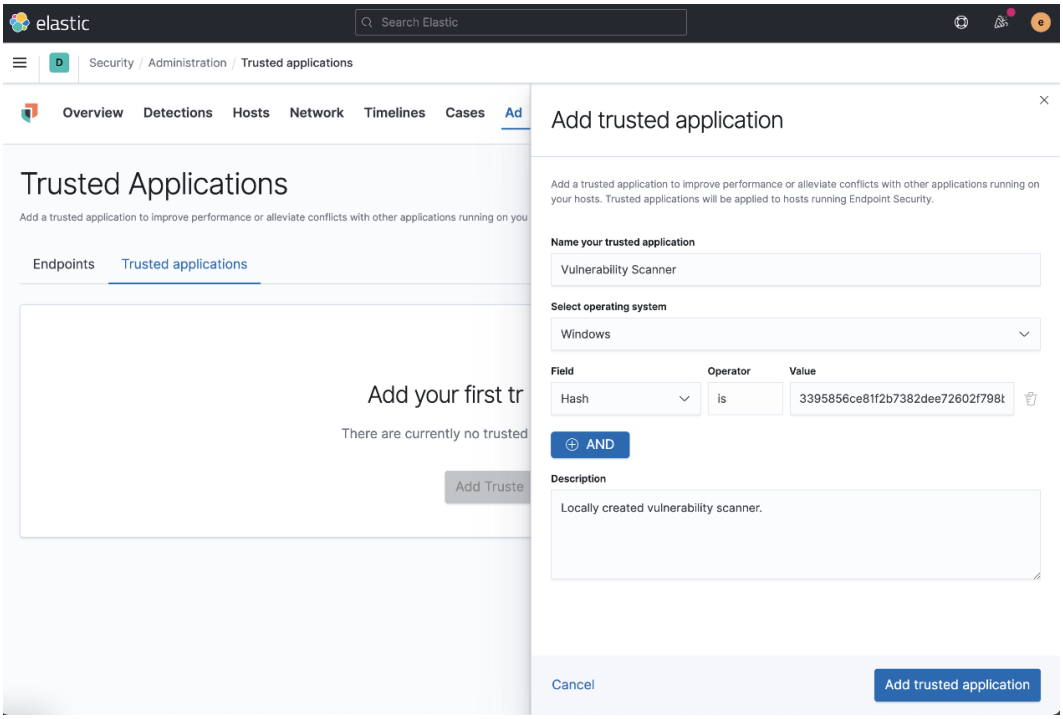

Figure 8.58 – Adding trusted applications

We can name the application, select the appropriate operating system, and define a path, filename, or hash value. We can select multiple fields, so if there is a file that is trusted but could also be abused, we could define the name, hash, and location:

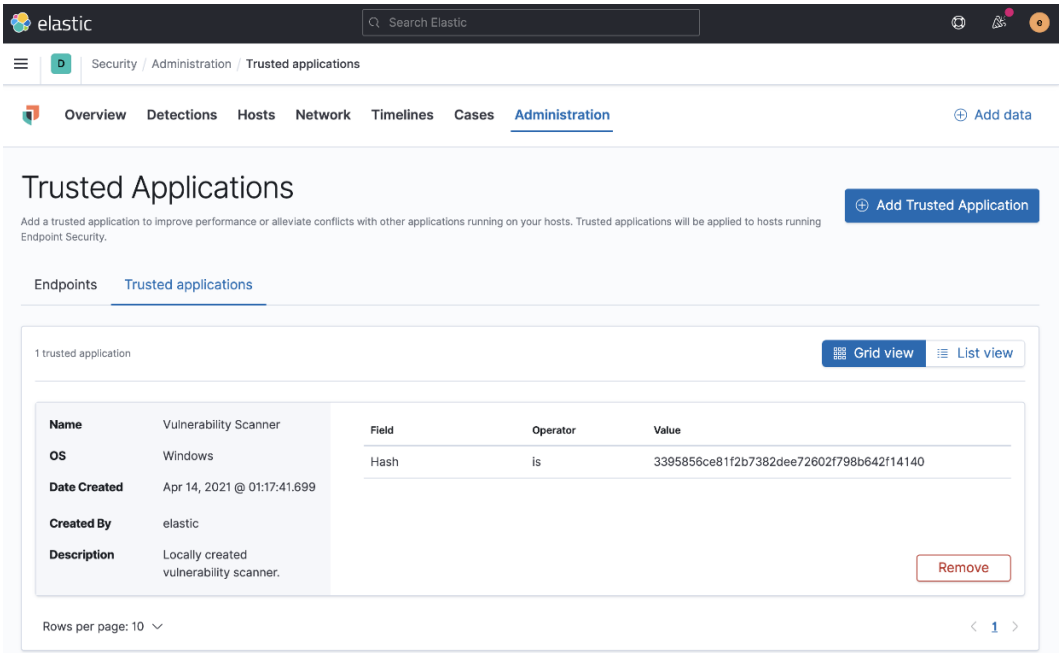

Figure 8.59 – Adding trusted applications

We can monitor all of our trusted applications and see some information about them.

In this section, we discussed the administration of the Elastic Security solution, specifically the endpoints and the trusted applications.

Summary

In this chapter, we thoroughly explored the Elastic Security app. We dug into each of the app sections and explored the detection engine. From the detection engine, we created five different types of rules and generated sample data for analysis. We also explored specific host and network sections that display security-related information. We created timelines for events using EQL. We used cases to track events in combination with timelines. Finally, we explored the administration of the Security solution, looking at adding trusted applications.

The skills you gained in this chapter will allow you to identify malicious events, correlate endpoint and network data together, and begin the analysis process.

In the next chapter, we'll spend even more time in the Security solution, specifically leveraging timelines to further investigate the Tesla Agent event we observed in this chapter.

Questions

As we conclude, here is a list of questions for you to test your knowledge regarding this chapter's material. You will find the answers in the *Assessments* section of the *Appendix*:

1. External host alerts can be collected from where?

   a. Osquery

   b. Zeek

   c. Suricata

   d. Filebeat

2. External network alerts can be collected from where?

   a. Osquery

   b. Zeek

   c. Tanium

   d. Filebeat

3. Indicator match rules can be fed from what module?

   a. Filebeat System Module

   b. Packetbeat

   c. Auditbeat

   d. Filebeat Threat Intel Module

4. Which of the following query languages can timelines use for correlations?

   a. KQL

   b. SQL

   c. EQL

   d. Lucene

5. What is the name of the tool that allows you to visually explore alerts?

   a. Resolver

   b. Hosts

   c. Network

   d. Timelines

Further reading

To learn more about the topics in this chapter, see the following:

- Elastic detection rules: `https://github.com/elastic/detection-rules`
- Building block rules: `https://www.elastic.co/guide/en/security/7.12/building-block-rule.html`
- Tesla Agent: `https://malpedia.caad.fkie.fraunhofer.de/details/win.agent_tesla`
- `Schtasks.exe`: `https://docs.microsoft.com/en-us/windows/win32/taskschd/schtasks`
- attrib: `https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/attrib`