

# SecOps Automation and Response— Cortex XSOAR

CONCEPTS GUIDE

AUGUST 2021



# Table of Contents

---

Preface .....	1
Related Guides.....	3
Other Resources .....	3
Purpose of This Guide.....	4
Audience .....	4
Introduction .....	5
The Challenges .....	5
How to Respond .....	5
Cortex XSOAR for SecOps .....	6
Data Management Concepts.....	10
JavaScript Object Notation.....	10
Accessing Data.....	14
Cortex XSOAR Concepts.....	26
Elements.....	26
Incident Lifecycle.....	37
SecOps Automation Scenarios.....	43
Implementing Automation Scenarios.....	43
Example Automation Scenarios .....	44
Summary.....	46

# Preface

---

## GUIDE TYPES

*Overview guides* provide high-level introductions to technologies or concepts.

*Reference architecture guides* provide an architectural overview for using Palo Alto Networks® technologies to provide visibility, control, and protection to applications built in a specific environment. These guides are required reading prior to using their companion deployment guides.

*Deployment guides* provide decision criteria for deployment scenarios, as well as procedures for combining Palo Alto Networks technologies with third-party technologies in an integrated design.

## DOCUMENT CONVENTIONS



Notes provide additional information.



Cautions warn about possible data loss, hardware damage, or compromise of security.

**Blue text** indicates a configuration variable for which you need to substitute the correct value for your environment.

In the IP box, enter **10.5.0.4/24**, and then click **OK**.

**Bold text** denotes:

- Command-line commands.

```
# show device-group branch-offices
```

- User-interface elements.

In the **Interface Type** list, choose **Layer 3**.

- Navigational paths.

Navigate to **Network > Virtual Routers**.

- A value to be entered.

Enter the password **admin**.

*Italic text* denotes the introduction of important terminology.

An *external dynamic list* is a file hosted on an external web server so that the firewall can import objects.

**Highlighted text** denotes emphasis.

Total valid entries: **755**

## ABOUT PROCEDURES

These guides sometimes describe other companies' products. Although steps and screen-shots were up-to-date at the time of publication, those companies might have since changed their user interface, processes, or requirements.

## GETTING THE LATEST VERSION OF GUIDES

We continually update reference architecture and deployment guides. You can access the latest version of this and all guides at this location:

<https://www.paloaltonetworks.com/referencearchitectures>

## WHAT'S NEW IN THIS RELEASE

Palo Alto Networks made the following changes since the last version of this guide:

- Changed wording for clarity

# Related Guides

---

Cortex™ XSOAR is a security orchestration, automation, and response (SOAR) solution that manages alerts, standardizes processes, and automates responses.

The *SecOps Automation and Response—Cortex XSOAR* suite of guides details how to use Cortex XSOAR, from understanding its concepts and user interface through deployment and using playbooks to implement a structured and automated incident response.



**SecOps: Reference Architecture Guide**—Provides solutions for prevention, detection, investigation, and response to help security-operations teams prevent threats and efficiently manage alerts.

**SecOps Automation and Response—Cortex XSOAR: Concepts Guide**—Describes concepts and terminology essential to using Cortex XSOAR in order to automate responses to security incidents.

**SecOps Automation and Response—Cortex XSOAR: User Interface Guide**—Describes user interface components that are important when you use the operations guides.

**SecOps Automation and Response—Cortex XSOAR: Deployment Guide**—Provides detailed, step-by-step instructions for deploying Cortex XSOAR, including post-installation tasks such as the required integrations to external systems.

**SecOps Automation and Response—Cortex XSOAR Phishing Investigation: Operations Guide**—Suggests a method for automatically investigating and responding to an email-based phishing incident.

## OTHER RESOURCES

**Cortex XSOAR developer hub (<https://xsoar.pan.dev>)**—Includes documentation and reference materials about all Cortex XSOAR components.

**Cortex XSOAR Administrator's Guide**—Serves as a comprehensive product reference and includes information about the numerous supported methods for installing Cortex XSOAR.

# Purpose of This Guide

---

This guide describes reference architectures for enabling a common point of integration between your cybersecurity applications, inline traffic control and enforcement devices, communication platforms, threat-intelligence services, and identity and access management. This guide includes design guidance and best practices for automating the security operations and response for your organization. This guide provides an in-depth discussion of Cortex XSOAR and how it enables your organization to implement a business process through automation.

This guide:

- Discusses the common issues impacting security operations teams.
- Describes how the use of standard data formats simplifies information sharing between applications.
- Introduces the high-level concepts for Cortex XSOAR, which enables security operations (SecOps) teams to eliminate manual tasks, identify critical threats, and take automated actions.
- Describes the in-depth technical concepts for Cortex XSOAR that you need to understand before customizing the Cortex XSOAR environment, configuring integration instances, creating playbooks, managing incidents, and running playbooks.
- Provides a framework for SecOps architectural discussions between Palo Alto Networks and your organization.
- Is required reading prior to using the [SecOps Automation and Response–Cortex XSOAR: Deployment Guide](#). The deployment guide provides step-by-step details for deploying Cortex XSOAR, as well as post-installation tasks.
- Is required reading prior to using any of the SecOps Automation and Response Operations Guides. Each of these operations guides provides an in-depth discussion of an automation scenario. Each guide also includes step-by-step configuration details for integrating Cortex XSOAR to external systems and developing a Cortex XSOAR playbook for the scenario.

## AUDIENCE

This architecture guide is for technical readers, including solution architects, security engineers, and security support staff, who want to orchestrate and automate the prevention and investigation of, and response to, security threats. It assumes the reader is familiar with the basic concepts of threat prevention, networking, and security operations and possesses a basic understanding of automation, machine learning, and analytics. This guide also assumes the reader is familiar with the basic concepts of JSON, flowcharts, conditional statements, and Boolean logic.



# Introduction

---

## THE CHALLENGES

The greatest network-security challenge organizations face is the ever-increasing onslaught of attacks. A cybersecurity “perfect storm” is underway, characterized by a pandemic-driven transition to teleworking, the flexibility demands of mobile workers, and the rapid adoption of the public cloud for business-critical applications. These factors combine to broaden an organization’s attack surface and provide more entry vectors for attackers. Meanwhile, the attackers are adopting specialized cybersecurity tools powered by cheap and plentiful compute resources to take advantage of the circumstances.

Security teams attempt to counter the attackers by adding new tools, each specifically targeting different attack vectors. This approach adds two other significant challenges. Complexity increases cumulatively as you deploy new security tools. Each tool has its own unique user interface, infrastructure requirements, and event management. As a result, security analysts spend much of their time transitioning between their disparate systems, trying to manually correlate the activities reported by their tools. With the continued addition of new tools and an increase in attacks, the volume of security alerts is constantly on the rise. Security analysts struggle with identifying false positives and duplicate incidents, which contributes to an information overload that eventually leads to burn out.

Although goals for many security operation teams, developing standard response processes and using a consistent investigation methodology prove difficult when analysts spend most of their day in rapid-response mode. Under these conditions, the response quality is highly dependent on individual analysts. Without clear and consistent procedures, error rates increase, and attacks remain undetected. Unfortunately, unlike machines, even with detailed and prescriptive procedures, humans are fallible and susceptible to fatigue.

## HOW TO RESPOND

An organization needs to swiftly and effectively respond to attacks in order to minimize the damage or eliminate the damage entirely. Rather than replace the existing tools, you can use *orchestration* to interconnect the disparate tools and access their capabilities more effectively by following a consistent process driven by automation. Another benefit of orchestration is that after you interconnect the tools, you can share relevant data across systems and eliminate manual correlation tasks. The clear benefit of this approach is that by combining your tools so they act as a single system, you can achieve more effective results. Overall, you can streamline your incident response process by eliminating manual steps and introducing automation and machine-learning where it makes sense.

For example, if one of your vendors discloses a security vulnerability, you can use Cortex XSOAR to perform a query on the Common Vulnerabilities and Exposures (CVE) record for that vulnerability and to determine if you have assets on your network that are affected by the CVE. You could also use Cortex XSOAR to initiate an incident response.

Previously, without the benefit of orchestration, the manual process would have been more cumbersome. First, you would have needed to subscribe to one or more CVE data feeds in order to receive notifications. Then for each notification, you would have had to review the details in order to identify the affected products and software versions. The final step would have been to determine if affected products were in use, and if so, their current software version. After confirming these details, you would have initiated an incident response.

Cortex XSOAR is a security orchestration, automation, and response (SOAR) solution that manages alerts, standardizes processes, and automates responses, giving your security teams more time to be more proactive and effective by eliminating multiple trivial manual tasks. Cortex XSOAR combats security challenges facing security operations with three main areas of focus: workflow automation, incident management, and collaboration. Cortex XSOAR helps security teams to reduce mean time to response, create consistent incident management processes, and increase team productivity.

Cortex XSOAR employs a visual playbook editor, a front-end for an automation engine that simplifies the implementation of security incident response workflows. There is native support for hundreds of products and thousands of actions, and you can also customize Cortex XSOAR to support any third-party devices or workflows. In addition to the automation engine, the platform includes a built-in ticketing system, collaboration facilities, and automated report generation. Using machine learning, Cortex XSOAR provides incident-handling guidance that it derives from past analyst-actions and historical information. Together, these features enable security analysts to be more productive by spending less time on routine tasks and allow them to focus their skills on the formidable challenges organizations face.

This concepts guide discusses the in-depth technical aspects of Cortex XSOAR, and it describes methods for integrating Cortex XSOAR with the Palo Alto Networks security portfolio and commonly used third-party products. You can use this guide to determine which options are most effective in meeting your organization's business requirements. This guide also includes an overview of the key data management concepts with which you should be familiar before using Cortex XSOAR.

## CORTEX XSOAR FOR SECOPS

Cortex XSOAR (formerly *Demisto*) simplifies and streamlines the work of security operations in multiple ways.

### Integrating Disparate Systems

As the security industry has evolved, many organizations have collected siloed security tools lacking integration capabilities. Each tool has its own unique user interface and takes time to master. It is inefficient and time consuming to use these tools independently and then manually correlate attack details across systems.

Cortex XSOAR integrates multiple external systems and provides orchestration across these systems. In this role, Cortex XSOAR functions as a communication broker and enables your security team to correlate and respond across platforms. Cortex XSOAR integrates with the external systems using open APIs, but as a security analyst, you do not need to have broad API knowledge to use Cortex XSOAR effectively. Palo Alto



Networks maintains a marketplace for Cortex XSOAR content, which you can access directly through the Cortex XSOAR web interface. The marketplace includes published content from both Palo Alto Networks and third-party vendors. All contributors must submit the content they develop to Palo Alto Networks. The Cortex XSOAR Content Team reviews all content and must approve it before it becomes available to customers.

## Enabling Rapid Response

During the early stages of an incident, response time is critical, and a manual process can never compete with automation. If you can quickly confirm an attack, identify the affected people and devices, and respond, you can minimize or eliminate the damage from the attack. However, if you delay the response even slightly, the overall damage from the attack can increase significantly. If your analysts manually respond to alerts, this gives the attackers a head start that is difficult to overcome. When the information you need is scattered across multiple systems, it takes you longer to fully understand the attack.

Cortex XSOAR improves your ability to respond quickly by automating time-consuming, mundane tasks and by quickly gathering all the information you need from your different systems. For certain incident types, you can automatically execute Cortex XSOAR playbooks without waiting for any manual intervention.

## Using Consistent Processes

Analysts with differing skillsets and experience do not consistently treat incidents the same way and reach the same conclusions. After you have determined an effective process, you want it consistently and rigorously followed.

As you transition to using Cortex XSOAR, you translate your security-analysis business processes into playbooks. Each playbook aligns to a specific workflow with individual tasks. With Cortex XSOAR, when a security analyst investigates an attack, they execute the playbook for that attack type. Because Cortex XSOAR guides them along the way, each analyst follows the same standardized response process.

## Preserving and Sharing Operational Knowledge

Veteran team members, through their extensive experience, often develop skills that rely on unwritten information. They sometimes share this knowledge informally, through mentoring or shadowing, but without a formal process to document this information, when those individuals change roles or leave the organization, their information disappears with them.

If you embed the acquired knowledge into a formalized business process, you can retain information even when staff changes occur. The playbooks you develop for Cortex XSOAR should be continuously expanded, updated, and revised to include best practices commonly used by team members. When used in this way, the playbooks become active documentation and provide a medium you can use to capture unwritten information.

## Reducing Errors

During security investigations, manual processes are inherently more error prone than automated processes. For example, if you are transcribing a username or an IP address from one system's user interface to another, you can easily mistype the data or omit a character during a copy-and-paste operation. Accuracy and attention to detail is essential for a successful investigation, and even the simplest of errors can have a huge impact.

If you use machine-to-machine communication to share the data, those types of errors are virtually impossible. Through automation with Cortex XSOAR, you can identify the fields that are common across platforms and map them into a shared data repository. This improves the overall speed and accuracy of the investigation and creates a digital audit trail for the key elements of the investigation. Cortex XSOAR preserves all data and presents the data in a format suitable for rapid analysis.

## Improving the Use of Skilled Resources

Many cybersecurity jobs require significant experience and skills. Skilled cybersecurity workers are scarce, which makes it hard to recruit for the knowledge and skills that organizations need. After you have skilled team members, you do not want to assign them mundane, repetitive tasks that an automation could easily accomplish. Your skilled cybersecurity staff should spend their time working on more user-intensive tasks or complex investigations that require detailed analysis.

Adding Cortex XSOAR to your existing set of security tools enables your organization to achieve better use of skilled resources. You can replace manual processes with playbook-driven automation, typically getting faster and more accurate results. Cortex XSOAR can collect data from multiple systems, enrich the information by using threat-intelligence feeds, and correlate information seen in current or past incidents. After the initial automated data collection and enrichment phase is complete, your team can continue the investigation using Cortex XSOAR to complete the analysis. This approach is a much more effective way to make use of your limited resources and is also likely to improve their efficiency.

## Providing Actionable Documentation

Well-run security teams have documented processes and do their best to follow them. Many of these processes include flowcharts and checklists to guide an analyst about how to respond to a particular incident type. These types of processes tend to rely strongly on interpretation, especially when they include written instructions for complex security analysis tools.

Cortex XSOAR combines documented processes with playbook-driven automation. A Cortex XSOAR playbook includes both automated and manual tasks and minimizes the need for any user-specific interpretation. Where possible, Cortex XSOAR fully automates tasks. Where necessary, Cortex XSOAR involves the appropriate team member to provide missing data, perform a specific task, or review results. As a result, the documented process is closely aligned to the automated process. This approach allows your organization to develop and implement more consistent and actionable processes.

## Performing Structured Incident Investigation

Most organizations use trouble-ticketing systems to track the status of an investigation and assign resources to specific tasks. General purpose trouble-ticketing systems, such as ServiceNow, work well for traditional support models but are not designed for comprehensive security analysis and response. These systems can track tasks and assign resources, but without extensive customization, they are not suitable for correlating and presenting security threat-related information in a useful way.

When performing an investigation, Cortex XSOAR collects and stores all information retrieved by an automation task (or entered in a manual task) and preserves it as part of the incident. Cortex XSOAR also includes capabilities to tag artifacts derived during the investigation as evidence. Cortex XSOAR also excels at correlating and presenting threat-related information. You import your threat-intelligence feeds directly into XSOAR and use those feeds to analyze the indicators of compromise within your security incidents.

Although not intended to replace tools like ServiceNow, Cortex XSOAR includes a native ticketing system to manage security incidents throughout their entire lifecycle. You can also integrate Cortex XSOAR directly with many common trouble-ticketing systems, which you can then use to automatically create Cortex XSOAR incidents and initiate your response playbooks.

## Cortex XSOAR and the Multi-Platform Evolution

To get the most out of your security toolset, improve response time, and use resources more effectively, you need to ensure your systems work together. Cortex XSOAR, in combination with your other security tools, enables consistent incident-response capabilities and extends access across multiple platforms.

If data is not available, or analysts cannot access the data, you cannot respond effectively. Cortex XSOAR provides an option for the SOC to attain bounded access to restricted systems. With integrations, you can configure your external systems to enable Cortex XSOAR to perform only the required actions. SOC users do not require full access to these systems.

After the peak anxiety period during an active event response has passed, relevant information becomes increasingly difficult to collect and details are harder to recall. Cortex XSOAR retains a full-fidelity historical timeline for each incident, with a detailed record of all analysis and response actions, which also includes artifacts and related metrics. You can use this information for after-action-reports and for retrospective analysis to support continuous process improvement.

The speed of the response, quality of the analysis, and confirmation through detailed evidence are critical to a quick and effective response so that your organization can confidently defend against threats and rapidly thwart attacks.

# Data Management Concepts

---

One of the key functions of Cortex XSOAR is to integrate with disparate systems in order to facilitate and secure end-to-end workflows. It is collecting data from each of these systems and needs to have a standards-based method to normalize and store this diverse information. Understanding how this data is stored and how to access specific information is important in building effective playbook tasks and performing efficient searches in Cortex XSOAR.

## JAVASCRIPT OBJECT NOTATION

When Cortex XSOAR creates an incident and then proceeds to execute a playbook, it generates a significant amount of data that you need to track. Most of this information takes the form of a setting or variable and its associated value. For example, an incident will have an id number, a severity (low, medium, high), a type (exploit, malware, phishing), and so on. Cortex XSOAR stores all incident-related information as context data.

Internally, Cortex XSOAR stores context data as a JavaScript Object Notation (JSON) object. By using this format, Cortex XSOAR normalizes information acquired from external systems into a common format.

Having a basic understanding of JSON enables you to quickly access and extract information from the context data, use this information effectively within your playbook tasks, and perform advanced functions such as formatting task-input parameters and selecting data for custom reports.

Although JSON resembles JavaScript object literal syntax, it is not dependent on the JavaScript programming language. JSON provides a language-independent method for representing structured data in a standard format that is both easy for machines to parse and reasonable for humans to read. Developers commonly choose JSON format when they need to transmit data across a network between web applications, because JSON is lightweight and stored as a string.

A JSON object exists as:

- A set of unordered key/value pairs, similar to a Python dictionary or an associative array in C.
- An ordered list of values, similar to a Python list or an array in C.
- A nested combination of both unordered key/value pairs and an ordered list of values.

### JSON Object Example 1

A JSON key is a string enclosed in double quotes followed by a colon. A JSON value can be of type string (enclosed in double quotes), number, boolean (true/false), null, array or another nested JSON object. You then enclose the entire key/value definition in curly braces.

Here is a simple JSON object:

```
{
  "name": "Amy"
}
```

In this example, the key is **name** and the associated value is the string **Amy**.

### JSON Object Example 2

You can add more key/value pairs to your JSON object. However, you must ensure there is a comma between each of the pairs and ensure the last pair does not have a trailing comma. The following example JSON object adds additional key/value pairs and introduces different value types.

```
{
  "name": "Amy",
  "title": "Security Architect",
  "id": 1234,
  "certifications": ["PCNSE", "PCCSE", "CISSP"],
  "full_time": true
}
```

In this example, both **Amy** and **Security Architect** are string values, **1234** is a number value, and **true** is a Boolean value.

The *certifications* key uses an array for its value. The array is enclosed by square brackets, and items are separated by a comma. Although in this example the array values are all strings, items in the array do not have to be of the same type. Within a JSON array, you can assign array elements to be different types if required. For example, element 1 value is a string, element 2 value is a number, and element 3 value is a Boolean.

### JSON Object Example 3

Using the building blocks from the previous examples, you can create more sophisticated JSON objects to meet your needs. You can nest JSON objects or create an array of JSON objects to represent your data. You can extend the previous example to represent multiple employees by creating a key called *employees* with an array as the value where each item in the array is itself a JSON object.

```
{
  "company": "Example Co",
  "location": "San Francisco, CA",
  "employees": [
    {
      "name": "Amy",
      "title": "Security Architect",
      "id": 1234,
      "certifications": ["PCNSE", "PCCSE", "CISSP"],
      "full_time": true
    },
    {
      "name": "Brian",
      "title": "Security Engineer",
      "id": 5678,
      "certifications": ["PCNSA", "CCIE"],
      "full_time": true
    },
    {
      "name": "Chris",
      "title": "Security Analyst",
      "id": 9012,
      "certifications": [],
      "full_time": false
    }
  ]
}
```

### JSON Objects in Cortex XSOAR

Cortex XSOAR stores all data associated with an incident as a JSON object. This complete set of information is called *context data*. To help you interpret it, Cortex XSOAR includes a user-friendly tool. The Cortex XSOAR context-data explorer uses a graphical tree, similar to a file browser, to present the context data.



In the following example, you use the context-data explorer to view context data for an incident. After expanding the ServiceNow object, you can see each of the key/value pairs that the object includes.

The screenshot shows a 'Context Data' window with a search bar and a tree view. The tree view is expanded to show the 'ServiceNow' object, which contains the following key-value pairs:

- OpenedBy: 5927e80bb1f9127512d6b44cf15ad330
- Priority: 5 - Planning
- Active: true
- Summary: Permit domain apple.com
- CreatedOn: 2021-02-16 21:39:58
- Number: INC0010122
- Creator: 5927e80bb1f9127512d6b44cf15ad330
- State: 1
- ID: 654a7176db122010891f2fb7489619ea
- OpenedAt: 2021-02-16 21:39:33

For the ServiceNow object, the corresponding JSON object is as follows.

```
{
  "ServiceNow": [
    {
      "OpenedBy": " 5927e80bb1f9127512d6b44cf15ad330",
      "Priority": "5 - Planning",
      "Active": true,
      "Summary": "Permit domain apple.com",
      "CreatedOn": "2021-02-16 21:39:58",
      "Number": "INC0010122",
      "Creator": " 5927e80bb1f9127512d6b44cf15ad330",
      "State": 1,
      "ID": "654a7176db122010891f2fb7489619ea",
      "OpenedAt": "2021-02-16 21:39:33"
    }
  ]
}
```

Although only a few advanced functions in Cortex XSOAR require you to manipulate JSON objects directly, understanding the basics of JSON helps you navigate context data and configure your playbook tasks.

## ACCESSING DATA

### Using Context Data in a Playbook Task

Many playbook tasks use information from context data. To use this information effectively in your tasks, you must be able to extract values in the required format from the context data.

If you know the key, the key location in the context data, and the value is already in the correct format, then you can use a direct reference to the key when you configure a task.

The first step in determining a direct reference is to identify the path to the key in the context data. Direct references use dot notation to represent the path through the context data tree and uniquely identify the key you want. *Dot notation* functions similarly to how a directory path is represented in a file system where each level of the hierarchy is listed left-to-right, except that dot notation uses a dot (period) as a separator instead of a slash.

#### Dot Notation Example

This example uses the following context data.

Context Data	
	Number
▼ root: {} 3 items ↗	
▼ ServiceNow: {} 1 item ↗	
▼ Ticket: {} 1 item ↗	
	<b>Number: INC0010122</b>

You can reference the ServiceNow ticket number by using dot notation.

```
ServiceNow.Ticket.Number
```

The dot notation is the unique path to the *ServiceNow.Ticket.Number* object in the context data.

#### Cortex XSOAR Direct Reference

After you have determined the dot notation that describes the path to the desired context data, you can specify the direct reference for Cortex XSOAR.

Enclose the path to the object in curly braces and prepend with the \$ to format a Cortex XSOAR direct reference.

```
${ServiceNow.Ticket.Number}
```

When executing an automation task, Cortex XSOAR evaluates this expression to first identify the key in the context data and then get the value for the key. The automation task then uses this value.

With the example context data, when Cortex XSOAR executes an automation task, it evaluates the direct reference `${ServiceNow.Ticket.Number}` as `INC0010122`.



#### Note

Cortex XSOAR provides a **Set** automation script that you can use to create key/value pairs within the context data. If you need to access the value for a key that you created using Set, you must use the direct reference method.

Example:

If you set the key/value pair of

```
"ExampleKey": "Example value"
```

you use

```
${ExampleKey}
```

as the direct reference to retrieve the value.

### Cortex XSOAR Task with Direct Reference as an Input

A Cortex XSOAR automation script such as **Print** requires an input **value**. When you execute this task, Cortex XSOAR prints the text that you enter in the **value** box.

Commonly, when using **Print**, you simply provide a static text string as the input when you configure the task.

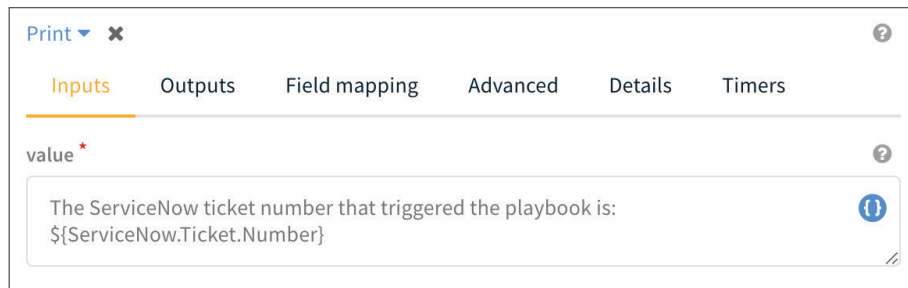
This automation generates the following output.

```
Task Result #7: Print hello world
Command: !Print value="Hello world." (Scripts)

Hello world.
```

When you need to include static text with a context data value as a task input, or when you need to include

multiple context data values as a task input, use the direct reference format for each context value.



This automation generates the following output.

**Task Result #18: Print ServiceNow Ticket Number** [Go to Task](#)

**Command:** `!Print value="The ServiceNow ticket number that triggered the playbook is: INC0...` (Scripts) [Refresh](#) [Stop](#) [Help](#)

The ServiceNow ticket number that triggered the playbook is: INC0010122

## How to Determine Context Data Path

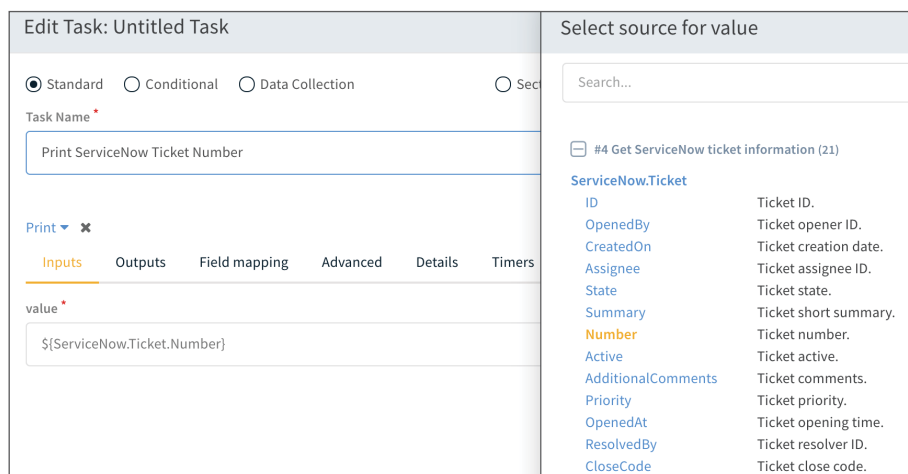
When you are configuring a Cortex XSOAR automation task, but you do not know the key or the key location in the context data, you can use the Cortex XSOAR Select Source For tool.

When you are configuring an automation task, you can launch the Select Source For tool for any input by clicking the [i](#) icon in the corresponding box.



With this tool, you can browse for specific keys or you can use the search feature. After you identify the desired key, you can click to choose the key. Using this method automatically generates the direct reference and then populates the task box with it.

The following example displays the direct reference that Cortex XSOAR generates for the **value** input when you choose the **Number** key from the ServiceNow.Ticket context data.



If the context data value you need is already in the correct format that the Cortex XSOAR automation task requires, you should use the direct reference method.

If you are unfamiliar with Cortex XSOAR, you should start by using Select Source For tool to search for and choose your direct references. However, there is no functional difference between the method of entering the value directly when you know the key and location and the method of searching for and choosing the key when you are uncertain of the key and location.

**Note**

When using the **Set** automation script to create key/value pairs within the context data, you cannot use the Select Source For tool to choose the direct reference.

Be sure to note the keys that you define when using Set, and then format and enter the appropriate direct reference syntax when needed.

## Filters and Transformers

If the context data contains the value you need, but the value is not in the correct format that the Cortex XSOAR automation task requires, you need to filter and/or transform the context data before you can use it in an automation task.

If you need to be selective about which context data values a task uses from the context data, then you can use a Cortex XSOAR filter to select values from the context data only if the values meet the filter operator's match criteria.

Some filter examples include:

- **Starts with** operator—Match only strings that start with “INC”
- **Greater than** operator—Match only numbers greater than 7

If you need to modify a context data value before using it within an automation task, then you can use a Cortex XSOAR transformer to modify a value from the context data by using a transform operator.

Some transformer examples include:

- **To lower case** operator—Transform all letters in a string to lower case
- **Round** operator—Transform a floating point number to nearest integer

You can use filters and transformers individually or together. If you apply multiple filters/transformers, Cortex XSOAR first applies all filters in the order you specify and then applies all transformers in the order you specify.

As an alternative to using a direct reference, Cortex XSOAR also references context data values by using a get/where/transformers (GWT) structure.

Get	ServiceNow.Ticket.Number
Where	No filters applied
Transformers	No transformers applied

The GWT structure includes three fields:

- **Get**—Required field that you use dot notation to specify the context data key.
- **Where**—Optional field that you use to apply filters. When you apply a filter, you typically match values from the same context data key that you specified in the Get field. In some more complex examples that involve lists, you might choose to match values from a different context data key. In the simplest examples, your filter consists of a single filter operator. However, you can combine multiple filter operators in a single filter, which Cortex XSOAR evaluates using a logical **OR**. You can also use multiple filters, which Cortex XSOAR evaluates using a logical **AND**.
- **Transformer**—Optional field that you use to apply transformers. When you apply a transformer, it operates on the value that you specify in the Get field. If you have applied any filters, Cortex XSOAR applies them before applying any transformers. If necessary, you can use multiple transformers, and Cortex XSOAR applies them in the order you specify.

If you prefer, even though it is more complex to configure, you can use the GWT structure in your automation tasks without applying any filters or transformers. However, if you use the GWT structure, you can only reference a single value and you cannot combine a value you reference with GWT structure with plain text.

These two examples both produce the same result.

```
Get ServiceNow.Ticket.Number
Where
Transformers
```

```
${ServiceNow.Ticket.Number}
```

To apply filters or transformers to a context data value, you must use the GWT structure. To configure an automation task with inputs that use the GWT structure, you must use the Select Source For tool.

The purpose of the following examples is to illustrate the proper usage of GWT structures when filtering or transforming context data.



## GWT Example 1

In this example, the context data consists of a simple array of key/value pairs.

```
{
  "Organization": [
    {
      "Domain": "example.com"
    },
    {
      "Domain": "example2.net"
    },
    {
      "Domain": "example3.com"
    }
  ]
}
```

First, you create a GWT structure that uses Get to access the *Organization.Domain* value.

### GWT Structure

Get	Organization.Domain
Where	No filters applied
Transformers	No transformers applied

### Context Data Returned Value After Applying GWT Structure

```
▼ root: [] 3 items
0: example.com
1: example2.net
2: example3.com
```

Next, you apply a filter to match only domains with the .com suffix. This filter excludes the domain with the .net suffix.

### GWT Structure

Get	Organization.Domain
Where	Organization.Domain Ends with .com
Transformers	No transformers applied

### Context Data Returned Value After Applying GWT Structure

```

▼ root: [] 2 items
0: example.com
1: example3.com

```

Alternatively, you apply a transformer to rewrite the domain, using uppercase characters.

### GWT Structure

Get	Organization.Domain
Where	No filters applied
Transformers	To upper case

### Context Data Returned Value After Applying GWT Structure

```

▼ root: [] 3 items
0: EXAMPLE.COM
1: EXAMPLE2.NET
2: EXAMPLE3.COM

```

### GWT Example 2

In this example, the context data consists of an array of nested JSON objects, where each object contains a set of key/value pairs.

```

{
  "WildFire": [ {
    "Report": [
      {
        "URL": "url-01.example.com",
        "verdict": "benign"
      },
      {
        "URL": "url-02.example.com",
        "verdict": "malware"
      },
      {
        "URL": "url-03.example.com",
        "verdict": "phishing"
      }
    ]
  }
]
}

```

First, you create a GWT structure that uses Get to access the *WildFire.Report* value.

#### GWT Structure

Get	WildFire.Report
Where	No filters applied
Transformers	No transformers applied

#### Context Data Returned Value After Applying GWT Structure

▼ root: [] 3 items
▼ 0: {} 2 items
URL: url-01.example.com
verdict: benign
▼ 1: {} 2 items
URL: url-02.example.com
verdict: malware

Next, you change the Get to access the *WildFire.Report.URL* value. You also apply a filter to only match when the *WildFire.Report.verdict* is either malware or phishing. This filter eliminates the URL with the benign verdict.

#### GWT Structure

Get	WildFire.Report.URL
Where	WildFire.Report.verdict Contains malware OR WildFire.Report.verdict Contains phishing
Transformers	No transformers applied

#### Context Data Returned Value After Applying GWT Structure

▼ root: [] 2 items
0: url-02.example.com
1: url-03.example.com

Finally, you apply a transformer that counts how many URLs had a malicious verdict (either malware or phishing).

#### GWT Structure

Get	WildFire.Report
Where	WildFire.Report.verdict Contains malware OR WildFire.Report.verdict Contains phishing
Transformers	Count

## Context Data Returned Value After Applying GWT Structure

2

## Searching with Cortex XSOAR

Cortex XSOAR collects and stores a substantial quantity of security information and artifacts related to incidents that you investigate. To streamline access to this information, Cortex XSOAR includes broad search capabilities. You need to understand how to properly craft search queries to quickly identify relevant information and to include queries within automation tasks.

If you execute a search using the search box that Cortex XSOAR includes in the top right hand corner of every page, your search applies to all incidents, indicators, evidence, entries, and investigations. If Cortex XSOAR finds any matches, the search results include a summary for each match. You can directly access the details for any matches from the summary.

On the incidents, indicators, jobs, playbooks, automation and evidence board pages, Cortex XSOAR provides a query box. If you execute a search using the query box, your search is specific to the current screen. For example, on the incident screen, the search query only searches within incidents. If Cortex XSOAR finds any matches, the search results include the details of each match.

To properly format a search using Cortex XSOAR, you use the Lucene Query Syntax, an open-source project from Apache. For additional details on how to use Lucerne Query Syntax, see the reference page [https://lucene.apache.org/core/2\\_9\\_4/queryparsersyntax.html](https://lucene.apache.org/core/2_9_4/queryparsersyntax.html).

The valid list of fields for a search depends on whether you are using the search box, or a page-specific query box. If you perform your search using the search box, you can specify the category for your search by using the primary fields listed in Table 1. The primary fields require dependent secondary fields, which are specific to the primary fields. You separate the primary and secondary fields using a dot.

*Table 1 Search box primary fields*

Search field	What the search returns	Example query
<b>entry</b>	Results from the war room	entry.id:25@296
<b>evidence</b>	Items from the evidence board	evidence.id:*@220
<b>incident</b>	Incidents (with matching incident fields)	incident.owner:brian
<b>indicator</b>	Indicators	indicator.value:example.com
<b>investigation</b>	Investigation IDs (searches among incidents)	investigation.created:>="2021-03-01"

If you perform your search by using the page-specific query box, the primary field is pre-determined and you need to specify only the secondary field.

If you do not specify a field, Cortex XSOAR assumes a default field, which is dependent on where you perform your search. You can build complex search queries using operators, such as **AND** and **OR**. You can also use the prohibit operator - to exclude matching results.

With all searches, you must specify a term for which you search. If you specify a field and a term, then you separate them with a colon. If the term includes any spaces, you must include double quotes at the beginning and end of the term. If necessary, you can also use regular expressions to match items within the term.

For most search types, to quickly access the information you need, you should use the page-specific query boxes. When you use the arrow keys within the search boxes, Cortex XSOAR provides a list of valid fields.

### Example Search 1

This example shows two search methods that return only indicators with type URL.

Search box:

```
indicator.type:URL
```

Indicator page query box:

```
type:URL
```

### Example Search 2

In this section, the example queries use the incident page query box.

This example shows a search to display only incidents that are not closed (active or pending).

```
-status:closed
```

This example shows a search to display only incidents that have a severity of high or medium and that have an active status:

```
(severity:high or severity:medium) and status:active
```

## Regular Expressions

Regular expressions, commonly referred to as a *regex*, are a mini programming language designed to find patterns in text. As an example, you can employ them in an application such as a text editor to find and replace a word throughout a document. If you work with a command line and have used an asterisk to list all files with a particular extension (for example, \*.txt) then the concept of regular expressions should be familiar. To find and extract specific sequences of characters from large text documents or string sets, you can construct complex regular expressions. Cortex XSOAR includes a regular expression utility to support the use of regular expressions when you perform searches.













Metacharacters are designated symbols with special meaning used by regular expressions to match patterns in text. Inside a regular expression, you evaluate a period to mean “match any single character” and you evaluate a `\d` to mean “match any single digit.” Regular expression utilities consider non-metacharacters to be regular characters and match them literally. A regular expression is the combination of metacharacters and literal characters that constitute a search string, which you then use to evaluate and compare against a target text document or string.

Table 2 Commonly used regular expression metacharacters

RegEx metacharacter	Function
.	Matches any single character.
?	The preceding item matches zero or one time.
*	The preceding item matches zero or more times.
+	The preceding item matches one or more times.
[abc]	Character set. A single character of a, b or c.
[a-z]	Character set. A single character from the range of a through z lower case.
\d	Any digit.
\s	Any whitespace character (space, tab).
a{3}	Match exactly 3 of the preceding item (aaa).
a{1,4}	Match from 1 to 4 of the preceding item (a, aa, aaa, aaaa).
^	Start of line.
\$	End of line.

For some more complex matches, you may want to use the literal character itself instead of its special metacharacter meaning. For example, if you want to match `+` as the addition operator instead of matching one or more instances of the preceding character, you escape the metacharacter by putting a back slash in front of it. For example, to indicate addition, you would use `2\+2=4`.

Cortex XSOAR uses regular expressions extensively to classify string values into data types. For predefined indicator types, Cortex XSOAR provides the regex value that matches each type. If you need to define your own indicator types, then you would need to determine the regex values for them.

<input type="checkbox"/>	Name	Enabled	Reputation command	Formatting script	Regex ↓	Layout	Content Pack
<input type="checkbox"/>	 CVE	True	cve		<code>CVE-\d{4}-\d{4},7</code>	CVE Indicator	Common Types
<input type="checkbox"/>	 Email	True	email		<code>\b[A-Za-z0-9._%#+=\p{L}-]+@[A-Za-z0-9\p{L}-]+\.[A-Za-z]{2,}\b</code>	Email Indicator	Common Types
<input type="checkbox"/>	 File SHA-256	False	file		<code>\b[a-fA-F\d]{64}\b</code>		Common Types
<input type="checkbox"/>	 File SHA-1	False	file		<code>\b[a-fA-F\d]{40}\b</code>		Common Types
<input type="checkbox"/>	 File	True	file		<code>\b[a-fA-F\d]{32}\b \b[a-fA-F\d]{40}\b \b[a-fA-F\d]{64}\b \b[a-fA-F\d]{128}\b</code>	unifiedFileRep	Common Types
<input type="checkbox"/>	 File MD5	False	file		<code>\b[a-fA-F\d]{32}\b</code>		Common Types
<input type="checkbox"/>	 IP	True	ip	UnEscapeIPs	<code>\b(?:??:25[0-5] 2[0-4][0-9] 1[0-9][0-9] 1-9?[0-9])?(\. \ \. \ \.){3}(?:25[0-5] 2[0-4][0-9] 1[0-9][0-9] 1-9?[0-9])\b</code>	IP Indicator	Common Types
<input type="checkbox"/>	 CIDR	True	cidr		<code>\b(?:??:25[0-5] 2[0-4][0-9] 1[0-9][0-9] 1-9?[0-9])?(\. \ \. \ \.){3}(?:25[0-5] 2[0-4][0-9] 1[0-9][0-9] 1-9?[0-9]) (\/\( - )\ 1-2)[0-9]{3}[0-2]    \b</code>		Common Types
<input type="checkbox"/>	 IPv6CIDR	True			<code>\b(?:[0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}((:[0-9a-fA-F]{1,4}){1,7}): ((?:0-9a-fA-F){1,4}){1,6}:(?:0-9a-fA-F){1,4} (:(?:0-9a-fA-F){1,4}){1,5}(:[0-9a-fA-F]{1,4}){1,5} (?:0-9a-fA-F){1,4}(:[0-9a-fA-F]{1,4}){1,4} (:[0-9a-fA-F]{1,4}){1,3} (:[0-9a-fA-F]{1,4}){2} (:[0-9a-fA-F]{1,4}){1,2} : \b(?:0-9a-fA-F){1,4}\b</code>		Common Types
<input type="checkbox"/>	 Domain	True	domain	ExtractDomainAndFQDNFromIriAndEmailUnEscapeURLs	<code>(?:\b(?:[0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}((:[0-9a-fA-F]{1,4}){1,7}): ((?:0-9a-fA-F){1,4}){1,6}:(?:0-9a-fA-F){1,4} (:(?:0-9a-fA-F){1,4}){1,5}(:[0-9a-fA-F]{1,4}){1,5} (?:0-9a-fA-F){1,4}(:[0-9a-fA-F]{1,4}){1,4} (:[0-9a-fA-F]{1,4}){1,3} (:[0-9a-fA-F]{1,4}){2} (:[0-9a-fA-F]{1,4}){1,2} : \b(?:0-9a-fA-F){1,4}\b (?:(?:[0-9a-fA-F]{1,4}){1,4} (?!(?:[0-9a-fA-F]{1,4}){1,4}) (?!(?:[0-9a-fA-F]{1,4}){1,4}) (?!(?:[0-9a-fA-F]{1,4}){1,4}) (?!(?:[0-9a-fA-F]{1,4}){1,4}))\. \b(?:0-9a-fA-F){1,4}\b</code>	Domain Indicator	Common Types
<input type="checkbox"/>	 URL	True	url		<code>\b(?:https? ftp http https?)\:\/\/\S*\b</code>	URL Indicator	Common Types
<input type="checkbox"/>	 IPv6	True	ipv6	VerifyIPv6Indicator	<code>(?:[a-fA-F0-9]{1,4}:){1,2}[0-9a-fA-F0-9]{1,4}</code>		Common Types



You can use regular expressions within Cortex XSOAR automation tasks for a variety of reasons:

- To ensure that a user properly formats input and the value is within the valid range (example: IP address)
- To extract information from an input and classify the information to a specific type (example: URL)

### Example Regular Expression

In this example, you want to ensure that a ticket number matches the format of a ServiceNow incident number and are using the Cortex XSOAR **MatchRegexV2** automation script.

ServiceNow incident numbers start with the uppercase letters INC, which is followed by seven digits (example: INC0010249).

An example regular expression that matches this pattern is `^INC\d{7}$`.

*Table 3 Component of the regular expression*

Component	Details
^	Start of line. Ensures the regex will not match if there are characters in front of INC.
INC	Must match these three upper case letters in this order.
\d	Any digit. Same as using [0-9] to match any single number
{7}	Exactly seven of \d (exactly 7 digits).
\$	End of line. Ensures the regex will not match if there are more than 7 digits at the end.

You can use this regular expression directly in a Cortex XSOAR automation task.

regex \*

`^INC\d{7}$`



#### Note

To use a regular expression with a search box or query box, use the **value** field with the regular expression as the term. You must surround the regular expression with `"/` at the beginning and `/"` at the end.

Example regular expression that matches `www*.com`:

```
w{3}.*.com
```

Search phrase:

```
value: "/w{3}.*.com/"
```

# Cortex XSOAR Concepts

---

The SecOps group within a typical organization uses and accesses a variety of tools, including cybersecurity applications for logging and event management, inline traffic control and enforcement devices, communication platforms, threat intelligence services, and an identity and access management framework. These tools are all critical components in the ongoing effort to defend against cyberattacks.

Cortex XSOAR does not replace this broad set of tools but instead provides a common point of integration that, using orchestration and automation, communicates with these existing tools and enables your organization to implement a process for interacting with these tools.

## ELEMENTS

This section introduces the basic concepts of Cortex XSOAR and describes how the various components of the system interact.

The information in this guide is based on Cortex XSOAR version 6.0.

### Incidents

You use a Cortex XSOAR *incident* to manage the lifecycle of any potential security data threat that you have to identify and remediate. After you create an incident, either manually or through automation, you follow a *playbook* to gather information from your cybersecurity applications and other tools. Like a trouble-ticketing system, Cortex XSOAR facilitates the investigation of the incident. Each incident includes case details, which include the owner and severity, the assigned team members, a list of related incidents, and information related to the incident that Cortex XSOAR has collected.

While the incident is under investigation, Cortex XSOAR displays the *playbook's* progress and highlights any tasks that require attention from team members. Cortex XSOAR also hosts a virtual war room for each incident. You use the war room to run commands and review their results and to collect artifacts and evidence that are critical to the incident investigation. The war room provides a chronological journal of all activities related to the incident.

After you complete an investigation, you close the incident. However, Cortex XSOAR retains the complete incident history that you can review at any time. Cortex XSOAR also uses the historical information in order to identify related incidents, which share similar characteristics such as malicious indicators.

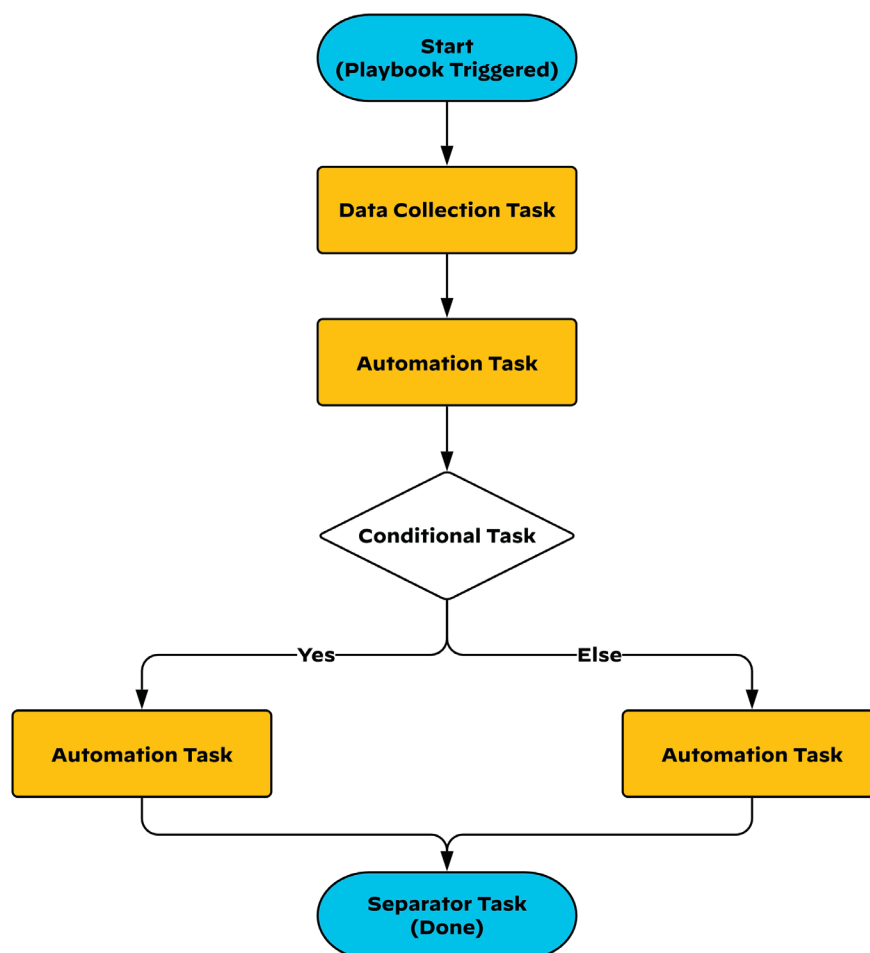
### Playbooks

*Playbooks* are the core element of Cortex XSOAR. The Cortex XSOAR *playbook* structure is similar to a flowchart and enables you to organize and automate many of your security processes, including incident investigation. The primary value of a *playbook* is that you can automate your incident response, you can ensure that the response is consistent and structured, and you can capture and preserve evidence obtained during the response.

A standard installation of Cortex XSOAR includes many system playbooks. You can also add playbooks to Cortex XSOAR by installing content packs from the Cortex XSOAR Marketplace. Cortex XSOAR tags these two types of playbooks as *system* playbooks.

As you gain experience with Cortex XSOAR, you can start to build your own custom playbooks. You access both system playbooks and custom playbooks from the Cortex XSOAR playbook library. The primary focus of this guide is to provide the background knowledge you require so that you can either develop your own playbooks or customize system playbooks.

Figure 1 Example playbook



You can view the progress of a playbook by using the Cortex XSOAR work plan, which provides a real-time visual representation of a running playbook. If the playbook requires manual input from a security analyst, they provide it directly within the work plan.

## Where to Get Playbooks and Content for Cortex XSOAR

The Cortex XSOAR Marketplace is a digital storefront for collaborating on, contributing to, and sharing security orchestration content for Cortex XSOAR. You access the marketplace directly from Cortex XSOAR. Within the marketplace, you share or download materials in content packs. Each content pack includes a repository of required and optional elements that Cortex XSOAR imports when you install the content pack.

These elements may include:

- Integrations and automation commands.
- Automation scripts.
- Playbooks.
- Reports and widgets.
- Dashboards and layouts.
- Incident types and layouts.
- Classifiers.
- Indicator types and fields.

Palo Alto Networks, Cortex XSOAR technology partners, and third-party developers all publish content packs to the marketplace. Most content on the marketplace is free, and premium content typically includes free trial periods for evaluation. Support for marketplace content varies depending on the publisher. Contributions can be either officially supported (by Palo Alto Networks, a technology partner, or an individual developer) or community supported. In the marketplace, for all content packs published by Cortex XSOAR, Palo Alto Networks provides support. For other content packs, partners and developers provide support.

Developers provide periodic updates to their content packs and provide notifications through the marketplace. To keep Cortex XSOAR current, you should plan to regularly review and install available updates.

### Tasks

Playbooks are composed of *Tasks*, the building blocks of each playbook. Each task is an autonomous element, and although you interconnect tasks within a playbook, data does not flow directly between tasks. In any task, you can use context data created by any previously completed task as an input, regardless of whether a connection between the tasks exists. You should consider each task to be a part of the overall business process defined by the playbook.

In a Cortex XSOAR playbook, you can create the following task types:

- **Standard**—You use this task type to invoke an automation. After you choose an automation for the standard task, you typically need to specify one or more input arguments from the context data for the automation. After you run a standard task, if the task includes outputs, Cortex XSOAR adds the specified outputs to the context data. You can also configure the task to add its results as an incident note or as incident evidence. When creating a standard task, you can choose from automations in the Cortex XSOAR Task Library.
- **Conditional**—You use this task type to choose a branch when your playbook has multiple branches. In its simplest form, Cortex XSOAR chooses a branch based on a parameter in the context data. If the parameter matches a certain value, the condition is true and the playbook continues execution using the “true” path. If the parameter does not match a certain value, the condition is false, and the playbook continues execution using the “false” path. In other situations where you need to filter or transform the context data derived from previous tasks, the conditional task supports a broad range of comparison operators, which for complex conditions you can combine by using Boolean logic.

Cortex XSOAR can also use a conditional task to interact directly with the task owner by asking a question and capturing the answer. If the case owner is logged into Cortex XSOAR, they can respond directly within the incident workplan. You can also configure Cortex XSOAR to send an email or Slack message to the case owner, which prompts them to respond using a webform.

- **Data collection**—You use this task type to collect responses to a specific set of questions, referred to as a *survey*. Cortex XSOAR adds the responses to the context data. When you add this task type, you can specify the list of survey recipients by name or by role. If they are logged into Cortex XSOAR, they can respond directly within the incident workplan. You can also configure Cortex XSOAR to send the survey notification through email or Slack message, which prompts the recipient to respond through a web form. By default, if Cortex XSOAR receives one reply, it marks the task completed and continues. When you create the task, you can specify the minimum number of replies.
- **Section header**—This task type performs no action. Section headers group related tasks within the playbook. You should give section headers descriptive names. By default, all playbooks start with a section header named *Playbook Triggered*, and you cannot rename this initial task. As a best practice, you should terminate each playbook with a section header named *Done*.

## Sub-Playbooks

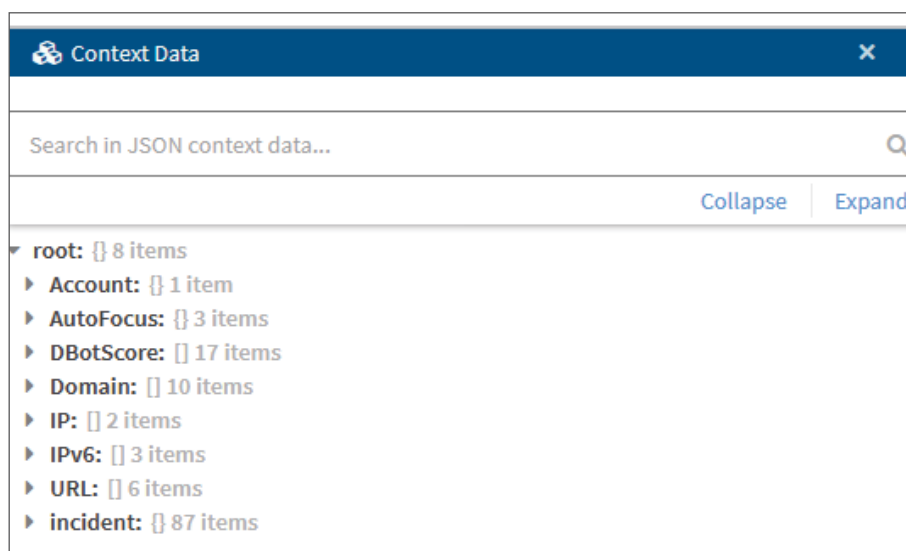
Like with tasks, you can also insert a playbook into the task flow of a parent playbook. You can create a playbook specifically for use as a *sub-playbook*. Using a sub-playbook, you can replace a set of related tasks with a single, reusable building block, which simplifies the parent playbook(s). If necessary, you can configure the sub-playbook to loop, repeating for each of a set of inputs. A sub-playbook maintains its own context data each time it runs. When you add a sub-playbook, you can choose whether the sub-playbook shares its context data globally with the parent playbook or keeps it private to the sub-playbook.

## Context Data

Cortex XSOAR extracts and collects data from various sources, such as playbook tasks, command results, and incoming events, and then stores the data within an incident as *context data*. Cortex XSOAR uses JSON format to store the context data. As discussed previously, JSON uses keys and values to store data objects. Because Cortex XSOAR uses a common format to store data, regardless of the original source, you can easily combine tasks that use different external systems such as Service Now for customer-support ticketing, Microsoft 365 for email, and the Palo Alto Networks portfolio for security.

Within a playbook, any task can access from any previously collected context data and use it as an input argument. Similarly, any task can store an output to the context data and make the output available to follow-on tasks. Because of the way that Cortex XSOAR stores outputs using the context data, you do not need to pass data directly from one task to another as you might within a software programming language.

Cortex XSOAR includes a built-in utility that you can use to manage and understand the context data for an incident. Although not explicitly named in the Cortex XSOAR user interface, you can access the Context Data explorer from any incident and use it to view the current context data. You can search within the context data by using JSON keys to find the assigned value, or alternatively, you can search on a specific value to find all JSON keys with the assigned value. The Context Data explorer is an essential tool with which you should become familiar. As you build custom playbooks and add tasks, you need to identify which data elements support the playbook logic—that is, which data elements you can use as task arguments.



Cortex XSOAR maintains a special set of context data objects that correspond to incident fields.

As discussed earlier in “Accessing Data,” Cortex XSOAR makes it easy to reference context data values in their exact stored format. However, in some cases, you might need to choose a specific value from a list or extract a substring of a string value. When you need to manipulate the context data, you use Cortex XSOAR filters and transformers. The filters provide a tool to extract certain values (example filter operators: **includes** or **doesn't include**), and the transformers provide a tool to convert values (example transformer operators: **StripChars** or **To lower case**). You can combine a filter and a transformer together to get the value that you require. If necessary, you can use multiple filters and multiple transformers, and Cortex XSOAR evaluates them in sequence.

For a full list of filter operators and transformer operators, see [Filters and Transformers](#) in the Cortex XSOAR Administrators Guide.

## Indicators

*Indicators* are artifacts associated with an investigation and are a critical component of the Cortex XSOAR system. Within Cortex XSOAR, the concept of an indicator is generic and Cortex XSOAR does not imply that an indicator is always malicious, as the terms “indicator of compromise” or “indicator of attack” might imply. Until Cortex XSOAR enriches an indicator, the reputation for that indicator is set to None. If you have configured Cortex XSOAR with an integration that provides threat reputation information, such as Palo Alto Networks AutoFocus™, then it uses the integration to enrich the indicator. If Cortex XSOAR successfully enriches the indicator, it sets the indicator reputation field to Good, Suspicious, or Bad.

When you have multiple threat reputation sources, Cortex XSOAR assigns an indicator’s reputation by using the reputation value returned by the source with the highest reliability score. In cases where multiple sources with the same reliability score return different reputation values for the indicator, Cortex XSOAR assigns the worst reported reputation value.

Cortex XSOAR stores indicators according to their type and includes a variety of built-in indicator types. If necessary, you can create custom indicator types.

Table 4 Cortex XSOAR indicator types

Category	Type name	Description
IP address	IP IPv6 CIDR IPv6CIDR	IPv4 address IPv6 address IPv4 address block IPv6 address block
File	File File MD5 File SHA-1 File SHA-256 ssdeep	Any file (includes all hash types) Any file (known by MD5 hash) Any file (known by SHA-1 hash) Any file (known by SHA-256 hash) Any file (known by ssdeep hash)
Domain	Domain DomainGlob	FQDN FQDN with wildcard
System	Account Host Registry Key	User account Hostname Windows registry key
URL	URL	Uniform resource locator
Email	Email	Email address or message ID
CVE	CVE	Common vulnerabilities and exposures record
STIX types	STIX Attack Pattern STIX Malware STIX Report STIX Threat Actor STIX Tool	Structured threat information expressions

Cortex XSOAR adds indicators to a global indicator repository through several methods:

- **External feeds and emails**—These integrations fetch indicators from a feed or emails with attached indicator lists.
- **Auto extract**—Cortex XSOAR automatically extracts indicator data from incoming events or playbook automation tasks.
- **Manual**—You can manually create an indicator of any type on the indicators page or using the Cortex XSOAR CLI.

Indicators are not specific to an incident and Cortex XSOAR does not store indicators within any incident's context data. If Cortex XSOAR observes an indicator as part of an incident investigation, it adds that indicator to the incident's indicator list. An indicator can be on the indicator list for multiple incidents. If you view the indicator details, you can see all of the related incidents. You can also directly investigate an indicator by creating an incident directly from the indicator.

After Cortex XSOAR adds an indicator, it remains until you delete it. By default, when you delete an indicator, Cortex XSOAR automatically adds it to the excluded indicator list, unless you specify otherwise. Cortex XSOAR ignores future processing of indicators after you add them to the excluded indicator list. This behavior helps to minimize duplicate incidents.



As an example, if a benign indicator, such as an internal domain or an IP address from the RFC-1918 range, has triggered an incident, you eventually close the incident after your investigation confirms that nothing malicious is associated with the indicator. To prevent Cortex XSOAR from creating a future incident based on the indicator, you delete and exclude the indicator. If you delete the indicator and do not add it to the excluded indicator list, Cortex XSOAR will create future incidents based on the indicator.

**Note**

If an indicator is on the excluded indicator list and you attempt to recreate the indicator, the create indicator request fails. You must first remove the indicator from the excluded indicator list.

## Integrations

To communicate with external systems, Cortex XSOAR uses integrations. At a basic level, an *integration* consists of software specifically developed to interact with an external system's API. You add new integrations to Cortex XSOAR by installing content packs, each of which includes one or more integrations. To enable communications with an external system, you configure and enable the appropriate integration between Cortex XSOAR and the external system, which can be a Palo Alto Networks product or a third-party product.

Cortex XSOAR supports integrations specific to these common categories (this list does not include less common categories):

- **Analytics and SIEM**—Security information and event-management tools or applications, which include Palo Alto Networks Cortex Data Lake, Splunk, and others.
- **Case management**—Customer service case-management and trouble-ticketing tools, which include ServiceNow, Jira, and others.
- **Data enrichment and threat intelligence**—Threat intelligence feeds, which include Palo Alto Networks AutoFocus, VirusTotal, and others.
- **Forensics and malware analysis**—Analysis tools for detecting and identifying malware and other threats. These tools include Palo Alto Networks WildFire®, FireEye, and others.
- **Messaging**—Email and collaboration tools, which include Microsoft 365, Slack, and others. Cortex XSOAR uses these types of integrations to send notifications.
- **Network security**—View and modify security policies on network security products and devices, which include Palo Alto Networks PAN-OS®, Cisco, and others.

Some integrations include a capability to import events as incidents. If you want to automatically create incidents on Cortex XSOAR, you must use an integration that includes this capability.

The Cortex XSOAR marketplace includes the Palo Alto Networks content packs and integrations listed in the following table.

Table 5 Palo Alto Networks integrations

Category	Content Pack	Integrations
Analytics and SIEM	Cortex Data Lake	Cortex Data Lake
Data enrichment and threat intelligence	Palo Alto Networks Threat Vault	Palo Alto Networks Threat Vault
Data enrichment and threat intelligence	AutoFocus	AutoFocus Daily Feed Palo Alto Networks AutoFocus v2
Data enrichment and threat intelligence	Expanse v2	Expanse Expander Feed Expanse v2
Data enrichment and threat intelligence	Palo Alto Networks PAN-OS EDL Service	Palo Alto Networks PAN-OS EDL Service
Data enrichment and threat intelligence	Palo Alto Networks Cortex XDR - Investigation and Response	Cortex XDR - IOC Palo Alto Networks Cortex XDR - Investigation and Response
Forensics and malware analysis	Palo Alto Networks WildFire	Palo Alto Networks WildFire v2
Network security	PAN-OS	Palo Alto Networks PAN-OS
Network security	Prisma™ Access	Prisma Access Prisma Access Egress IP feed
Network security	Palo Alto Networks IoT	Palo Alto Networks IoT
Uncategorized	Prisma Cloud Compute	Palo Alto Networks - Prisma Cloud Compute

## Scripts and Commands

Cortex XSOAR executes actions locally and on external systems by running either an automation script or automation command. You can use both in your playbook's standard tasks.

Automation scripts are included with a Cortex XSOAR Marketplace content pack. Although using an automation script sometimes requires configuring integration instances, you do not usually have to do so.

Automation commands are included with a Cortex XSOAR Marketplace content pack integration. You must configure the integration instance before you can use an automation command.

You can view a complete list of the available scripts and commands in the *Task Library*, which you can access from the Playbook screen when you edit a playbook.

## Automation Scripts

Some Cortex XSOAR content packs include automation scripts. After you install one of these content packs, Cortex XSOAR adds any included automation scripts to the Automation Library. Cortex XSOAR includes all scripts from the Automation Library in the Task Library that you use to build playbooks.

Cortex XSOAR also includes a variety of built-in automation scripts. You can see these in the Task Library, but Cortex XSOAR does not include them in the Automation Library.

You can then create playbook standard tasks that use the automation scripts. Some automation scripts do not require an associated integration instance (example: Base64Decode), and other automation scripts do require an associated integration instance (example: ADGetUser).



### Note

The Cortex XSOAR playbook task editor allows you to choose a script to use as an automation even if a required integration instance does not exist.

To ensure that Cortex XSOAR has the required integration configured, search the task library for the script that you wish to choose. The task library displays only tasks that have a required integration instance configured.

## Automation Commands

Most Cortex XSOAR integrations include automation commands. After you install a content pack and configure an instance of an integration from that content pack, Cortex XSOAR adds any included automation commands to the Task Library.

You can then create playbook standard tasks that use the automation commands. All automation commands require an associated integration instance (example: cdl-query-traffic-logs).



### Note

You can typically differentiate between an automation script and an automation command by the format of their names.

Automation script names almost always begin with capital letters and use CapWords (or CamelCase) formatting.

Example: **ExtractIndicatorsFromTextFile**

Automation command names use only lowercase letters and include dashes between words.

Example: **ews-get-searchable-mailboxes**

You can run any automation script and/or command from the Cortex XSOAR CLI.

A security analyst typically performs tasks using Cortex XSOAR playbooks. However, during playbook development, when you are adding a task with an unfamiliar automation script or command, you can use the Cortex XSOAR CLI to manually execute the script or command. To execute automation scripts or commands, begin your CLI command with the ! character.

## System Commands

As a Cortex XSOAR administrator, you typically perform system tasks within the web interface. Alternatively, you can also use the Cortex XSOAR CLI to execute commands that perform most of these tasks.

Administrative commands you can execute are known as *system commands*. These commands perform Cortex XSOAR administrative tasks or operations and are not specific to an integration. Typically, only an experienced Cortex XSOAR administrator with advanced skills uses system commands. To execute system commands, begin your CLI command with the / character.

## Playground

The Cortex XSOAR Playground is a non-production environment where you can safely develop and test automations and playbooks. To use the playground, you do not need to create an incident. You can choose any playbook and run it. The playground includes its own context data, which is entirely independent from the context data used for incidents. All tasks you run in the playground use this context data for inputs and outputs.



### Note

Although you can run any playbook in the playground, to successfully complete all tasks, some playbooks require valid incident data derived from actual events.

In these cases, you can partially test your prototype playbooks in the playground and then complete your verification by using test incidents.

If you want to use the Cortex XSOAR CLI to run automation scripts and commands outside of a playbook, the playground includes its own war room.

After you have finished developing a new playbook and tested it in the playground, you can start to use the playbook to investigate an incident.

## Reports

Cortex XSOAR includes two types of reports:

- **General report**—This includes a broad variety of built-in summaries from which you can choose, including daily incidents, last-24-hours incidents, last-7-days incidents, and last-30-days incidents. When you choose a general report, you can also differentiate between open and closed incidents. Depending on the report you choose, Cortex XSOAR uses PDF, Microsoft Word, or CSV format.
- **Incident report**—This is a built-in summary of incidence-specific information, including details such as the investigation timeline, indicators, war-room notes, and evidence. You can generate an incident report only from the incident, and you have the option to generate it manually or to generate it automatically by including an automation task within the incident playbook. For the incident reports, you can choose either PDF or Microsoft Word format.

If you need a custom report, you can create a completely new one, or you can copy a built-in general report and modify the copy. Cortex XSOAR provides a Widgets Library for the custom reports, and to add them, you can simply drag-and-drop widgets such as “Top Active Playbooks” to your report. After you have added widgets to your report, you can edit the arrangement, specify the time range for the report, and choose your preferred format.

If you want to create a report that summarizes information across multiple incidents, you can add custom widgets that include incident data or indicator data. The Cortex XSOAR reports present this information most clearly when you configure widgets to display data in table format. You can choose which columns to include in the tables. By default, these widgets include data from all incidents or all indicators from the time interval you specify. To make the widget more specific, you can add a data query, such as “type:Phishing”, to include only specific incident types.

## INCIDENT LIFECYCLE

### Planning and Customization

Cortex XSOAR supports a variety of built-in incident types, including phishing, malware, access, and many others. The incident type governs how Cortex XSOAR presents incident data and assigns the default playbook. You can customize the system if your organization wants to create additional incident types, expand on the information included within an incident, or adjust the way you want the information for an incident displayed.

At the most basic level, you can add new incident fields. Cortex XSOAR provides flexible options for choosing new field types. You can add simple types such as *Short text*, *Number*, or *URL*. If you need more complex types, you can add *Grid (table)*, *Attachments*, or *HTML*. For a full list, see [Incident Field Types](#) in the Cortex XSOAR Administrator’s Guide.

Each incident type includes a default layout, which displays the incident fields so that you can quickly see information that is relevant to the incident. Although you cannot modify the built-in system layouts, you can make copies of the built-in layouts and then modify the copies. To create new layouts or modify existing layouts, you use the incident layout builder. To add new sections, fields, buttons, or tabs, you drag-and-drop from the incident layout builder library. You can also move items within the layout, so that you can quickly see the information you need to manage the incident.

If you need to create a new incident type, you should start by identifying what fields are necessary to capture the relevant information. Next, you should create a custom layout that includes the fields that you require and that displays them effectively. When you then create the incident type, you assign the custom layout. You may assign a playbook to the incident type and choose whether to automatically run the playbook; however, for new incident types, you may not have a completed playbook, so you may wait until later to assign the playbook.

Each incident type includes a configuration option, which allows you to select how the system auto-extracts incident indicators when Cortex XSOAR creates an incident.

Table 6 Incident type auto-extract options

Mode	Description
Use system default (default option)	Use the global server configuration for auto extract. By default, the global server configuration for auto extract is inline mode.
None	Cortex XSOAR does not automatically extract indicators.
Inline	Cortex XSOAR automatically extracts and enriches indicators and then adds the findings to the context data. Playbooks do not run until auto extract is complete.
Out of band	Cortex XSOAR automatically extracts and enriches indicators and then adds the findings to the context data. Playbooks run independently from auto extract and may run concurrently. If a playbook requires enriched data within the incident, the data may not be available for immediate use.

**Note**

If you accept the default option of *Use system default*, this choice results in the selection of Inline mode.

If you are a novice user of Cortex XSOAR, inline mode is usually appropriate. However, Inline mode may slow down your system performance.

## Configuring Integration Instances

Each new integration instance you create and configure adds one or more automations to the Cortex XSOAR Automation Library. Some integrations also allow you to import external events as incidents, commonly referred to as *fetching* an incident. If you configure the integration instance to fetch incidents, each time Cortex XSOAR receives an event from an integration instance, Cortex XSOAR processes the event as a potential incident. By default, an integration instance polls its external system every minute.

You can configure more than one instance of an integration. Cortex XSOAR allows you to specify which instance of an integration you want to use when invoking an automation. If you do not specify the instance, the automation runs multiple times, once for each instance.

## Classifying and Mapping

As the first step of event processing, Cortex XSOAR determines which incident type to assign to a new incident. The Cortex XSOAR classifier uses fields parsed from a fetched event in order to assign the appropriate incident type to an incident. As an example, if you use an email integration and the fetched event includes a *received\_by* field and value of that field is *phishing@example.com*, the classifier sets the incident type to phishing. If the value of that field is any other value, the incident type remains unset.

If you want to create a new classifier, you must know the relevant event fields that are available to choose from. The easiest method to create a new classifier requires that you have an active integration instance that you have configured to fetch incidents. After receiving an event, Cortex XSOAR extracts the fields and values from the event and the Incidents Classification Editor displays them as a list. You can then choose the field that contains the desired classifier value and use it to assign the incident type.

Separate from the event classification, you can map the fields from your event data to the fields defined in your incident layout. As an example, your trouble-ticketing system uses “priority” and Cortex XSOAR uses “severity” as fields to track how critical a case or incident is. If the fields have different names on the different systems, you can use mapping to perform the translation. Although mapping is not a requirement, it is easier for you to select data to use in your playbooks if the data is in the incident fields.

If you want to create a new mapper, you must know the relevant event fields that are available to choose from. The easiest method to create a new mapper requires that you have an active integration instance that you have configured to fetch incidents. After receiving an event, Cortex XSOAR extracts the fields and values from the event and the Incidents Incoming Mapping Editor displays them as a list. For each Cortex XSOAR incident field that you want to map, you select it, and then you choose the corresponding event field.



#### Note

Cortex XSOAR always preserves the original event fields within the *incident.labels* context data regardless of whether or not the fields are mapped. You can access this context data within your playbook, but Palo Alto Networks recommends that you map the relevant event fields to incident fields.

## Applying Incident Pre-Processing Rules

Before Cortex XSOAR creates an incident, you can apply pre-processing rules to incoming events. You use these rules primarily to prevent duplicate incidents, but you can select from a variety of supported actions.

Table 7 Rule actions for pre-processing rules

Action	Description
Close	Create a new incident and then immediately close it.
Drop	Drop the incoming event. No incident is created.
Drop and update	Update the Dropped Duplicate Incidents section of an existing incident that you specify in the rule, and then drop the incoming event. No incident is created.
Link	Create an entry in the Linked Incidents section of an existing incident that you specify in the rule.
Link and close	Create an entry in the Linked Incidents section of an existing incident that you specify in the rule. Close the new incident.
Run a script	Select a script from the automation library to run on the incoming incident. You may only select from scripts with the tag <code>preProcessing</code> .



## Creating Incidents

You can create incidents manually or, if you have integrations configured, Cortex XSOAR can automatically create incidents for you. After you have an incident, you can start the process of investigating the details. Cortex XSOAR provides multiple work areas for each incident. Depending on the incident type and your role, you may use some or all work areas during your investigation.

The built-in layout for most incidents includes multiple tabs that correspond to the various work areas:

- **Case Info**—A summary of the incident, such as case details, work plan, evidence, and notes. You can add or remove these sections by customizing the layout.
- **Investigation**—An overview of the information collected about the investigation, such as indicators, email information, and screenshots. You can add or remove these sections by customizing the layout.
- **War Room**—A comprehensive history and collection of all investigation actions, artifacts, and collaboration. You can consider the war room to be a chronological journal of all activity related to the incident.
- **Work Plan**—A visual representation of the running playbook assigned to the incident.
- **Evidence Board**—The repository used to store all artifacts that you have designated as evidence. You can use these artifacts for current and future analysis, to reconstruct attack chains, and for root cause discovery.

## Running Playbooks

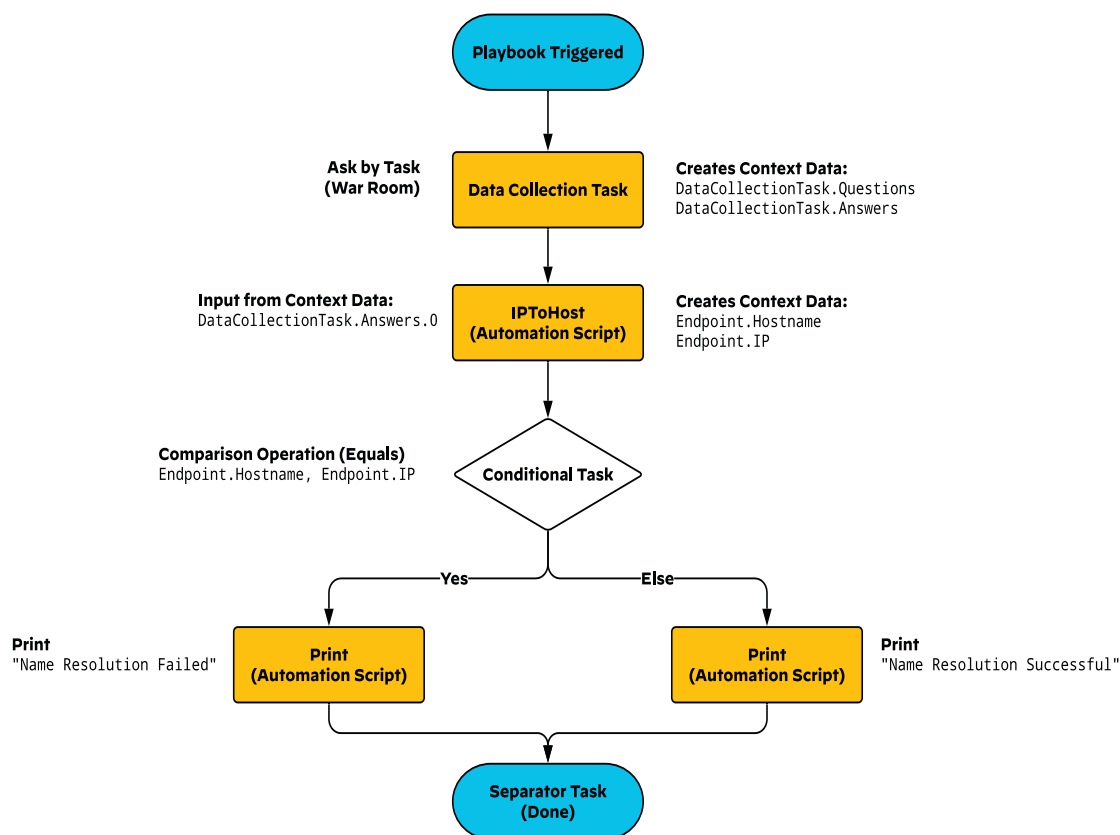
The investigation process for each incident follows a playbook, which you can automatically attach to the incident based on the incident type. You can also manually attach a playbook or change the attached playbook by editing the incident.

If you configure the incident type to run the playbook automatically, the playbook runs as soon as your integration instance fetches an event and creates the incident. Otherwise, you must navigate to the incidents screen and select the incident to start the investigation.

As a playbook runs, tasks can add information to the context data which then makes that information available for any tasks that follow. Subsequently, the next tasks can access any previously created context data within the incident and use as an input. Then, after execution, the task can continue to add new information to the context data for the incident.



Figure 2 Example playbook



You can view the progress of a running playbook in the incident work plan. The playbook executes sequentially and does not proceed to the next task until the previous task is complete. If the playbook forks into multiple paths, each path continues sequentially. The duration of task depends on several factors, which include the complexity of the task and the speed of the integration used by the task automation. If you are using a sub-playbook, the parent playbook does not proceed to the next task until all sub-playbook tasks are complete.

A running playbook can pause execution for several reasons:

- **A manual task is waiting for a response**—The playbook continues after you mark the task as completed.
- **A data collection task is waiting for a response**—The playbook continues after it receives a response.
- **A conditional task is waiting for a response**—The playbook continues after it receives a response.
- **A task encounters an error**—The playbook continues after you correct the error or after you manually mark the task as completed.
- **The playbook uses a polling task or polling sub-playbook**—The playbook logic requires that you wait until a specific condition exists before proceeding.

## Applying Incident Post-Processing Rules

After an incident investigation is complete and you are ready to close the incident, you can configure Cortex XSOAR to execute an automation script, such as **GenerateInvestigationSummaryReport**. Some other potential incident post-processing actions include closing a ticket on an external trouble-ticketing system or sending an email or slack notification.

You can only choose from automation scripts with the *post-processing* tag. To apply this tag to existing untagged automation scripts, you must first duplicate the automation script and then tag the duplicate. You can also develop and tag your own custom automation scripts.

# SecOps Automation Scenarios

---

Many organizations have a comprehensive incident response plan that they follow when under attack. Typically, these plans include guidance on how to identify different attack types and response scenarios specifically developed for each type.

As part of a transition to an effective SOAR model, you can develop automation scenarios that correspond to and enhance your manual responses. You can then craft a Cortex XSOAR playbook that maps the automation scenario to specific tasks. Depending on the attack type and automation scenario, your playbook can provide full or partial automation for your response.

## IMPLEMENTING AUTOMATION SCENARIOS

There are many ways to use the concepts discussed in the previous sections when creating and implementing automation scenarios using Cortex XSOAR. For each automation scenario, you need to understand the existing manual process from the incident response plan.

After you have installed Cortex XSOAR and performed any required post-installation tasks (as described in [SecOps Automation and Response Deployment Guide for Cortex XSOAR](#)), you can transition to a workflow based on automation.

First, identify the attack type from your response plan. Check if any system playbooks for the attack type are either already in the Cortex XSOAR playbook library or are available in the Cortex XSOAR Marketplace. If a playbook is available in the marketplace, then download and install the content pack that includes the playbook.

Next, identify all the external systems your response plan uses. Check if integrations for the external systems are available in the Cortex XSOAR marketplace. If Cortex XSOAR has an integration for an external system, you must first download and install the content pack that includes the integration from the Cortex XSOAR marketplace before you can integrate the external system with Cortex XSOAR. After you install the content pack for an integration, you create an integration instance for Cortex XSOAR that includes the specific details used to communicate with the external system.

For each external system, you may need to perform configuration tasks which enable Cortex XSOAR to communicate and interact with the system. These tasks may be as simple as adding a user account, assigning roles or privileges, or creating an API key. You may have to perform more complex tasks, depending on the level of automation you require.

Finally, you need to build the Cortex XSOAR playbook for the automation scenario. The playbook should mirror the manual process, but you should introduce automation where possible. You can eliminate manual data entry by using Cortex XSOAR data collection tasks. You can replace manual tasks where a security analyst accesses an external system with automation tasks where Cortex XSOAR accesses the external systems directly. If your manual process includes a decision tree, you use conditional tasks to implement the decision tree within your Cortex XSOAR playbook.

After your playbook is complete, you can assign it as the default playbook for an incident type, and optionally set the playbook to run automatically when Cortex XSOAR creates an incident. If preferred, you can always manually choose and assign a specific playbook to an incident.

In many cases, you can use existing Cortex XSOAR system playbooks. You can copy a system playbook and then customize the copy to better meet your needs. Or in a new playbook, you can add an unmodified system playbook as a sub-playbook task.

To build and practice your skills with using Cortex XSOAR, Palo Alto Networks has created [a deployment guide](#) and [a set of operations guides](#) specifically tailored for some example automation scenarios.

## EXAMPLE AUTOMATION SCENARIOS

This guide introduces two specific automation scenarios. These scenarios provide illustrative examples that you can use while becoming familiar with Cortex XSOAR. Each scenario includes different methods for incident creation, different external systems, and different automation tasks.

- **URL Filtering Exception Automation Scenario**—This scenario uses Cortex XSOAR to automate the processing of requests to create exceptions to your organization's URL filtering policy.
- **Automated Phishing Investigation and Response Automation Scenario**—This scenario uses Cortex XSOAR to investigate a suspected phishing attack, collect and verify the attack details, and present the details to the SecOps analyst. This scenario also includes optional processes that you can use to remove the phishing email from the recipients' mailboxes and to notify users who accessed a malicious URL.

For the specific, step-by-step configuration details you need for building and running the actual playbook for each automation scenario, see the associated SecOps Automation and Response Operations Guides. The playbooks in these operations guides assume certain required conditions are in place and do not include extensive error checking, which you would include in a published playbook. You should not consider these playbooks a substitute for the system playbooks you can download from the Cortex XSOAR Marketplace. You should use them to learn the capabilities of Cortex XSOAR and gain experience.

### URL Filtering Exception Automation Scenario

In this scenario, your organization has implemented a URL filtering policy that blocks access to dangerous or undesirable URLs based on their categorization. Users occasionally attempt access to a blocked URL to which they have a legitimate need for access. In these cases, you must configure an exception.

In this scenario, a user opens a ServiceNow ticket requesting an exception for the blocked URL. You had configured Cortex XSOAR to integrate with Service Now and periodically poll ServiceNow for new tickets. After the user opens the new ticket, Cortex XSOAR retrieves it as an event and creates an incident. As part of the incident response, Cortex XSOAR automatically executes a playbook to determine if the URL is properly formatted and then, if the format is valid, performs a reputation analysis for the URL.

Cortex XSOAR starts the analysis by creating an indicator for the URL and then retrieves reputation data for the URL from Palo Alto Networks AutoFocus. To provide additional information for the SecOps analyst, Cortex XSOAR retrieves the URL's current filtering category by querying an active Palo Alto Networks next-generation firewall and retrieves a rasterized image of the URL's web page.

When the automated analysis is complete, the assigned SecOps analyst uses Cortex XSOAR to review the incident details and then to approve or deny the exception request. If the SecOps analyst approves the request, Cortex XSOAR automatically adds the URL to an external dynamic list (EDL). Across your organization, the Palo Alto Networks next-generation firewalls that perform URL filtering periodically retrieve the EDL and then merge the URL filtering exceptions to the active URL filtering policy.

For more information about this automation scenario, see [SecOps Automation and Response–Cortex XSOAR URL Filtering Policy: Operations Guide](#)

## Automated Phishing Investigation and Response Automation Scenario

In this scenario, your SecOps team has set up a phishing mailbox on the email system, which they monitor using Cortex XSOAR, and they request that users forward all suspected phishing messages to that mailbox. A user has forwarded one such email.

You had configured Cortex XSOAR to integrate with the email system and periodically check for emails to the monitored mailbox. When the new email arrives, Cortex XSOAR retrieves it as an event and creates an incident. As part of the incident response, Cortex XSOAR automatically executes a playbook to analyze the email and, optionally, to automatically respond if the email contains phishing or malware content.

Cortex XSOAR starts the basic analysis by retrieving the original message that the end-user forwarded. In addition to extracting the email headers, which include domain and IP address indicators, the analysis extracts URL indicators. To determine if unknown URL indicators are malicious, Cortex XSOAR submits them to WildFire.

When the analysis completes, before taking any further action, the assigned SecOps analyst uses Cortex XSOAR in order to review the attack information that Cortex XSOAR has automatically gathered.

## Automatic Response Options

If you want Cortex XSOAR to take actions based on the results, this automation scenario includes some advanced options. The first option is to search the email system for all recipients of the original message and, if the analysis determines that the message contained a malicious phishing URL, remove the message from all mailboxes systemwide. The second option assumes that users at your organization connect to the network through Prisma Access. Cortex XSOAR searches the Prisma Access logs stored in Cortex Data Lake to determine if any users tried to access the malicious URL in the original message. If the logs confirm that users did try and connect, then you can notify the users. Although it is beyond the scope of this guide, you could use this information to create new incidents to interact with users who might need further attention, such as attending a training session focused on phishing awareness or initiating a forensic examination of their computer system.

For more information about this automation scenario, see [SecOps Automation and Response –Cortex XSOAR Phishing Investigation: Operations Guide](#).

# Summary

---

This guide has discussed many of the cybersecurity challenges faced by organizations.

These challenges include:

- Ever-increasing number of attacks.
- Workforce transitions driven by remote teleworkers.
- Workload migrations to public cloud.
- Too many security tools and overwhelmed security staff.

In addition to providing a comprehensive and proven security portfolio, Palo Alto Networks specifically combats these challenges with orchestration and automation, driven by Cortex XSOAR.

Cortex XSOAR provides automated responses to security attacks by using playbooks and integration with cybersecurity tools from Palo Alto Networks and third-party vendors. Cortex XSOAR helps security teams to reduce mean time to response, create consistent incident management processes, and increase team productivity.

Included as a tutorial, this guide provides an introductory discussion of data management concepts that you need to be successful with Cortex XSOAR.

This guide also introduces the critical Cortex XSOAR concepts that you need before you start using Cortex XSOAR and includes an in-depth discussion of the incident lifecycle that Cortex XSOAR uses.

This guide has prepared you to take the next steps with Cortex XSOAR: first deploying Cortex XSOAR and then building and running playbooks for SecOps automation scenarios.



## HEADQUARTERS

Palo Alto Networks  
3000 Tannery Way  
Santa Clara, CA 95054, USA

<http://www.paloaltonetworks.com>

Phone: +1 (408) 753-4000

Sales: +1 (866) 320-4788

Fax: +1 (408) 753-4001

[info@paloaltonetworks.com](mailto:info@paloaltonetworks.com)

© 2021 Palo Alto Networks, Inc. Palo Alto Networks is a registered trademark of Palo Alto Networks. A list of our trademarks can be found at <http://www.paloaltonetworks.com/company/trademarks.html>. All other marks mentioned herein may be trademarks of their respective companies. Palo Alto Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.



You can use the [feedback form](#) to send comments about this guide.