Configuring Multi-Tenancy for Unified Threat Defense

Multi-tenancy for Unified Threat Defense provides Snort IPS and Web Filtering for multiple users. You can define policies for one or more tenants in a single Cisco CSR 1000v instance. Each policy can have a threat inspection profile and a web filtering profile. The following sections describe how to configure multi-tenancy for Unified Threat Defense. Many of the commands used in these configuration steps are similar to those used in configuring single-tenancy—see: Snort IPS andWeb Filtering .

Information About Multi-Tenancy for Unified Threat Defense

Multi-tenancy for Snort IPS and Web Filtering allows you to define policies for one or more tenants, in one Cisco CSR 1000v instance. This feature was introduced in Cisco IOS XE Everest 16.6.1.

Each tenant is a VPN routing and forwarding instance with one or more VPN routing and forwarding tables (VRFs). A Unified Threat Defense (UTD) policy is associated with a threat inspection profile and web filtering profile. Multiple tenants can share a UTD policy.

The system logs include the name of the VRF which allows you to produce statistics per-tenant.

The CLI commands used in multi-tenancy mode are similar to those used in single-tenancy mode (see Snort IPS and Web Filtering). In multi-tenancy, you enter a sub-mode `utd engine standard multi-tenancy` and configure UTD policies, web filtering and threat-inspection profiles. After exiting the `utd engine standard multi-tenancy` sub-mode, the UTD policies are applied.

The benefits of web filtering and threat inspection (Snort IPS/IDS) are explained in the following sections:

- Benefits of Web Filtering
- Overview of Snort Virtual Service Interfaces, on page 3

Web Filtering Overview

Web Filtering allows you to provide controlled access to the internet by configuring URL-based policies and filters. Web Filtering helps to control access to websites by blocking malicious or unwanted websites and therefore making the network more secure. You can blocked list individual URLs or domain names and configure allowed list policies for the same. You can also make provision to allow or block a URL based on reputation or category.

Snort IPS Overview

The Snort IPS feature enables Intrusion Prevention System (IPS) or Intrusion Detection System (IDS) for branch offices on Cisco 4000 Series Integrated Services Routers and Cisco Cloud Services Router 1000v Series. This feature uses the Snort engine to provide IPS and IDS functionalities.

Snort is an open source network IPS that performs real-time traffic analysis and generates alerts when threats are detected on IP networks. It can also perform protocol analysis, content searching or matching, and detect a variety of attacks and probes, such as buffer overflows, stealth port scans, and so on. The Snort engine runs as a virtual container service on Cisco 4000 Series Integrated Services Routers and Cisco Cloud Services Router 1000v Series.

The Snort IPS feature works in the network intrusion detection and prevention mode that provides IPS or IDS functionalities. In the network intrusion detection and prevention mode, Snort performs the following actions:

- Monitors network traffic and analyzes against a defined rule set.
- Performs attack classification.
- Invokes actions against matched rules.

Based on your requirements, you can enable Snort either in IPS or IDS mode. In IDS mode, Snort inspects the traffic and reports alerts, but does not take any action to prevent attacks. In IPS mode, in addition to intrusion detection, actions are taken to prevent attacks.

The Snort IPS monitors the traffic and reports events to an external log server or the IOS syslog. Enabling logging to the IOS syslog may impact performance due to the potential volume of log messages. External third-party monitoring tools, which supports Snort logs, can be used for log collection and analysis.

Snort IPS Solution

The Snort IPS solution consists of the following entities:

- Snort sensor—Monitors the traffic to detect anomalies based on the configured security policies (that includes signatures, statistics, protocol analysis, and so on) and sends alert messages to the Alert/Reporting server. The Snort sensor is deployed as a virtual container service on the router.
- Signature store—Hosts the Cisco Signature packages that are updated periodically. These signature packages are downloaded to Snort sensors either periodically or on demand. Validated signature packages are posted to Cisco.com. Based on the configuration, signature packages can be downloaded from Cisco.com or a local server.

  The following domains are accessed by the router in the process of downloading the signature package from cisco.com:

  - api.cisco.com

- apx.cisco.com
- cloudsso.cisco.com
- cloudsso-test.cisco.com
- cloudsso-test3.cisco.com
- cloudsso-test4.cisco.com
- cloudsso-test5.cisco.com
- cloudsso-test6.cisco.com
- cloudsso.cisco.com
- download-ssc.cisco.com
- dl.cisco.com
- resolver1.opendns.com
- resolver2.opendns.com

Note If you are downloading signature packages from a local server to hold the signature packages, only HTTP is supported.

Signature packages must be manually downloaded from Cisco.com to the local server by using Cisco.com credentials before the Snort sensor can retrieve them.

The Snort container performs a domain-name lookup (on the DNS server(s) configured on the router) to resolve the location for automatic signature updates from Cisco.com or on the local server, if the URL is not specified as the IP address.

- Alert/Reporting server—Receives alert events from the Snort sensor. Alert events generated by the Snort sensor can either be sent to the IOS syslog or an external syslog server or to both IOS syslog and external syslog server. No external log servers are bundled with the Snort IPS solution.
- Management—Manages the Snort IPS solution. Management is configured using the IOS CLI. Snort Sensor cannot be accessed directly, and all configuration can only be done using the IOS CLI.

Overview of Snort Virtual Service Interfaces

The Snort sensor runs as a service on routers. Service containers use virtualization technology to provide a hosting environment on Cisco devices for applications.

You can enable Snort traffic inspection either on a per interface basis or globally on all supported interfaces. The traffic to be inspected is diverted to the Snort sensor and injected back. In Intrusion Detection System (IDS), identified threats are reported as log events and allowed. However, in Intrusion Prevention System (IPS), action is taken to prevent attacks along with log events.

The Snort sensor requires two VirtualPortGroup interfaces. The first VirtualPortGroup interface is used for management traffic and the second for data traffic between the forwarding plane and the Snort virtual container

service. Guest IP addresses must be configured for these VirtualPortGroup interfaces. The IP subnet assigned to the management VirtualPortGroup interface should be able to communicate with the Signature server and Alert/Reporting server.

The IP subnet of the second VirtualPortGroup interface must not be routable on the customer network because the traffic on this interface is internal to the router. Exposing the internal subnet to the outside world is a security risk. We recommend the use of 192.0.2.0/30 IP address range for the second VirtualPortGroup subnet. The use of 192.0.2.0/24 subnet is defined in RFC 3330.

You can assign the Snort virtual container service IP address on the same management network as the router on which the virtual service is running. This configuration helps if the syslog or update server is on the management network and is not accessible by any other interfaces

Restrictions for Configuring Multi-Tenancy for Unified Threat Defense

-
- Domain-based filtering is not supported.
- Up to 25 tenants are supported on each Cisco CSR 1000v instance.
- A maximum of 25 policies are supported.
- A maximum of 50,000 concurrent sessions are supported on a Cisco CSR 1000v.
-
- The blocked list/allowed list rules support only a regular expression (regex) pattern. Currently, 64 patterns are supported for each blocked list/allowed list rule. However, each tenant can have multiple rules.
- Local block server does not support serving HTTPS block page.When the URL filter tries to inject block page or redirect message, it does not support HTTPS traffic.
- When there is a username and password in the URL, URL filter does not remove them from the URL before matching the blocked list/allowed list pattern. However, the category/reputation lookup does not have this limitation and removes the username and password from the URL before lookup.
- HTTPS inspection is limited. Web filtering uses server certificate to obtain the URL/domain information. It is not possible to inspect the full URL path.
- UTD does not inter-operate with WCCP, and NBAR under inter-VRF scenario.
- The Snort IPS command `threat inspection profile` *`profile-name`* uses an alphanumeric profile-name, not an ID (number).

How to Configure Multi-Tenancy for Unified Threat Defense

To deploy multi-tenancy for Unified Threat Defense on supported devices, perform the following tasks:

Before you begin

Provision the device upon which you wish to install web filtering and threat inspection for multi-tenancy. This feature is currently only supported on the Cisco CSR 1000v.

Obtain the license. UTD is available only for routers running security packages and you will require a security license to enable the service. Contact Cisco Support to obtain a security license.

SUMMARY STEPS

1. Install and activate the virtual-service: Installing the UTD OVA File for Multi-Tenancy, on page 5.
2. Configure the VirtualPortGroup interfaces and the virtual-service: How to Configure VirtualPortGroup Interfaces and Virtual Service for Multi-Tenancy, on page 6.
3. Configure the VRFs: How to Configure VRFs for Multi-Tenancy, on page 9.
4. Configure threat inspection and web filtering for multi-tenancy: How to Configure Multi-Tenancy Web Filtering and Threat Inspection, on page 10

DETAILED STEPS

Step 1 Install and activate the virtual-service: Installing the UTD OVA File for Multi-Tenancy, on page 5.

Step 2 Configure the VirtualPortGroup interfaces and the virtual-service: How to Configure VirtualPortGroup Interfaces and Virtual Service for Multi-Tenancy, on page 6.

Step 3 Configure the VRFs: How to Configure VRFs for Multi-Tenancy, on page 9.

Step 4 Configure threat inspection and web filtering for multi-tenancy: How to Configure Multi-Tenancy Web Filtering and Threat Inspection, on page 10

Installing the UTD OVA File for Multi-Tenancy

The virtual-service OVA file is an Open Virtualization Archive file that contains a compressed, installable version of a virtual machine. You must download this OVA file to the router and then install the virtual-service. The virtual-service OVA file is not bundled with Cisco IOS XE release images that are installed on the router. OVA files may be available pre-installed in the router's flash memory.

For installing the OVA file, you must use a Cisco IOS XE image with a security license. During installation, the security license is checked.

Example of installing the virtual service:

```
Device> enable
Device# virtual-service install name utd package
bootflash:utdsnort.1.0.4_SV2983_XE_16_6.20170623_174453_RELEASE.ova
Device# show virtual-service list

Name Status    Package Name
------------------------------------------------------------------------------
utd Activated utdsnort.1.0.4_SV2983_XE_16_6.20170
```

Example of upgrading the virtual service:

```
Device> enable
Device# virtual-service upgrade name utd package
bootflash:utdsnort.1.0.4_SV2983_XE_16_6.20170623_174453_RELEASE.ova
Device# show virtual-service list
```

```
Name Status   Package Name
------------------------------------------------------------------------------
utd Activated utdsnort.1.0.4_SV2983_XE_16_6.20170
```

Example of uninstalling the virtual service:

```
Device> enable
Device# virtual-service uninstall name utd
Device# show virtual-service list

Virtual Service List:
```

How to Configure VirtualPortGroup Interfaces and Virtual Service for Multi-Tenancy

As shown in this procedure, for multi-tenancy you must configure two VirtualPortGroup interfaces and guest IP addresses for both interfaces.

Note The VirtualPortGroup interface for data traffic must use a private or nonroutable IP address. We recommend the use of 192.0.2.0/30 IP address range for this interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface VirtualPortGroup** *interface-number*
4. **ip address** *ip-address mask*
5. **exit**
6. **interface VirtualPortGroup** *interface-number*
7. **ip address** *ip-address mask*
8. **exit**
9. **virtual-service** *name*
10. **profile multi-tenancy**
11. **vnic gateway VirtualPortGroup** *interface-number*
12. **guest ip address** *ip-address*
13. **exit**
14. **vnic gateway VirtualPortGroup** *interface-number*
15. **guest ip address** *ip-address*
16. **exit**
17. **activate**
18. **end**
19. **show virtual-service list**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface VirtualPortGroup** *interface-number* **Example:** `Device(config)# interface VirtualPortGroup 0`	Enters interface configuration mode and configures a VirtualPortGroup interface. This interface is used for management traffic when the management interface GigabitEthernet0 is not used.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.1.1.1 255.255.255.252`	Sets a primary IP address for an interface. This interface needs to be routable to the signature update server and external log server.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**interface VirtualPortGroup** *interface-number* **Example:** `Device(config)# interface VirtualPortGroup 1`	Configures an interface and enters interface configuration mode. Configure a VirtualPortGroup interface. This interface is used for data traffic.
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.0.2.1 255.255.255.252`	Sets a primary IP address for an interface. This IP address should not be routable to the outside network. The IP address is assigned from the recommended 192.0.2.0/30 subnet.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**virtual-service** *name* **Example:** `Device(config)# virtual-service utd`	Configures a virtual container service and enters virtual service configuration mode. The *name* argument is the logical name that is used to identify the virtual container service.
Step 10	**profile multi-tenancy** **Example:** `Device(config-virt-serv)#profile multi-tenancy`	Configures a resource profile. For multi-tenancy mode (Cisco CSR 1000v only), this `profile multi-tenancy` command must be configured.

	Command or Action	Purpose
Step 11	**vnic gateway VirtualPortGroup** *interface-number* **Example:** `Device(config-virt-serv)# vnic gateway VirtualPortGroup 0`	Enters the virtual-service virtual network interface card (vNIC) configuration mode. Creates a vNIC gateway interface for the virtual container service and maps the vNIC gateway interface to the virtual port group interface. This is the interface that was configured in Step 3.
Step 12	**guest ip address** *ip-address* **Example:** `Device(config-virt-serv-vnic)# guest ip address 10.1.1.2`	Configures a guest vNIC address for the vNIC gateway interface.
Step 13	**exit** **Example:** `Device(config-virt-serv-vnic)# exit`	Exits virtual-service vNIC configuration mode and returns to virtual service configuration mode.
Step 14	**vnic gateway VirtualPortGroup** *interface-number* **Example:** `Device(config-virt-serv)# vnic gateway VirtualPortGroup 1`	Enters virtual-service vNIC configuration mode. Configures a vNIC gateway interface for the virtual container service and maps the interface to the virtual port group. The interface (*interface-number*) configured in Step 6) is used by the Snort engine for monitoring user traffic.
Step 15	**guest ip address** *ip-address* **Example:** `Device(config-virt-serv-vnic)# guest ip address 192.0.2.2`	Configures a guest vNIC address for the vNIC gateway interface.
Step 16	**exit** **Example:** `Device(config-virt-serv-vnic)# exit`	Exits virtual-service vNIC configuration mode and returns to virtual service configuration mode.
Step 17	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates an application installed in a virtual container service.
Step 18	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual service configuration mode and returns to privileged EXEC mode.

Step 19	**show virtual-service list**
	Example:
	`Device# show virtual-service list`
	
	`Virtual Service List:`
	
	`Name  Status  Package Name`
	`----------`
	`utd  Activated  utdsnort.1.0.4_SV2983_XE_16_6.20170`	

How to Configure VRFs for Multi-Tenancy

This procedure describes the typical steps required for configuring VRFs for the tenants, which are later used in: How to Configure Multi-Tenancy Web Filtering and Threat Inspection, on page 10.

Note For inter-VRF traffic, if the traffic flowing between two VRFs has ingress and egress interfaces configured for UTD, rules are applied to decide which VRF represents the session. The UTD policy for the selected VRF then applies to all packets in the inter-VRF traffic.

SUMMARY STEPS

1. **vrf definition** *vrf-name*
2. **rd** *route-distinguisher*
3. **address-family ipv4**
4. **exit address-family**
5. Repeat steps 1 to 4 for each VRF.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition 100`	Defines the name of the VRF and enters VRF configuration mode.
Step 2	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Creates the routing and forwarding tables and associates the *route-distinguisher* with the VRF instance named *vrf-name*. The router uses the route-distinguisher to identify the VRF to which a packet belongs. The route-distinguisher is of one of the following two types: • Autonomous System-related. An AS number xxx and an arbitrary number y—xxx:y • IP address-related. An IP address A.B.C.D and an arbitrary number y—A.B.C.D:y
Step 3	**address-family ipv4** **Example:** `Device(config-vrf)# address-family ipv4`	Enters address family configuration mode for configuring routing sessions using the IP Version 4 address.
Step 4	**exit address-family** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode.
Step 5	Repeat steps 1 to 4 for each VRF.	

How to Configure Multi-Tenancy Web Filtering and Threat Inspection

To configure threat inspection (IPS/IDS) and web filtering for multi-tenancy (multiple tenants/VRFs), perform the following steps.

In this procedure, the definition of blocked list and allowed lists are shown in the initial steps 1 to 5. The main configuration steps (in UTD standard engine configuration mode for multi-tenancy) are shown in step 6 onwards.

Note For details about threat inspection and web filtering for single-tenancy, see Snort IPS and Web Filtering.

Before you begin

Remove any existing single-tenancy UTD configuration, using the `no utd engine standard` command.

You must have previously configured a VRF for each tenant—see How to Configure VRFs for Multi-Tenancy, on page 9.

Procedure

	Command or Action	Purpose
Step 1	**parameter-map type regex** *blacklist-name* **Example:** `Device(config)# parameter-map type regex urlf-blacklist1`	Defines a blocked list parameter map, which is used later in step 17.

Step 2	**pattern** *URL-name*
	Example:
	`Device(config-profile)# pattern www\.cnn\.com`
	`Device(config-profile)# pattern www\.msnbc\.com`
		Defines the URL to be on the blocked list. Note that the periods within *URL-name* must be preceded by an escape "\" character. Repeat this step to configure multiple URLs to be on the blocked list.

Step 3	**parameter-map type regex** *whitelist-name* **Example:** `Device(config-profile)# parameter-map type regex urlf-whitelist1`	Defines an allowed list parameter map, which is used later in step 20.
Step 4	**pattern** *URL-name* **Example:** `Device(config-profile)# pattern www\.nfl\.com`	Defines the URL(s) to be on the allowed list. Note that, for URLs on the blocked list, periods within *URL-name* must be preceded by an escape "\" character. Repeat this step to configure multiple URLs to be on the allowed list.
Step 5	**exit** **Example:** `Device(config-profile)# exit`	

	Command or Action	Purpose
Step 6	**utd multi-tenancy** **Example:** `Device(config)# utd multi-tenancy`	This command acts a switch, in preparation for the following `utd engine standard multi-tenancy` command.
Step 7	**utd engine standard multi-tenancy** **Example:** `Device(config)# utd engine standard multi-tenancy`	Enters UTD standard engine configuration mode for multi-tenancy. **Note** Later. after you exit the UTD standard engine configuration mode in step 50, the policy configurations are applied.
Step 8	**web-filter sourcedb** *sourcedb-number* **Example:** `Device(config)# web-filter sourcedb 1`	Configures a web filtering sourcedb profile—*sourcedb-number*, which is numeric. This is used later in step 29.
Step 9	**logging level** {**alerts** | **critical** | **debugging** | **emergencies** | **errors** | **informational** | **notifications** | **warnings**} **Example:** `Device(config)# logging level errors`	Sets the level of system messages that are reported upon for web filtering events. Messages of the specified level and lower are reported. (Each level has a numeric value as shown in the table below.) ***Table 1: System Message Severity Levels*** **Level** / **Description** 0 – emergencies / System unusable 1 – alerts / Immediate action needed 2 – critical / Critical condition 3 – errors / Error condition 4 – warnings / Warning condition 5 – notifications / Normal but significant condition 6 – informational / Informational messages only 7 – debugging / Appears during debugging only
Step 10	**web-filter block local-server profile** *profile-id* **Example:** `Device(config-utd-multi-tenancy)# web-filter block local-server profile 1` The content text is displayed by the local server.	Configures the a local block server profile for web filtering. The range of values for *profile-id* is 1–255. See Configure URL-based Web Filtering with a Local Block Server. **Note** When configuring commands for multi-tenancy, compared to single-tenancy, you do not use the initial `utd` keyword.

	Command or Action	Purpose
Step 11	**block-page-interface loopback** *id* **Example:** `Device(config-utd-mt-webf-blk-srvr)# block-page-interface loopback 110`	Associates a loopback interface with this profile. The IP address of this loopback interface is then used as the IP address of the block local-server.
Step 12	**content text** *display-text* **Example:** `Device(config-utd-mt-webf-blk-srvr)# content text "Blocked by Web-Filter"`	Specifies the warning text that appears after a blocked page is accessed.
Step 13	**http-ports** *port-number* **Example:** `Device(config-utd-mt-webf-blk-srvr)# http-ports 80`	The http-ports value is a string of ports separated by commas. The nginx HTTP server listens to these ports.

Step 14	**web-filter block page profile** *profile-name*
	Example:
	`Device(config-utd-multi-tenancy)# web-filter block page profile 1`
	`Device(config-utd-mt-webf-block-urc)# text "this page is blocked"`
		See Configure URL-based Web Filtering with an Inline Block Page, except that the command used here for multi-tenancy does not use the `utd` keyword which is used for single-tenancy.).

Step 15	**web-filter url profile** *web-filter-profile-id*
	Example:
	`Device(config-utd-multi-tenancy)# web-filter url profile 1`
	`Device(config-utd-mt-webfltr-url)#`
		Specifies a URL profile for web filtering—*web-filter-profile-id*. Values: 1–255. After this command, you can configure alerts for blocked lists, allowed lists, and categories. For further information, see: Configure URL-based Web Filtering with an Inline Block Page.
		Note When configuring commands for multi-tenancy, compared to single-tenancy, you do not use an initial `utd` keyword.

Step 16	**blacklist** **Example:** `Device(config-utd-mt-webfltr-url)# blacklist`	Enters web filtering blocked list configuration mode.
Step 17	**parameter-map regex** *blacklist-name* **Example:** `Device(config-utd-mt-webf-url-bl)# parameter-map regex urlf-blacklist1`	Specifies a parameter-map regular expression using the blocked list that was defined earlier in step 1.

Step 18	**exit**
	Example:
	`Device(config-utd-mt-webf-url-bl)# exit`
	`Device(config-utd-mt-webfltr-url)#`
		Exits web filtering blocked list configuration mode.

	Command or Action	Purpose

Step 19	**whitelist**
	Example:
	`Device(config-utd-mt-webfltr-url)# whitelist`
	`Device(config-utd-mt-webf-url-wl)#`
		Enters web filtering allowed list configuration mode.

Step 20	**parameter-map regex** *whitelist-name* **Example:** `Device(config-utd-mt-webf-url-wl)# parameter-map regex urlf-list1`	Specifies a parameter-map regular expression using the allowed list that was defined earlier in step 3.

Step 21	**exit**
	Example:
	`Device(config-utd-mt-webf-url-wl)# exit`
	`Device(config-utd-mt-webfltr-url)#`
		Exits web filtering allowed list configuration mode.

Step 22	**exit**
	Example:
	`Device(config-utd-mt-webfltr-url)# exit`
	`Device(config-utd-multi-tenancy)#`
		Exits web filtering URL profile mode.

Step 23	**utd global** **Example:** `Device(config-utd-multi-tenancy)# utd global`	The commands entered for `utd global` apply to all tenants or policies e.g the commands shown below: `logging host syslog` and `threat inspection` for this Cisco CSR 1000v instance.

Step 24	**logging** {**host** *hostname* | **syslog**}
	Example:
	In this example, alerts are logged to a designated host log file.
	`Device(config-utd-mt-utd-global)# logging host systemlog1`
	Example:
	In this example, alerts are logged to IOS syslogs.
	`Device(config-utd-mt-utd-global)# logging syslog`
		The `logging` command specifies either a host name or IOS syslog, to which syslog messages are sent.

Step 25	**threat inspection** **Example:** `Device(config-utd-mt-utd-global)# threat inspection`	Enters global threat inspection mode.
Step 26	**signature update server** {**cisco** | **url** *url* } [**username** *username* [**password** *password*]] **Example:** `Device(config-utd-mt-utd-global-threat)# signature update server cisco username abcd password cisco123`	Configures the signature update server parameters. You must specify the signature update parameters with the server details. If you use www.cisco.com for signature updates, you must provide the username and password. If you use a local server for signature updates, based on the server settings you can provide the username and password.

	Command or Action	Purpose
		The router must be able to resolve the domain name by being connected to the internet.
Step 27	**signature update occur-at** {**daily** | **monthly** *day-of-month* | **weekly** *day-of-week*} *hour minute* **Example:** `Device(config-utd-mt-utd-global-threat)# signature update occur-at daily 0 0`	Configures the signature update interval parameters. This configuration will trigger the signature update to occur at midnight.
Step 28	**web-filter** **Example:** `Device(config-utd-mt-utd-global-threat)# web-filter`	This command, used in combination with the following `sourcedb` command, specifies the URL source database for web filtering.
Step 29	**sourcedb** *sourcedb-number* **Example:** `Device(config-utd-mt-utd-global-threat)# sourcedb 1`	Assigns a web filtering source database. Only one source database can be active.
Step 30	**exit** **Example:** `Device(config-utd-mt-utd-global-threat)# exit`	Exits threat inspection configuration mode.
Step 31	**exit** **Example:** `Device(config-utd-mt-global)# exit`	Exits global update configuration mode.
Step 32	**threat-inspection whitelist profile** *policy-name* **Example:** `Device(config-utd-multi-tenancy)# threat-inspection whitelist profile wh101`	Associates an allowed list profile with the policy currently being configured. A similar command is used in single-tenancy, but with a `utd` keyword.
Step 33	**signature id** *id* **Example:** `Device(config-utd-mt-list)# signature id 101`	Specify the ID *id* that you have previously identified as a threat; for example, after observing the ID in an alert log file. Repeat this command for multiple signature IDs.
Step 34	**exit** **Example:** `Device(config-utd-mt-whitelist)# exit`	Exits an allowed list configuration mode.
Step 35	**threat-inspection profile** *profile-name* **Example:** `Device(config-utd-multi-tenancy)# threat-inspection profile 101`	Configures a threat inspection profile, which can be reused by multiple tenants. You can configure multiple threat-inspection profiles. Within a profile you can configure multiple allowed lists. `profile-name` is alphanumeric.

	Command or Action	Purpose
Step 36	**threat** {**detection** | **protection** } **Example:** `Device(config-utd-mt-threat)# threat protection`	Specifies Intrusion Detection System (IDS) or Intrusion Prevention System (IPS) as the operating mode for the Snort engine. The default is **threat detection**
Step 37	**policy** {**balanced** | **connectivity** | **security**} **Example:** `Device(config-utd-mt-threat)# policy security`	Configures the security policy for the Snort engine. • The default security policy type is **balanced**.
Step 38	**logging level**{**alert** | **crit** | **debug** | **emerg** | **err** | **info** | **notice** | **warning**}	Provides logs in one of these categories: • alert—provides alert level logs (severity=2) • crit—critical level logs (severity=3) • debug—all logs (severity=8) • emerg—emergency level logs (severity=1) • err—error level logs (severity=4) Default. • info—info level logs (severity=7) • notice—notice level logs (severity=6) • warning—warning level logs (severity=5)
Step 39	**whitelist profile** *profile-name* **Example:** `Device(config-utd-mt-threat)# whitelist profile wh101`	You can also specify allowed list profiles in a profile only for allowed lists in another place—the `threat-inspection whitelist profile` command above. (Optional) Enables allowed lists under the UTD engine.
Step 40	**exit** **Example:** `Device(config-utd-mt-threat)# exit`	Exits threat inspection mode.
Step 41	Repeat steps 35 to 40 to add additional threat-inspection profiles.	
Step 42	**policy** *policy-name* **Example:** `Device(config-utd-multi-tenancy)# policy pol101`	Defines the policy that will be associated with multiple tenants. A threat detection (IPS) and web filtering profile are added to the policy.
Step 43	**vrf** [*vrf-name* | global] **Example:** This example shows the configuration of two tenants (VRFs) and two policies. `Device(config-utd-mt-policy)# vrf vrf101`	Repeat the `vrf vrf-name` command for each of the VRFs (tenants) that will use the UTD policy. These VRFs previously defined, see: How to Configure VRFs for Multi-Tenancy, on page 9. Alternatively use `vrf global` to associate with the global (default) VRF and enables VRF under the interface.

	Command or Action	Purpose
Step 44	**all-interfaces** **Example:** `Device(config-utd-mt-policy)# all-interfaces`	(Optional) Associates all interfaces under the VRF with the policy.
Step 45	**threat-inspection profile** *profile-name* **Example:** `Device(config-utd-mt-policy)# threat-inspection profile 101`	(Optional) Associates the policy with a previously defined threat inspection profile, see Step 35.
Step 46	**web-filter url profile** *web-filter-profile-id* **Example:** `Device(config-utd-mt-policy)# web-filter url profile 1`	(Optional) Associates the policy with a previously defined web filtering profile, see step 15.
Step 47	**fail close** **Example:** `Device(config-utd-mt-policy)# fail close`	(Optional) Drops IPS/IDS packets on engine failure. Default is `fail open`.
Step 48	**exit**	Exits from policy configuration mode.
Step 49	Repeat steps 42 to 48 for each policy	
Step 50	**exit** **Example:** `Device(config-utd-multi-tenancy)# exit`	Exits the `utd engine standard multi-tenancy` mode. The policy configurations are applied, which may take a few minutes. During this time, further `utd engine standard multi-tenancy` configuration mode commands cannot be entered.

Step 51	**exit**
	Example:
	`Device(config)# exit`
	`Device#`	

Step 52	**show logging**
	Example:
	`Device(config)# show logging`
	`..UTD MT configuration download has started`
	`..UTD MT configuration download has completed`	

Step 53	**interface** *sub-interface* **Example:** `Device(config)# interface GigabitEthernet4.101`	Specify a sub-interface to be used for the tenant (VRF).
Step 54	**encapsulation dot1Q** *vlan-id* **Example:** `Device(config-if)# encapsulation dot1Q 101`	Applies a VLAN ID to the sub-interface.

	Command or Action	Purpose
Step 55	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding vrf101`	Associates a VRF instance with the sub-interface.
Step 56	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 111.0.0.1 255.255.255.0`	Specifies the sub-interface IP address of the VRF.
Step 57	**ip route** *ip-address subnet-mask sub-interface* **Example:** In this example, the VRF's subnet GigabitEthernet4.101 is linked to the global routing table using the static IP address 111.0.0.0 255.255.255.0. `Device(config-if)# ip route 111.0.0.0 255.255.255.0 GigabitEthernet4.101`	(Optional) This `ip route` command and the `ip route vrf` command in the following step are optional—you can use these steps if you want to configure route leaking using a static route between the VRF and the global routing table. This configures a static route to the VRF subnet from the VRF interface, so that the VRF subnet is accessible from the global routing table. For further information on configuring route leaking, see Route Leaking in MPLS/VPN Networks.
Step 58	**ip route vrf** *vrf-name ip-address subnet-mask* **global** **Example:** `Device(config-if)# ip route vrf vrf101 0.0.0.0 0.0.0.0 5.2.1.1 global`	(Optional) This step and the previous step are optional——you can use these steps if you want to configure route leaking using a static route between the VRF and the global routing table. For further information on configuring route leaking, see Route Leaking in MPLS/VPN Networks. Specifies the static VRF default route to the global routing table.
Step 59	**utd enable**	(Optional) Enables UTD on an interface. You can use this command if the `all-interfaces` command was not configured (in step 44).
Step 60	To configure a sub-interface for each tenant (VRF), repeat steps 53 to 59.	
Step 61	**exit**	Exits interface configuration mode.

The profiles for web filtering and threat inspection (IPS) have now been applied.

Example Configuration—Multi-Tenancy for Unified Threat Defense

This example shows a typical running configuration after configuring Multi-Tenancy for UTD for two tenants.

Note The following example mentions parameter maps `urlf-blacklist1` and `urlf-whitelist1`. The configuration of these parameter maps is not shown in the example. For further information on blocked list and approved list parameter-maps, see Configure URL-based Web Filtering with an Inline Block Page.

```
utd multi-tenancy
utd engine standard multi-tenancy
 web-filter block page profile 1
  text "This page is blocked"
 web-filter block page profile 2
  text "This page is blocked"
 web-filter url profile 1
  alert all
  blacklist
   parameter-map regex urlf-blacklist1
  whitelist
   parameter-map regex urlf-whitelist1
  categories block
   social-network
   sports
  block page-profile 1
  log level error
 web-filter url profile 2
  alert all
  blacklist
   parameter-map regex urlf-blacklist2
  categories block
   shopping
   news-and-media
   sports
   real-estate
   motor-vehicles
  block page-profile 2
  log level error
  reputation
   block-threshold low-risk
 web-filter sourcedb 1
  logging level error
 threat-inspection whitelist profile wh101
  signature id 101
 threat-inspection profile 101
  threat protection
  policy security
  logging level debug
  whitelist profile wh101
 threat-inspection profile 102
  threat detection
  policy security
  logging level debug
 utd global
  logging host 172.27.58.211
  logging host 172.27.58.212
  logging host 172.27.56.97
  threat-inspection
   signature update server cisco username abc password ]RDCe[B\^KFI_LgQgCFeBEKWP^SWZMZMb]KKAAB

   signature update occur-at daily 0 0
  web-filter
   sourcedb 1
 policy pol102
  vrf vrf102
  all-interfaces
  threat-inspection profile 102
  web-filter url profile 2
 policy pol101
  vrf vrf101
  all-interfaces
  threat-inspection profile 101
```

```
web-filter url profile 1
fail close
```

Verifying Unified Threat Defense Engine Standard Configuration

Use the following commands to verify your configuration.

SUMMARY STEPS

1. **enable**
2. **show utd multi-tenancy**
3. **show utd engine standard global**
4. **show utd engine standard status**
5. **show utd engine standard statistics**
6. **show utd engine standard statistics daq** [**dp** | **cp**]
7. **show utd engine standard statistics url-filtering** [**engine** | *no*]
8. **show utd engine standard statistics url-filtering vrf name** *vrf-name*
9. **show utd engine standard statistics internal**
10. **show utd engine standard logging event**
11. **show logging** | **include** CONFIG_DOWNLOAD
12. **show utd threat-inspection whitelist** [**profile** *profile-name*]
13. **show utd threat-inspection profile** *profile-name*
14. **show utd** [**policy** *profile-name*]
15. **show utd web-filter url** [**profile** *profile-name*]
16. **show utd web-filter block local-server** [**profile** *profile-name*]
17. **show utd web-filter sourcedb** [**profile** *profile-name*]
18. **show utd engine standard statistics daq dp** [**engine** *engine-num*] [**vrf** [**name** *vrf-name* | **global**]]
19. **show utd engine standard config threat-inspection whitelist** [**profile** *profile-name*]
20. **show utd engine standard config web-filter url profile** *profile-name*
21. **show utd engine standard config** [**vrf name** *vrf-name*]
22. **show utd engine standard config threat-inspection profile** *profile-name*
23. **show utd engine standard threat-inspection signature update status**
24. **show platform software qfp active feature utd config** [**vrf**[{**id** *vrf-id* | **name** *vrf-name* | **global** }]
25. **show platform software utd interfaces**
26. **show platform hardware qfp active feature utd config** [**vrf** {**id** *vrf-id* | **name** *vrf-name* | **global** }]
27. **show platform hardware qfp active feature utd stats** [**clear** | **divert** | **drop** | **general** | **summary**] [**vrf** {**id** *vrf-id* | **name** *vrf-name* | **global** }] [all] [**verbose**]
28. **show platform hardware qfp active feature utd stats summary** [**vrf name** *vrf-name* | **all**]
29. **show platform hardware qfp active feature utd stats drop all**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device# enable
```

Enables privileged EXEC mode. Enter your password if prompted.

Step 2 **show utd multi-tenancy**

Displays the current status of multi-tenancy.

Example:

```
Device# show utd multi-tenancy
Multitenancy is enabled
```

Step 3 **show utd engine standard global**

Displays the global settings for utd engine standard.

Example:

```
Device# show utd engine standard global
UTD Engine Standard Global: enabled
Threat-inspection: enabled
Web-filter: enabled
Logging:
```

Step 4 **show utd engine standard status**

Verify that the status of the UTD engine is Green.

Example:

```
Device# show utd eng standard status
Engine version       : 1.0.2_SV2983_XE_16_8

Profile              : Multi-tenancy
System memory        :
              Usage  : 3.50 %
              Status : Green
Number of engines    : 1

Engine        Running    CFT flows  Health     Reason
=========================================================
Engine(#1):   Yes        0          Green      None
=========================================================

Overall system status: Green

Signature update status:
========================
Current signature package version: 29.0.c
Last update status: Failed
Last successful update time: None
Last failed update time: Thu Jan 11 13:34:36 2018 PST
Last failed update reason: [Errno 113] No route to host
Next update scheduled at: None
Current status: Idle
```

Step 5 **show utd engine standard statistics**

Example:

```
Device# show utd engine standard statistics
*************Engine #1*************
===============================================================================
Memory usage summary:
```

```
Total non-mmapped bytes (arena): 80125952
Bytes in mapped regions (hblkhd): 359546880
Total allocated space (uordblks): 68314032
Total free space (fordblks): 11811920
Topmost releasable block (keepcost): 112
================================================================================
Packet I/O Totals:
Received: 49088
Analyzed: 49088 (100.000%)
Dropped: 0 ( 0.000%)
Filtered: 0 ( 0.000%)
Outstanding: 0 ( 0.000%)
Injected: 640
================================================================================
Breakdown by protocol (includes rebuilt packets):
Eth: 49394 (100.000%)

<output removed for brevity>

Total: 49394
================================================================================
Action Stats:
Alerts: 65 ( 0.132%)
Logged: 65 ( 0.132%)
Passed: 0 ( 0.000%)
```

Step 6 **show utd engine standard statistics daq [dp | cp]**

Show Snort DAQ statistics.

Example:

```
Device# show utd engine standard statistics daq dp
IOS-XE DAQ Counters(Engine #1):
--------------------------------
Frames received 654101
Bytes received 549106120
RX frames released 654101
Packets after vPath decap 654101
Bytes after vPath decap 516510928
Packets before vPath encap 651686
Bytes before vPath encap 514800669
Frames transmitted 651686
Bytes transmitted 544447557

<output removed for brevity>
```

Example:

```
Device# show utd engine standard statistics daq cp
IOS-XE DAQ CP Counters(Engine #1):
-----------------------------------
Packets received :16353210
Bytes received :1112018252
Packets transmitted :16353210
Bytes transmitted :1700733776
Memory allocation :16353212
Memory free :16353210
CFT API error :0
VPL API error :0
Internal error :0
External error :0
Memory error :0
Timer error :0
```

```
RX ring full 0
CFT full 0
sPath lib flow handle exhausted 0
Memory status changed to yellow :1
Memory status changed to red :0
Process restart notifications :0
```

Step 7 **show utd engine standard statistics url-filtering** [**engine** | *no*]

Gives the URL statistics for all the tenants combined: the number of hits for sites on the blocked list, number of hits for sites on the allowed list, and the number of sites that are blocked by category block and reputation block.

Example:

```
Device# show utd engine standard statistics url-filtering
UTM Preprocessor Statistics
---------------------------
URL Filter Requests Sent:               377226166       379846771       381117940
URL Filter Response Received:           377009606       379622845       380892658
Blacklist Hit Count:                    0               0               0
Whitelist Hit Count:                    0               0               0

Reputation Lookup Count:                376859139       379458008       380706804
Reputation Action Block:                0               0               0
Reputation Action Pass:                 307             280             102
Reputation Action Default Pass:         376858832       379457728       380706702
Reputation Score None:                  376858832       379457728       380706702
Reputation Score Out of Range:          0               0               0

Category Lookup Count:                  376859139       379458008       380706804
Category Action Block:                  0               0               0
Category Action Pass:                   307             280             102
Category Action Default Pass:           376858832       379457728       380706702
Category None:                          376858832       379457728       380706702


Device# show utd engine standard statistics url-filtering engine1
UTM Preprocessor Statistics
-----------------------------------------------------
URL Filter Requests Sent:               377226166
URL Filter Response Received:           377009606
Blacklist Hit Count:                    0
Whitelist Hit Count:                    0

Reputation Lookup Count:                376859139
Reputation Action Block:                0
Reputation Action Pass:                 307
Reputation Action Default Pass:         376858832
Reputation Score None:                  376858832
Reputation Score Out of Range:          0

Category Lookup Count:                  376859139
Category Action Block:                  0
Category Action Pass:                   307
Category Action Default Pass:           376858832
Category None:                          376858832
```

Step 8 **show utd engine standard statistics url-filtering vrf name** *vrf-name*

Gives per-tenant URL statistics by using the additional parameters—**vrf name** *vrf-name* .

Example:

```
Device# show utd engine standard statistics url-filtering vrf name vrf101
UTM Preprocessor Statistics
---------------------------
URL Filter Requests Sent: 764
URL Filter Response Received: 764
Blacklist Hit Count: 3
Whitelist Hit Count: 44

Reputation Lookup Count: 764
Reputation Action Block: 0
Reputation Action Pass: 58
Reputation Action Default Pass: 706
Reputation Score None: 706
Reputation Score Out of Range: 0

Category Lookup Count: 764
Category Action Block: 5
Category Action Pass: 53
Category Action Default Pass: 706
Category None: 706
```

Step 9 **show utd engine standard statistics internal**

Example:

```
Device# show utd engine standard statistics internal
*************Engine #1*************
===============================================================================
Memory usage summary:
Total non-mmapped bytes (arena): 80125952
Bytes in mapped regions (hblkhd): 359546880
Total allocated space (uordblks): 68314032
Total free space (fordblks): 11811920
Topmost releasable block (keepcost): 112
===============================================================================
Packet I/O Totals:
Received: 49088
Analyzed: 49088 (100.000%)
Dropped: 0 ( 0.000%)
Filtered: 0 ( 0.000%)
Outstanding: 0 ( 0.000%)
Injected: 640
===============================================================================
Breakdown by protocol (includes rebuilt packets):
Eth: 49394 (100.000%)
VLAN: 49394 (100.000%)
IP4: 49394 (100.000%)
Frag: 0 ( 0.000%)
ICMP: 5 ( 0.010%)
UDP: 2195 ( 4.444%)
TCP: 47194 ( 95.546%)

<output removed for brevity>
```

Step 10 **show utd engine standard logging event**

Displays the logs which contains alerts and URLs that are either on the blocked or allowed list per VRF.

Example:

```
Device# show utd engine standard logging event

2017/08/04-16:01:49.205959 UTC [**] [Instance_ID: 1] [**] Drop [**]
```

```
UTD WebFilter Category/Reputation [**] [URL: www.cricinfo.com] ** [Category: Sports]
** [Reputation: 96] [VRF: vrf101] {TCP} 23.72.180.26:80 -> 111.0.0.254:53509
2017/08/04-16:02:12.253330 UTC [**] [Instance_ID: 1] [**] Pass [**]
 UTD WebFilter Whitelist [**] [URL: www.espn.go.com/m]
[VRF: vrf101] {TCP} 111.0.0.254:53511 -> 199.181.133.61:80
```

Step 11 **show logging** | **include** CONFIG_DOWNLOAD

Example:

```
show# logging | include CONFIG_DOWNLOAD
Aug 23 11:34:21.250 PDT: %IOSXE_UTD-4-MT_CONFIG_DOWNLOAD: UTD MT configuration download has started
Aug 23 11:54:18.496 PDT: %IOSXE_UTD-4-MT_CONFIG_DOWNLOAD: UTD MT configuration download has completed
```

Step 12 **show utd threat-inspection whitelist** [**profile** *profile-name*]

Displays all allowed list profiles or a specific allowed list profile.

Example:

```
Device# show utd threat-inspection whitelist
Whitelist Profile: wh101
Signature ID: 101
```

Example:

```
Device# show utd threat-inspection whitelist profile wh101
Whitelist Profile: wh101
Signature ID: 101
```

Step 13 **show utd threat-inspection profile** *profile-name*

Displays the details of a threat-inspection profile specified by the *profile-name*.

Example:

```
Device# show utd threat-inspection profile 101
Threat-inspection Profile: 101
Operational Mode: Intrusion Protection
Operational Policy: Security
Logging Level: debug
Whitelist Profile: wh101
```

Step 14 **show utd** [**policy** *profile-name*]

Displays all UTD policies or a specific UTD policy.

Example:

```
Device# show utd policy pol101
Policy name: pol101
VRF name: vrf101, VRF ID: 1
Global Inspection (across above VRFs): Enabled
Threat-inspection profile: 101
Web-filter URL profile: 1
Fail Policy: Fail-open
```

Step 15 **show utd web-filter url** [**profile** *profile-name*]

Displays all URL profiles or a specific profile.

Example:

```
Device# show utd web-filter url profile 1
URL Profile: 1
Alert: all
Blacklist Parameter Map Regex: urlf-blacklist1
Whitelist Parameter Map Regex: urlf-whitelist1
Block Categories:
dating
sports
Block Page Profile 1
Log level error
reputation block-threshold high-risk
```

Step 16 **show utd web-filter block local-server** [**profile** *profile-name*]

Displays all block page profiles or a specific block page profile.

Example:

```
Device# show utd web-filter block local-server profile 2
Block Local Server Profile: 2
Content text: "Blocked by Web-Filter"
HTTP ports: 80
```

Step 17 **show utd web-filter sourcedb** [**profile** *profile-name*]

Displays all sourcedb profiles or a specific sourcedb profile.

Example:

```
Device# show utd web-filter sourcedb
SourceDB Profile: 1
database update server interval hour 0 minute 0
Fail open
Log level: error
Proxy host port 0

SourceDB Profile: 2
database update server interval hour 0 minute 0
Fail open
Log level: error
Proxy host port 0
```

Example:

```
Device# show utd web-filter sourcedb profile 1
SourceDB Profile: 1
database update server interval hour 0 minute 0
Fail open
Log level: error
Proxy host port 0
```

Step 18 **show utd engine standard statistics daq dp** [**engine** *engine-num*] [**vrf** [**name** *vrf-name* | **global**]]

Displays serviceplane data acquistion (DAQ) statistics for all VRFs or a specific VRF.

Example:

The following example shows the serviceplane data acquisition statistics for VRF vrf101.

```
Device# show utd engine standard statistics daq dp vrf name vrf101
IOS-XE DAQ Counters(Engine #1):
--------------------------------
Frames received 374509
Bytes received 303136342
RX frames released 374509
Packets after vPath decap 374509
Bytes after vPath decap 284405526
Packets before vPath encap 372883
Bytes before vPath encap 283234522
Frames transmitted 372883
Bytes transmitted 300202270

Memory allocation 781856
Memory free 749636
Memory free via timer 29420
Merged packet buffer allocation 0
Merged packet buffer free 0

VPL buffer allocation 0
VPL buffer free 0
VPL buffer expand 0
VPL buffer merge 0
VPL buffer split 0
VPL packet incomplete 0

VPL API error 0
CFT API error 0
Internal error 52
External error 0
Memory error 0
Timer error 0

Kernel frames received 373590
Kernel frames dropped 0

FO cached via timer 0
Cached fo used 0
Cached fo freed 0
FO not found 0
CFT full packets 0
```

Step 19 **show utd engine standard config threat-inspection whitelist** [**profile** *profile-name*]

Displays the details of a threat-inspection allowed list profile stored in a container.

Example:

```
Device# show utd engine standard config threat-inspection whitelist
UTD Engine Standard Configuration:

UTD threat-inspection whitelist profile table entries:
Whitelist profile: wh101
Entries: 1
```

Step 20 **show utd engine standard config web-filter url profile** *profile-name*

Displays the details of the web-filter profile stored in the container.

Example:

```
Device# show utd engine standard config web-filter url profile 1
UTD Engine Standard Configuration:
```

```
UTD web-filter profile table entries
Web-filter URL profile: 1
Whitelist:
www.espn.com
www.nbcsports.com
www.nfl.com
Blacklist:
www.cnn.com
Categories Action: Block
Categories:
Social Network
Sports
Block Profile: 1
Redirect URL: http://172.27.56.97/vrf101.html
Reputation Block Threshold: High risk
Alerts Enabled: Whitelist, Blacklist, Categories, Reputation
Debug level: Error
Conditional debug level: Error
```

Step 21 **show utd engine standard config** [**vrf name** *vrf-name*]

Displays the details of the UTD policy, threat-inspection profile and web-filter profile associated with a particular VRF.

Example:

```
Device# show utd engine standard config vrf name vrf101
UTD Engine Standard Configuration:

UTD VRF table entries:
VRF: vrf101 (1)
Policy: pol101
Threat Profile: 101
Webfilter Profile: 1
```

Step 22 **show utd engine standard config threat-inspection profile** *profile-name*

Displays the details of a specific threat-inspection profile.

Example:

```
Device# show utd engine standard config threat-inspection profile 101
UTD Engine Standard Configuration:

UTD threat-inspection profile table entries:
Threat profile: 101
Mode: Intrusion Prevention
Policy: Security
Logging level: Debug
Whitelist profile: wh101

Description:
Displays the details of a threat-inspection profile stored in the container.
```

Step 23 **show utd engine standard threat-inspection signature update status**

Shows the output of the current signature package version, previous signature package version, and last status update.

Example:

```
Device# show utd engine standard threat-inspection signature update status
Current signature package version: 29.0.c
Current signature package name: default
Previous signature package version: None
```

```
---------------------------------------
Last update status: Failed
---------------------------------------
Last successful update time: None
Last successful update method: None
Last successful update server: None
Last successful update speed: None
---------------------------------------
Last failed update time: Thu Jan 11 13:34:36 2018 PST
Last failed update method: Manual
Last failed update server: http://172.27.57.252/UTD-STD-SIGNATURE-2983-1-S.pkg
Last failed update reason: [Errno 113] No route to host
---------------------------------------
Last attempted update time: Thu Jan 11 13:34:36 2018 PST
Last attempted update method: Manual
Last attempted update server: http://172.27.57.252/UTD-STD-SIGNATURE-2983-1-S.pkg
---------------------------------------
Total num of updates successful: 0
Num of attempts successful: 0
Num of attempts failed: 1
Total num of attempts: 1
---------------------------------------
Next update scheduled at: None
---------------------------------------
Current status: Idle
```

Step 24 **show platform software qfp active feature utd config** [**vrf**[{**id** *vrf-id* | **name** *vrf-name* | **global** }]

Shows the service node statistics. The VRF information can only be shown in the case of multi-tenancy. Displays the data plane UTD configuration. In the following example the security context information is highlighted.

Example:

```
Device# Global configuration
  NAT64: disabled
  SN threads: 12
  CFT inst_id 0 feat id 0 fo id 0 chunk id 4
  Context Id: 0, Name: Base Security Ctx
   Ctx Flags: (0xf0000)
        Engine: Standard
        SN Redirect Mode : Fail-close, Divert
        Threat-inspection: Enabled, Mode: IPS
        Domain Filtering : Not Enabled
        URL Filtering    : Not Enabled
   SN Health: Green
```

Step 25 **show platform software utd interfaces**

Example:

```
Device# show platform software utd interfaces

UTD interfaces
All dataplane interfaces
```

Step 26 **show platform hardware qfp active feature utd config** [**vrf** {**id** *vrf-id* | **name** *vrf-name* | **global** }]

Show UTD datapath configuration and status.

Example:

```
Device# show platform hardware qfp active feature utd config vrf name vrf101
Global configuration
  NAT64: disabled
  Drop pkts: disabled
  Multi-tenancy: enabled
  Data plane initialized: yes
  SN threads: 12
  CFT inst_id 0 feat id 1 fo id 1 chunk id 8
  SN Health: Green
```

Step 27 **show platform hardware qfp active feature utd stats** [**clear** | **divert** | **drop** | **general** | **summary**] [**vrf** {**id** *vrf-id* | **name** *vrf-name* | **global** }] [all] [**verbose**]

Displays dataplane UTD statistics, including counts of zeros

clear—Clear Statistics

divert—Display AppNav Redirect Statistics

drop—Display Drop Statistics

general—Display General Statistics

summary—Display Summary Statistics

verbose—Display Verbose Statistics

vrf Display per VRF stats—The VRF information can only be entered if multi-tenancy is enabled.

id—display stats associated with the VRF id

name—display stats associated with the VRF with the provided name

global—display the stats associated with the global VRF (i.e vrf-id 0)

Example:

```
Device# show platform hardware qfp active feature utd stats

Summary Statistics:
TCP Connections Created 29893
UDP Connections Created 24402
ICMP Connections Created 796
Pkts dropped pkt 258
byt 66365
Pkts entered policy feature pkt 715602
byt 562095214
Pkts entered divert feature pkt 662014
byt 516226302
Pkts slow path pkt 55091
byt 4347864
Pkts Diverted pkt 662014
byt 516226302
Pkts Re-injected pkt 659094
byt 514305557

Would-Drop Statistics:

Service Node flagged flow for dropping 258

General Statistics:
Non Diverted Pkts to/from divert interface 1022186
Inspection skipped - UTD policy not applicable 1081563
```

```
<output removed for brevity>
```

Example:

Step 28 **show platform hardware qfp active feature utd stats summary** [**vrf name** *vrf-name* | **all**]

Displays information about all VRFs or a specific VRF, taken from the summary option of the **show platform hardware qfp active feature utd stats** command.

Example:

```
Device# show platform hardware qfp active feature utd stats vrf name vrf101
Security Context: Id:1 Name: 1 : vrf101

Summary Statistics:
TCP Connections Created 18428
UDP Connections Created 13737
ICMP Connections Created 503
Pkts dropped pkt 258
byt 66365
Pkts entered policy feature pkt 407148
byt 296496913
Pkts entered divert feature pkt 383176
byt 283158966
Pkts slow path pkt 32668
byt 2571632
Pkts Diverted pkt 383176
byt 283158966
Pkts Re-injected pkt 381016
byt 281761395


<output removed for brevity>
```

Step 29 **show platform hardware qfp active feature utd stats drop all**

Displays information from all the VRFs taken from the drop option of the **show platform** command.

Example:

```
Device# show platform hardware qfp active feature utd stats drop all

Would-Drop Statistics:

No diversion interface                                          0
No egress interface                                             0
Inspection service down                                         0
Could not find divert interface                                 0
Could not find divert fib                                       0
UTD FIB did not contain oce_chain                               0
Invalid IP version                                              0
IPS not supported                                               0
Re-inject Error                                                 0
Service Node flagged flow for dropping                       1225
Could not attach feature object                                 0
Could not allocate feature object                               0
Error getting feature object                                    0
Policy: could not create connection                             0
NAT64 Interface Look up Failed                                  0
Decaps: VPATH connection establishment error                    0
Decaps: VPATH could not find flow, no tuple                     0
Decaps: VPATH notification event error                          0
Decaps: Could not delete flow                                   0
Decaps: VPATH connection classification error                   0
```

```
Encaps: Error retrieving feature object                                  0
Encaps: Flow not classified                                              0
Encaps: VPATH connection specification error                             0
Encaps: VPATH First packet meta-data failed                              0
Encaps: VPATH No memory for meta-data                                    0
Encaps: VPATH Could not add TLV                                          0
Encaps: VPATH Could not fit TLV into memory                              0
Service Node Divert Failed                                               0
No feature object                                                        0
Service Node not healthy                                               123
Could not allocate VRF meta-data                                         0
Could not allocate debug meta-data                                       0
Packet was virtually fragmented (VFR)                                    0
IPv6 Fragment                                                            0
IPv4 Fragment                                                            0
```

Troubleshooting Multi-Tenancy for Unified Threat Defense

Traffic is not Diverted

Problem Traffic is not diverted.

Possible Cause Vitual-service may not be activated.

Solution Check whether the virtual-service is activated by using the **show virtual-service list** command. The following is sample output from the command:

```
Device# show virtual-service list

Virtual Service List:


Name Status Package Name
--------------------------------------------------------------------------------
snort Activated utdsnort.1_0_1_SV2982_XE_16_3.20160701_131509.ova
```

Possible Cause Unified threat defense (UTD) may not be enabled for specified interface or interfaces.

Solution Use the **show platform software utd global** command to verify if UTD is enabled for the interface:

```
Device# show platform software utd global

UTD Global state
Engine               : Standard
Global Inspection    : Disabled
Operational Mode     : Intrusion Prevention
Fail Policy          : Fail-open
Container techonlogy : LXC
Redirect interface   : VirtualPortGroup1
UTD interfaces
GigabitEthernet0/0/0
```

Possible Cause The service node may not be working properly.

Solution Use the **show platform hardware qfp active feature utd config** command to verify if the health of the service node is green:

```
Device# show platform hardware qfp active feature utd config

Global configuration
NAT64: disabled
SN threads: 12
CFT inst_id 0 feat id 0 fo id 0 chunk id 4
Context Id: 0, Name: Base Security Ctx
Ctx Flags: (0x60000)
Engine: Standard
SN Redirect Mode : Fail-open, Divert
Threat-inspection: Enabled, Mode: IDS
Domain Filtering : Not Enabled
URL Filtering : Not Enabled
SN Health: Green
```

Solution Alternatively, in the case of multi-tenancy, you can use the **show platform hardware qfp active feature utd config vrf name** *vrf-name* command to verify if the health of the service node, for a specific VRF, is green:

```
Device# show platform hardware qfp active feature utd config vrf name vrf102
Global configuration
  NAT64: disabled
  Drop pkts: disabled
  Multi-tenancy: enabled
  Data plane initialized: yes
  SN threads: 12
  CFT inst_id 0 feat id 0 fo id 0 chunk id 4
  SN Health: Green
```

Possible Cause The Snort process may not be activated.

Solution Use the **show virtual-service detail** command to verify if the Snort process is up and running:

```
Device# show virtual-service detail

Virtual service UTDIPS detail
  State                : Activated
  Owner                : IOSd
  Package information
    Name               : utdsnort.1_0_1_SV2982_XE_16_3.20160701_131509.ova
    Path               : bootflash:/utdsnort.1_0_1_SV2982_XE_16_3.20160701_131509.ova
    Application
      Name             : UTD-Snort-Feature
      Installed version : 1.0.1_SV2982_XE_16_3
      Description      : Unified Threat Defense
    Signing
      Key type         : Cisco development key
      Method           : SHA-1
    Licensing
      Name             : Not Available
      Version          : Not Available

  Detailed guest status

------------------------------------------------------------------------
Process               Status            Uptime           # of restarts
------------------------------------------------------------------------
climgr                 UP         0Y 0W 0D  0: 0:35        1
logger                 UP         0Y 0W 0D  0: 0: 4        0
snort_1                UP         0Y 0W 0D  0: 0: 4        0
Network stats:
eth0: RX  packets:43, TX  packets:6
eth1: RX  packets:8, TX  packets:6

Coredump file(s): lost+found
```

```
  Activated profile name: None
  Resource reservation
    Disk                : 736 MB
    Memory              : 1024 MB
    CPU                 : 25% system CPU

  Attached devices
    Type              Name        Alias
    ---------------------------------------------
    NIC               ieobc_1     ieobc
    NIC               dp_1_0      net2
    NIC               dp_1_1      net3
    NIC               mgmt_1      mgmt
    Disk              _rootfs
    Disk              /opt/var
    Disk              /opt/var/c
    Serial/shell                  serial0
    Serial/aux                    serial1
    Serial/Syslog                 serial2
    Serial/Trace                  serial3
    Watchdog          watchdog-2

  Network interfaces
    MAC address             Attached to interface
    ------------------------------------------------------
    54:0E:00:0B:0C:02       ieobc_1
    A4:4C:11:9E:13:8D       VirtualPortGroup0
    A4:4C:11:9E:13:8C       VirtualPortGroup1
    A4:4C:11:9E:13:8B       mgmt_1

  Guest interface
  ---
  Interface: eth2
  ip address: 48.0.0.2/24
Interface: eth1
  ip address: 47.0.0.2/24

  ---

  Guest routes
  ---
  Address/Mask                          Next Hop                            Intf.
-------------------------------------------------------------------------------
0.0.0.0/0                             48.0.0.1                           eth2
0.0.0.0/0                             47.0.0.1                           eth1

  ---

  Resource admission (without profile) : passed
    Disk space    : 710MB
    Memory        : 1024MB
    CPU           : 25% system CPU
    VCPUs         : Not specified
```

Possible Cause The AppNav tunnel may not be activated.

Solution Use the **show service-insertion type utd service-node-group** and **show service-insertion type utd service-context** commands to verify if the AppNav tunnel is activated.

Solution The following is sample output from the **show service-insertion type utd service-node-group** command:

```
Device# show service-insertion type utd service-node-group

Service Node Group name : utd_sng_1
Service Context : utd/1
Member Service Node count : 1


Service Node (SN) : 30.30.30.2
Auto discovered : No
SN belongs to SNG : utd_sng_1
Current status of SN : Alive
Time current status was reached : Tue Jul 26 11:57:48 2016

Cluster protocol VPATH version : 1
Cluster protocol incarnation number : 1
Cluster protocol last sent sequence number : 1469514497
Cluster protocol last received sequence number: 1464
Cluster protocol last received ack number : 1469514496
```

Solution The following is sample output from the **show service-insertion type utd service-context** command:

```
Device# show service-insertion type utd service-context

Service Context : utd/1
Cluster protocol VPATH version : 1
Time service context was enabled : Tue Jul 26 11:57:47 2016
Current FSM state : Operational
Time FSM entered current state : Tue Jul 26 11:57:58 2016
Last FSM state : Converging
Time FSM entered last state : Tue Jul 26 11:57:47 2016
Cluster operational state : Operational

Stable AppNav controller View:
30.30.30.1

Stable SN View:
30.30.30.2

Current AppNav Controller View:
30.30.30.1

Current SN View:
30.30.30.2
```

Possible Cause Check data plane UTD statistics for the status of the traffic. If the traffic is not diverted, the number of packets diverted and rejected will be zero. If the numbers are nonzero, then traffic diversion is happening, and the Snort sensor is resending packets back to the dataplane.

Solution Use the **show platform hardware qfp active feature utd stats** command to verify the status of the traffic.

```
Device# show platform hardware qfp active feature utd stats

Security Context:    Id:0    Name: Base Security Ctx

Summary Statistics:
Active Connections                                              29
TCP Connections Created                                     712910
UDP Connections Created                                         80
Pkts entered policy feature                       pkt      3537977
                                                  byt    273232057
```

```
Pkts entered divert feature                          pkt              3229148
                                                     byt            249344841
Pkts slow path                                       pkt               712990
                                                     byt             45391747
Pkts Diverted                                        pkt              3224752
                                                     byt            249103697
Pkts Re-injected                                     pkt              3224746
                                                     byt            249103373
….
```

Solution Alternatively, in the case of multi-tenancy, you can use the **show platform hardware qfp active feature utd stats vrf name** *vrf-name* command to verify the status of the traffic, for a specific VRF.

```
Device# show platform hardware qfp active feature utd stats vrf name vrf 101

Security Context:    Id:1    Name: 1 : vrf101

Summary Statistics:
Active Connections                                                      2
TCP Connections Created                                             34032
UDP Connections Created                                             11448
ICMP Connections Created                                               80
Pkts dropped                                         pkt              626
                                                     byt             323842
Pkts entered policy feature                          pkt              995312
                                                     byt            813163885
Pkts entered divert feature                          pkt              639349
                                                     byt            420083106
Pkts slow path                                       pkt               45560
                                                     byt             7103132
Pkts Diverted                                        pkt              638841
                                                     byt            419901335
Pkts Re-injected                                     pkt              630642
                                                     byt            412139098
….
```

Signature Update is not Working

Problem Signature update from Cisco Borderless Software Distribution (BSD) server is not working.

Possible Cause Signature update may have failed due to various reasons. Check for the reason for the last failure to update the signatures.

Solution Use the **show utd engine standard threat-inspection signature update status** command to display the reason for the last failure to update the signatures:

```
Device# show utd eng standard threat-inspection signature update status
Current signature package version: 29.0.c
Current signature package name: default
Previous signature package version: None
----------------------------------------
Last update status: Failed
----------------------------------------
Last successful update time: None
Last successful update method: None
Last successful update server: None
Last successful update speed: None
```

```
---------------------------------------
Last failed update time: Thu Jan 11 13:34:36 2018 PST
Last failed update method: Manual
Last failed update server: http://172.27.57.252/UTD-STD-SIGNATURE-2983-1-S.pkg
Last failed update reason: [Errno 113] No route to host
---------------------------------------
Last attempted update time: Thu Jan 11 13:34:36 2018 PST
Last attempted update method: Manual
Last attempted update server: http://172.27.57.252/UTD-STD-SIGNATURE-2983-1-S.pkg
---------------------------------------
Total num of updates successful: 0
Num of attempts successful: 0
Num of attempts failed: 1
Total num of attempts: 1
---------------------------------------
Next update scheduled at: None
---------------------------------------
Current status: Idle
```

Possible Cause Domain Name System (DNS) is not configured correctly.

Solution Use the **show running-config | i name-server** command to display the name server details:

```
Device#  show run | i name-server

ip name-server 10.104.49.223
```

Possible Cause System error—Failed to process the username and password combination.

Solution Ensure that you have provided the correct credentials for signature package download.

Signature Update from the Local Server is not Working

Problem Signature update from the local server not working.

Possible Cause Last failure Reason: Invalid scheme—only HTTP/HTTPS supported.

Solution Ensure that you have provided the HTTP or secure HTTP (HTTPS) as the local download method.

Possible Cause Last failure Reason: Name or service not known.

Solution Ensure that the hostname or IP address provided for the local server is correct.

Possible Cause Last failure Reason: Credentials not supplied.

Solution Ensure that you have provided the credentials for local HTTP/HTTPS server.

Possible Cause Last failure Reason: File not found.

Solution Ensure that the signature file name or URL that you have provided is correct.

Possible Cause Last failure Reason: Download corrupted.

Solution

- Verify whether the retry signature update is corrupted as the previous signature download.
- Ensure that the correct signature package is available.

Logging to IOSd Syslog is not Working

Problem Logging to IOSd syslog is not working.

Possible Cause Logging to syslog may not be configured in the unified threat defense (UTD) configuration.

Solution Use the **show utd engine standard config** command to display the UTD configuration and to ensure that logging to syslog is configured.

```
Device# show utd engine standard config

UTD Engine Standard Configutation:
  Operation Mode : Intrusion Prevention
  Policy         : Security

Signature Update:
  Server    : cisco
  User Name : ccouser
  Password  : YEX^SH\fhdOeEGaOBIQAIcOVLgaVGf
  Occurs-at : weekly ;  Days:0 ; Hour: 23; Minute: 50

Logging:
  Server    :   IOS Syslog; 10.104.49.223
  Level     : debug

Whitelist Signature IDs:
  28878
```

Solution Use the following **show utd engine standard logging events** command to display the event logs for the UTD engine.

```
Device# show utd engine standard logging events

2016/06/13-14:32:09.524475 IST [**] [Instance_ID: 1] [**] Drop [**] [1:30561:1]
BLACKLIST DNS request for known malware domain domai.ddns2.biz -
Win.Trojan.Beebone [**] [Classification: A Network Trojan was Detected]
[Priority: 1] [VRF_ID: 2] {UDP} 11.1.1.10:58016 -> 21.1.1.10:53
2016/06/13-14:32:21.524988 IST [**] [Instance_ID: 1] [**] Drop [**] [1:30561:1]
BLACKLIST DNS request for known malware domain domai.ddns2.biz -
Win.Trojan.Beebone [**] [Classification: A Network Trojan was Detected] [Priority: 1]
[VRF_ID: 2] {UDP} a000:0:0:0:0:0:0:10:59964 -> b000:0:0:0:0:0:0:10:53
```

Logging to an External Server is not Working

Problem Logging to an external server is not working.

Possible Cause Syslog may not be running on the external server.

Solution Verify whether syslog server is running on the external server. Configure the following command on the external server to view its status:

```
ps -eaf | grep syslog

 root 2073 1 0 Apr12 ? 00:00:02 syslogd -r -m
```

Possible Cause Connectivity between unified threat defense (UTD) Linux Container (LXC) and external server may be lost.

Solution Verify the connectivity from the management interface to the external syslog server.

UTD Conditional Debugging

Conditional debugging is supported by multi-tenancy for Unified Threat Defense. For further details about how to configure conditional debugging, see:

http://www.cisco.com/c/en/us/td/docs/routers/asr1000/troubleshooting/guide/Tbls hooting-xe-3s-asr-1000-book.html#task_AC969BB06B414DCBBDEF7ADD29EF8131