Use Secure Web Appliance Best Practices

Contents

Introduction

This document describes the best practices for how to configure the Cisco Secure Web Appliance (SWA).

Background Information

This guide is intended as a reference for best practice configuration and It addresses many aspects of a SWA deployment, includes the supported network environment, policy configuration, monitoring, and troubleshooting. While the best practices documented here are important for all administrators, architects, and operators to understand, they are only guidelines and must be treated as such. Each network have its own specific requirements and challenges.

As a security device, the SWA interacts with the network in several unique ways. It is a both a source and destination of web traffic. It acts at the same time as a web server and a web client. At a minimum, it employs server-side IP address spoofing and man-in-the-middle techniques to inspect HTTPS transactions. It can also spoof client IP addresses, which adds another layer of complexity to the deployment and imposes additional requirements on the supporting network configuration. This guide addresses the most common issues related to the related network device configuration.

The SWA policy configuration has implications not only for security efficacy and enforcement, but also for the performance of the appliance. This guide addresses how the complexity of a configuration impacts system resources. It defines complexity in this context and describe how to minimize it in policy design. Attention is also paid to specific features and how they must be configured to increase security, scalability, and efficacy.

The Monitoring and Alerting section of this document explains the most effective ways to monitor the appliance; and also covers the monitoring of performance and availability, as well as system resource usage. It also provides information useful in basic troubleshooting.

Network Environment

ICMP

Path MTU Discovery, as defined in RFC 1191, The mechanism determines the maximum size of a packet along arbitrary paths. In the case of IPv4, a device can determine the Maximum Transmission Unit (MTU) of any packet along a path by setting the Don’t Fragment (DF) bit in the IP header of the packet. If, at some link along the path, a device cannot forward the packet without fragment it, an **Internet Control Message Protocol (ICMP) Fragmentation Needed (Type 3, Code 4)** message is sent back to the source. The client then resends a smaller packet. This continues until the MTU for the full path is discovered. IPv6 does not support fragmentation, and uses a Packet Too Big (Type 2) ICMPv6 message to indicate the inability to fit a packet through a given link.

Because the process of packets fragmentation can have severe impacts on the performance of a TCP flow, the SWA utilizes Path MTU Discovery. The mentioned ICMP messages must be enabled in relevant network devices to allow the SWA to determine the MTU for its path through the network. This behavior can be disabled in the SWA uses the **pathmtudiscovery** command-line interface (CLI) command. Doing this causes the default MTU to drop to 576 bytes (per RFC 879), severely impacts performance. The administrator must take the additional step of manually configuring the MTU in the SWA from **etherconfig** CLI command.

In the case of the **Web Cache Communication Protocol (WCCP)**, web traffic is redirected to the SWA from another network device along the client path to the internet. In this case, other protocols, such as

ICMP, are not redirected to the SWA. There is a possibility that the SWA could trigger an ICMP Fragmentation Needed message from a router on the network, but the message would not be delivered to the SWA. If this is a possibility in the network, Path MTU Discovery must be disable. As mentioned, with this configuration, the additional step of manually setting the MTU on the SWA using the **etherconfig** CLI command is required.

Firewalls

In a default configuration, the SWA does not spoof the client IP address when proxying a connection. This means that all outbound web traffic are sourced from the SWA IP address. It is necessary to ensure that **Network Address Translation (NAT)** devices have a large enough pool of external addresses and ports to accommodate this. It is a good idea to dedicate a specific address for this purpose.

Some firewalls employ **Denial-of-Service (DoS)** protections or other security features that trigger when large numbers of simultaneous connections are sourced from a single client IP address. When client IP Spoofing is not enabled, the SWA IP address must be excluded from these protections.

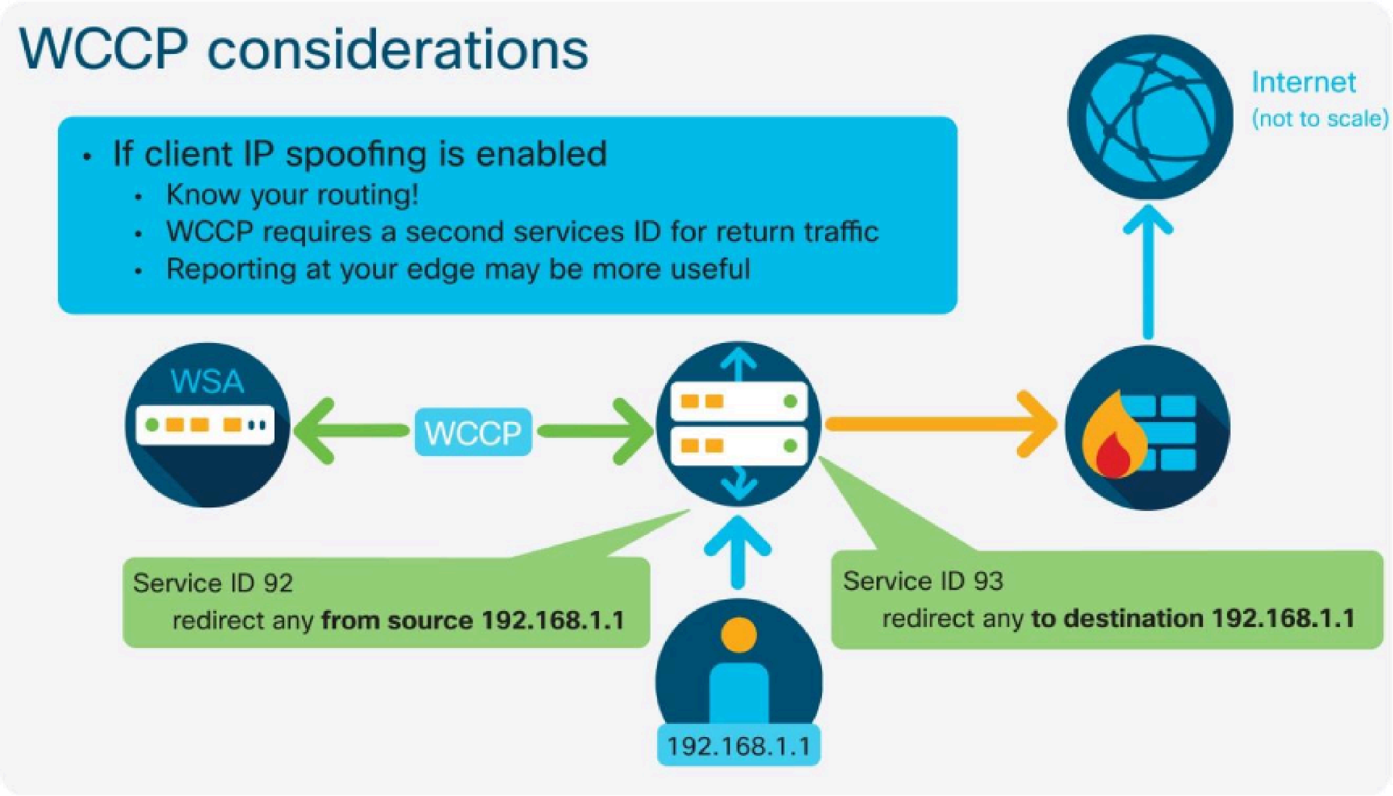
Unicast Reverse Path Forwarding

The SWA spoofs the server IP address when communicats with a client, and optionally can be configured to spoof the client IP address when it communicates with an upstream server. Protections such as **Unicast Reverse Path Forwarding (uRPF)** can be enabled on switches to ensure that an incoming packet matches the expected ingress port. These protections check the source interface of a packet against the routing table to ensure that it arrived on the expected port. The SWA needs to be exempted from these protections where appropriate.

IP Spoofing with WCCP

When the IP Spoofing feature is enabled in the SWA, outbound requests leaves the appliance use the source address of the original client request. This requires additional configuration of the related network infrastructure to ensure that return packets are routed to the SWA outbound interface, instead of the client that originated the request.

When WCCP is implemented on a network device (router, switch, or firewall), a service ID is defined that matches traffic based on an **Access Control List (ACL)**. The service ID is then applied to an interface and used to match traffic for redirection. If IP Spoofing is enabled, a second service ID must be created to ensure that return traffic is also redirected to the SWA.



WCCP considerations

- If client IP spoofing is enabled
  - Know your routing!
  - WCCP requires a second services ID for return traffic
  - Reporting at your edge may be more useful

Service ID 92
redirect any **from source 192.168.1.1**

Service ID 93
redirect any **to destination 192.168.1.1**

SWA Network Configuration

Interfaces

The SWA has five usable network interfaces: M1, P1, P2, T1, and T2. Each of these must be leveraged for their specific purpose when possible. It is beneficial to use each port for its own reasons. The M1 interface must be connected to a dedicated management network, and split-routing must be enabled to limit the exposure of administrative services. The P1 can be limited to client request traffic, In contrast, P2 is not allowed to accept explicit proxy requests. This decrease the amount of traffic on each interface and allow better segmentation in the network design.

The T1 and T2 ports are available for the **Layer 4 Traffic Monitor (L4TM)** feature. This feature monitors a mirrored layer 2 port and adds the ability to block traffic based on a blocked list of known malicious IP addresses and domain names. It does this by look at the source and destination IP addresses of traffic and sends a TCP reset packet, or Port Unreachable message if the blocked list is matched. Traffic sent with any protocol can be blocked with this feature.

Even if the L4TM feature is not enabled, transparent bypass can be enhanced when the T1 and T2 ports are attached to a mirrored port. In the case of WCCP, the SWA only knows the source and destination IP address of an incoming packet and must make a decision to proxy it, or to bypass it based on that information. The SWA resolves any entries in the bypass settings list every 30 minutes, regardless of the record’s **Time to Live (TTL)**. However, if the L4TM feature is enabled, the SWA can use snooped DNS queries to update these records more frequently. This reduces the risk of a false negative in a scenario where the client has resolved a different address from the SWA.

Management Network Routing

If the dedicated management network does not have internet access, each service can be configured to use the data routing table. This can be tailored to fit the network topology, but in general, it is recommended to use the management network for all system services and the data network for client traffic. As of AsyncOS version 11.0, the services for which routing can be set are:

- External URL feeds
- **Advanced Malware Protection (AMP)** file reputation and analysis
- Updates and upgrades
- DNS
- Active directory

For additional egress filtering of management traffic, static addresses can be configured for use in these services:

- **External URL feeds**:
  1. Custom depends on where they are hosted
  2. AMP file reputation and analysis
  3. cloud-sa.amp.cisco.com (North America)
  4. cloud-sa.eu.amp.cisco.com (Europe)
  5. cloud-sa.apjc.amp.cisco.com (Asia Pacific)

- **Update and upgrades**:
  1. downloads-static.ironport.com
  2. updates-static.ironport.com

TALOS Telemetry

The Cisco Talos group is well known for identifying new and emerging threats. All data sent to Talos is anonymized and stored in U.S. data centers. Participating in SensorBase enhances the categorization and identification of web threats and leads to better protection from the SWA, as well as other Cisco security solutions.

DNS

Domain Name Server (DNS) security best practices suggest that every network must host two DNS resolvers: one for authoritative records from within a local domain, and one for recursive resolution of Internet domains. To accommodate this, the SWA allows DNS servers to be configured for specific domains. If only one DNS server is available for both local and recursive queries, Consider the additional load it adds when used for all SWA queries. The better option can be to use the internal resolver for local domains and the root Internet resolvers for external domains. This is dependent on the administrator risk profile and tolerance.

By default, the SWA cache a DNS record for a minimum of 30 minutes, regardless of the record’s TTL. Modern websites that make heavy use of **Content Delivery Networks (CDNs)** have low TTL records as their IP addresses change frequently. This could result in a client caching one IP address for a given server and the SWA caching a different address for the same server. To counter this, the SWA default TTL can be lowered to five minutes from these CLI commands:

```
SWA_CLI> dnsconfig
...
Choose the operation you want to perform:
- NEW - Add a new server.
- EDIT - Edit a server.
- DELETE - Remove a server.
- SETUP - Configure general settings.
- SEARCH - Configure DNS domain search list.
[]> SETUP
...
Enter the minimum TTL in seconds for DNS cache.
...
```

Secondary DNS servers must be configured in case the primary is not available. If all servers are configured with the same priority, the server IP is chosen at random. Depending on the number of servers configured, the timeout for a given server can vary. The table is the timeout for a query for up to six DNS servers:

Number of DNS servers	**Query timeout (in sequence)**
1	60
2	5, 45
3	5, 10, 45
4	1, 3, 11, 45
5	1, 3, 11, 45, 1
6	1, 3, 11, 45, 1, 1

There are also advanced DNS options available only through the CLI. These options are available in CLI using the **advancedproxyconfig > DNS** command.

Select one of these options:

- 0—Always use DNS answers in order
- 1—Use client-supplied address then DNS
- 2—Limited DNS usage
- 3—Very limited DNS usage

For options 1 and 2, DNS is used if Web Reputation is enabled.

For options 2 and 3, DNS is used for explicit proxy requests, if there is no upstream proxy or in the event the configured upstream proxy fails.

For all options, DNS is used when Destination IP addresses are used in policy membership.

These options control how the SWA decides on the IP address to connect to when evaluating a client request. When a request is received, the SWA see a destination IP address and a hostname. The SWA must decide whether to trust the original destination IP address for the TCP connection, or to do its own DNS resolution and use the resolved address. The default is “0 = Always use DNS answers in order,” which means the SWA does not trust the client to supply the IP address.

- Option 1—The SWA tries the client-supplied IP address for the connection, but falls back to the resolved address if that fails. The resolved address is used for policy evaluation (web category, web reputation, and so on).
- Option 2—The SWA only uses the client-supplied address for the connection and does not fall back. The resolved address is used for policy evaluation (web category, web reputation, and so on).

- Option 3—The SWA only uses the client-supplied address for the connection and does not fall back. The client-supplied IP address is used for policy evaluation (web category, web reputation, and so on).

The chosen option depends on how much trust the administrator must place in the client when determining the resolved address for a given hostname. If the client is a downstream proxy, choose option 3 to avoid the added latency of unnecessary DNS lookups.

Load Balancing

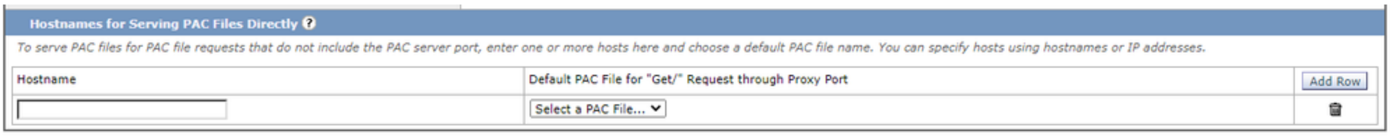
WCCP allows for transparent traffic load balancing when up to eight appliances are used. It allows for balancing traffic flows based on hash or mask, it can be weighted in case there is a mix of appliance models in the network, and devices can be added and removed from the service pool without downtime. Once the need exceeds what can be handled with eight SWAs, it is recommended to use a dedicated load balancer.

Specific best practices for WCCP configuration vary based on the platform used. For Cisco Catalyst® switches, best practices are documented in Cisco Catalyst Instant Access Solution White Paper .

WCCP has limitations when used with a Cisco Adaptive Security Appliance (ASA). Namely, client IP spoofing is not supported. In addition, the clients and SWA must be behind the same interface. For this reason, it is more flexible to use a layer 4 switch or router to redirect traffic. WCCP configuration on the ASA platform is described in WCCP on ASA: Concepts, Limitations, and Configuration.

For explicit deployments, a Proxy Autoconfiguration (PAC) file is the most widely deployed method, but it has many drawbacks and security implications that are beyond the scope of this document. If a PAC file is deployed, it is suggested to use Group Policy Objects (GPOs) to configure the location rather than relying on the Web Proxy Autodiscover Protocol (WPAD) that is a common target for attackers and can be exploited easily if misconfigured. The SWA can host multiple PAC files and control their expiration in the browser’s cache.

A PAC file can be requested directly from the SWA from a configurable TCP port number (9001 by default). If a port is not specified, the request can be sent to the proxy process itself as though it were an outbound web request. In this case, it is possible to serve a specific PAC file based on the HTTP host header present in the request.



Hostnames for Serving PAC Files Directly

To serve PAC files for PAC file requests that do not include the PAC server port, enter one or more hosts here and choose a default PAC file name. You can specify hosts using hostnames or IP addresses.

Hostname	Default PAC File for "Get/" Request through Proxy Port	Add Row
	Select a PAC File...	

Kerberos must be configured differently when used in a high availability environment. The SWA provides support for keytab files, which allows for multiple hostnames to be associated with a **Service Principle Name (SPN)**. For more information, see Creating a Service Account in Windows Active Directory for Kerberos Authentication in High Availability Deployments .

Active Authentication

Kerberos is more secure and widely supported authentication protocol than **NT LAN Manager Security Support Provider (NTLMSSP)**. The Apple OS X operating system does not support NTLMSSP, but can use Kerberos to authenticate if domain joined. Basic authentication must not be used, as it sends credentials unencrypted in the HTTP header and can be easily sniffed by an attacker on the network. If basic authentication must be used, credential encryption must be enabled to ensure that credentials are sent over an encrypted tunnel.

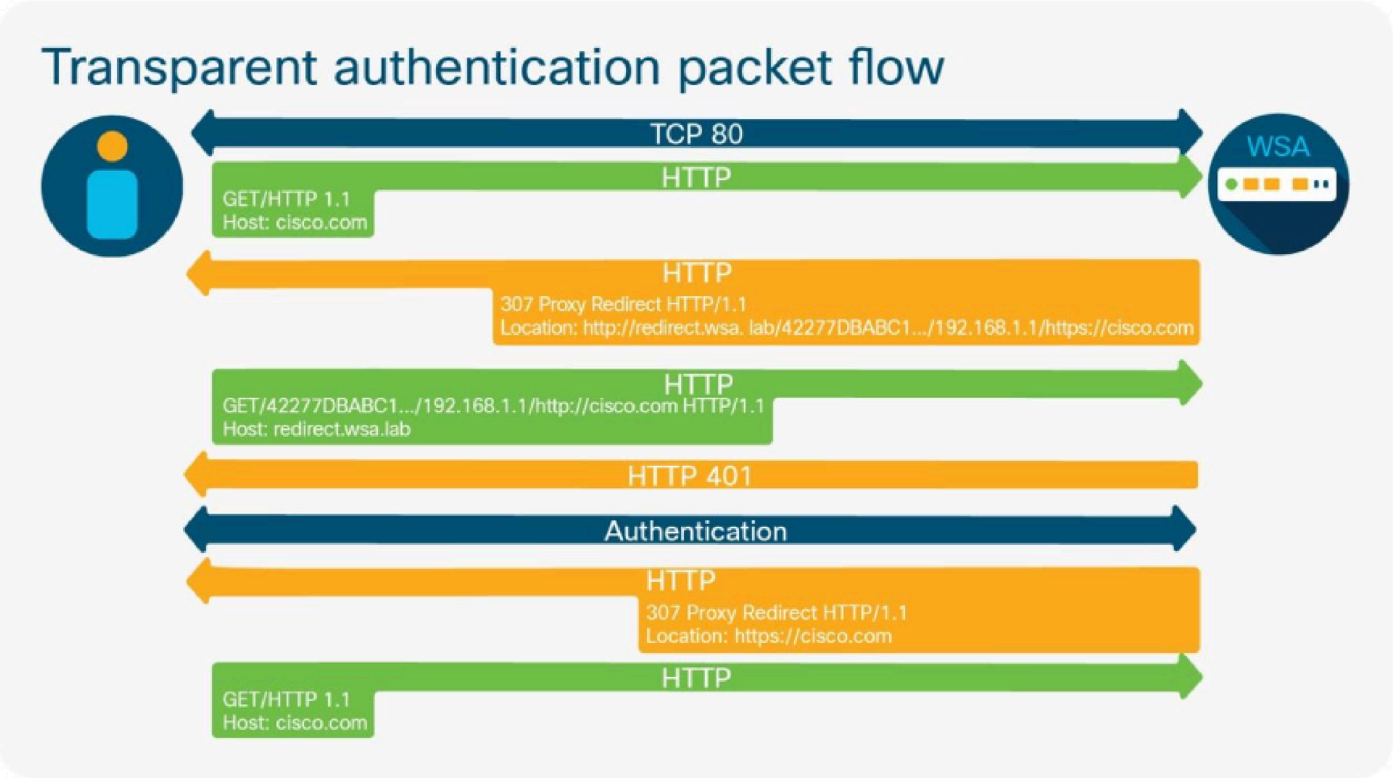
More than one domain controller must be added to the configuration to ensure availability, but there is no inherent load balancing of this traffic. The SWA send a TCP SYN packet to all configured domain controllers and the first to respond is used for authentication.

The redirect hostname that is configured in the authentication settings page determines where a transparent client is sent in order to complete authentication. For a Windows client to complete integrated authentication and achieve **Single Sign-On (SSO)**, the redirect hostname must be in the **Trusted Sites** zone in the **Internet Options** control panel. The Kerberos protocol requires that the **Fully Qualified Domain Name (FQDN)** be used to specify a resource, which means the “shortname” (or “NETBIOS” name) cannot be used if Kerberos is the intended authentication mechanism. The FQDN need to be manually added to the **Trusted Sites** (for example, through Group Policy). Additionally, the Automatic login with username and password must be set in the **Internet Options** control panel.

Additional settings are also required in Firefox for the browser to complete authentication with network proxies. Those settings can be configured in the **about:config** page. For Kerberos to complete successfully, the redirect hostname must be added to the **network.negotiate-auth.trusted-uris** option. For NTLMSSP, it must be added to the **network.automatic-ntlm-auth.trusted-uris** option.

Authentication surrogates are used to remember an authenticated user for a set duration after authentication has completed. IP surrogates must be used whenever possible to limit the number of active authentication events that occur. Actively authenticating a client is a resource-intensive task, especially when Kerberos is used. The surrogate timeout is 3600 seconds (one hour) by default and can be lowered, but the lowest recommended value is 900 seconds (15 minutes).

This image shows how “redirect.WSA.lab” is used as the redirect hostname:


Transparent authentication packet flow
WSA
TCP 80
HTTP
GET/HTTP 1.1
Host: cisco.com
HTTP
307 Proxy Redirect HTTP/1.1
Location: http://redirect.wsa. lab/42277DBABC1.../192.168.1.1/https://cisco.com
HTTP
GET/42277DBABC1.../192.168.1.1/http://cisco.com HTTP/1.1
Host: redirect.wsa.lab
HTTP 401
Authentication
HTTP
307 Proxy Redirect HTTP/1.1
Location: https://cisco.com
HTTP
GET/HTTP 1.1
Host: cisco.com

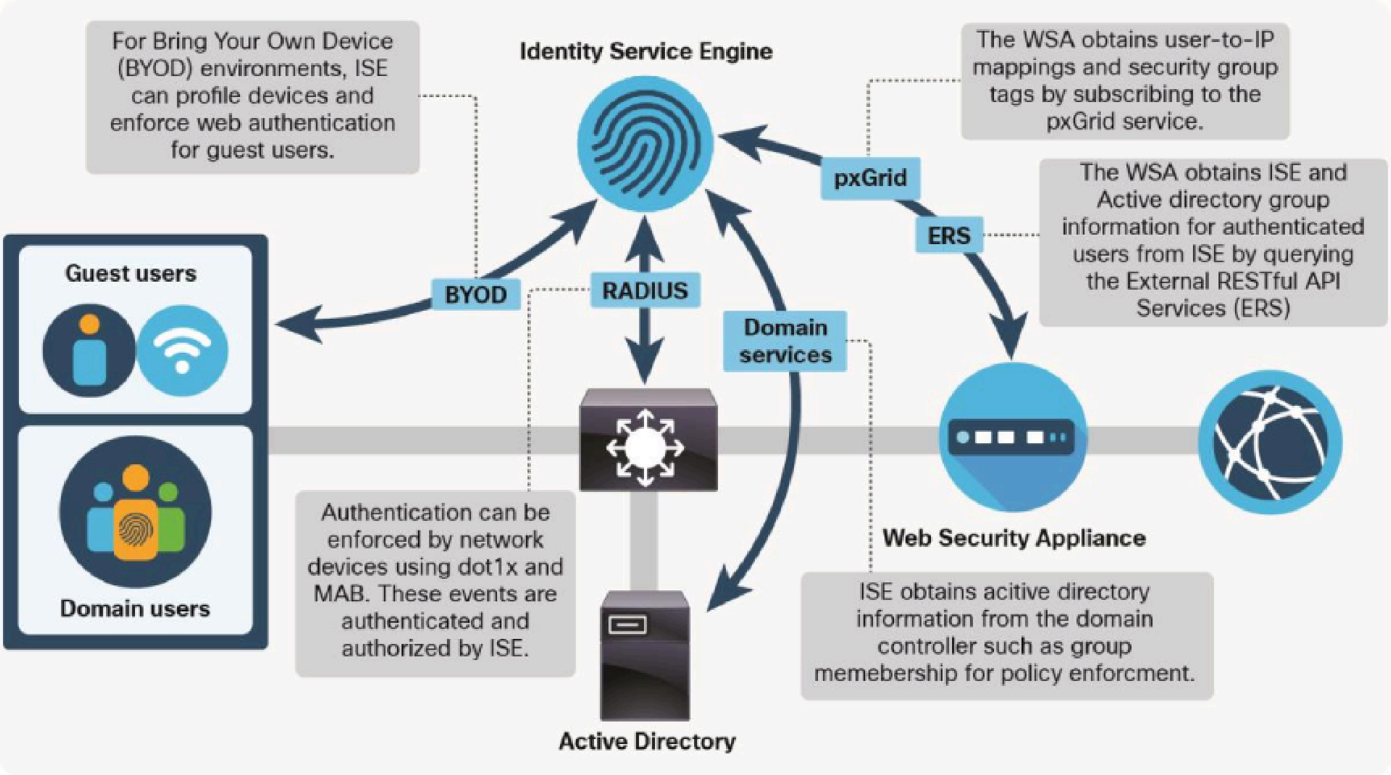
Passive Authentication

The SWA can leverage other Cisco security platforms to passively identify proxy users. Passively users identification eliminates the need for a direct authentication challenge and any Active Directory communication from the SWA, which in turn reduces latency and resource usage on the appliance. The currently available mechanisms for passive authentication are through the **Context Directory Agent (CDA)**, the **Identity Services Engine (ISE)**, and the **Identity Services Connector Passive Identity Connector (ISE-PIC)**.

ISE is a feature-rich product that helps administrators centralize their authentication services and leverage an extensive set of network access controls. When ISE learns about a user authentication event (either through Dot1x authentication or web authentication redirect), it populates a session database that contains information about the user and device involved in the authentication. The SWA connects to ISE over the **Platform Exchange Grid (pxGrid)** and obtains the user name, IP address, and Security Group Tag (SGT) associated with a proxy connection. Since AsyncOS version 11.7, the SWA can also query the **External Restful Service (ERS)** on ISE to obtain group information.

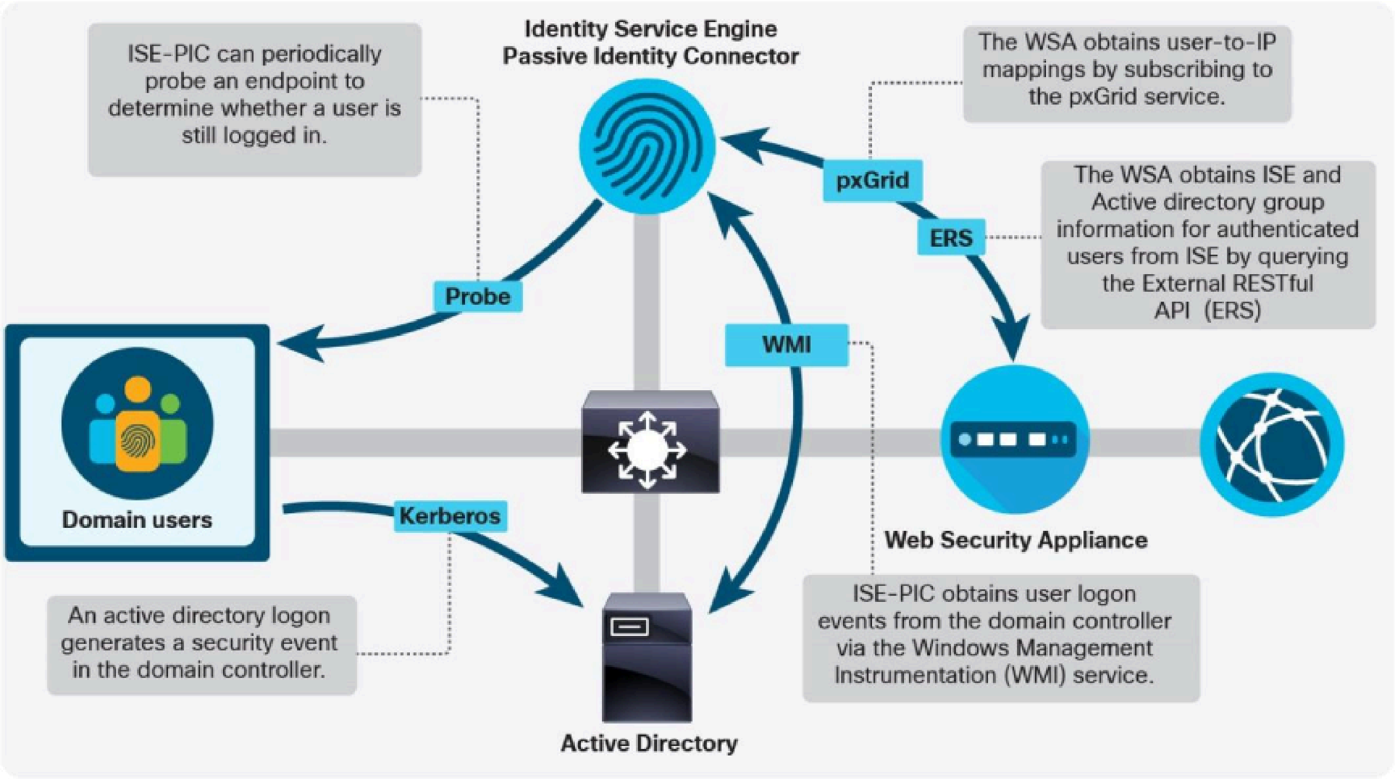
The suggested versions are ISE 3.1 and SWA 14.0.2-X and later, For more information about ISE Compatibility Matrix for SWA, see ISE Compatibility Matrix for Secure Web Appliance .

For more information on full integration steps, see Web Security Appliance End-User Guide.


For Bring Your Own Device (BYOD) environments, ISE can profile devices and enforce web authentication for guest users.
Identity Service Engine
The WSA obtains user-to-IP mappings and security group tags by subscribing to the pxGrid service.
pxGrid
ERS
The WSA obtains ISE and Active directory group information for authenticated users from ISE by querying the External RESTful API Services (ERS)
Guest users
BYOD
RADIUS
Domain services
Domain users
Authentication can be enforced by network devices using dot1x and MAB. These events are authenticated and authorized by ISE.
Web Security Appliance
ISE obtains acitive directory information from the domain controller such as group memebership for policy enforcment.
Active Directory

Cisco announces the end-of-life for the Cisco Context Directory Agent (CDA) Software, see Cisco Context Directory Agent (CDA).

As of CDA patch 6, is compatible with Microsoft Server 2016. However, administrators are actively encouraged to migrate their CDA deployments to ISE-PIC. Both solutions use WMI to subscribe to the Windows Security Event Log to generate user-to-IP mappings (known as “sessions”). In the case of CDA, the SWA queries these mappings with RADIUS. In the case of ISE-PIC, the same pxGrid and ERS connections are used as in the full ISE deployment. ISE-PIC functionality is available in a full ISE installation, as well as in a standalone virtual appliance.



ISE-PIC can periodically probe an endpoint to determine whether a user is still logged in.

The WSA obtains user-to-IP mappings by subscribing to the pxGrid service.

The WSA obtains ISE and Active directory group information for authenticated users from ISE by querying the External RESTful API (ERS)

An active directory logon generates a security event in the domain controller.

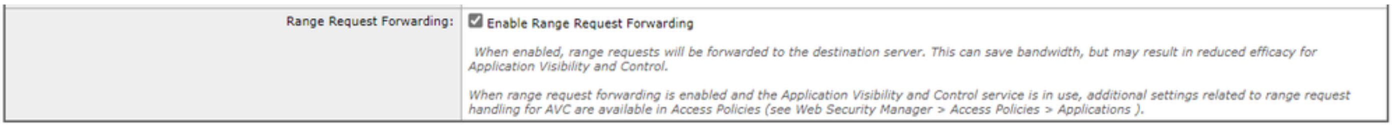
ISE-PIC obtains user logon events from the domain controller via the Windows Management Instrumentation (WMI) service.

Services Configuration

Web Proxy

Caching must be enabled in the web proxy configuration in order to save bandwidth and boost performance. This is becoming less important as the percentage of HTTPS traffic increases because the SWA does not by default cache HTTPS transactions. If the proxy is deployed to serve only explicit clients, forward mode must be specified in order to reject any traffic that is not specifically destined for the proxy service. In this way, the appliance attack surface is reduced and a good security principle is practiced: turn it off if it is not needed.

Range request headers are used in HTTP requests to specify the byte range of a file to be downloaded. It is commonly used by operating system and application update daemons to transfer small portions of a file at a time. By default, the SWA strip these headers so that it can obtain the entire file for the purposes of Antivirus (AV) scanning, file reputation and analysis, and **Application Visibility Control (AVC)**. Enabling the forwarding of range request headers globally in the proxy settings allow administrators to create individual access policies that forward or strip those headers. More information about this configuration is explained in **Access Policies**, section.



Range Request Forwarding: ☑ Enable Range Request Forwarding

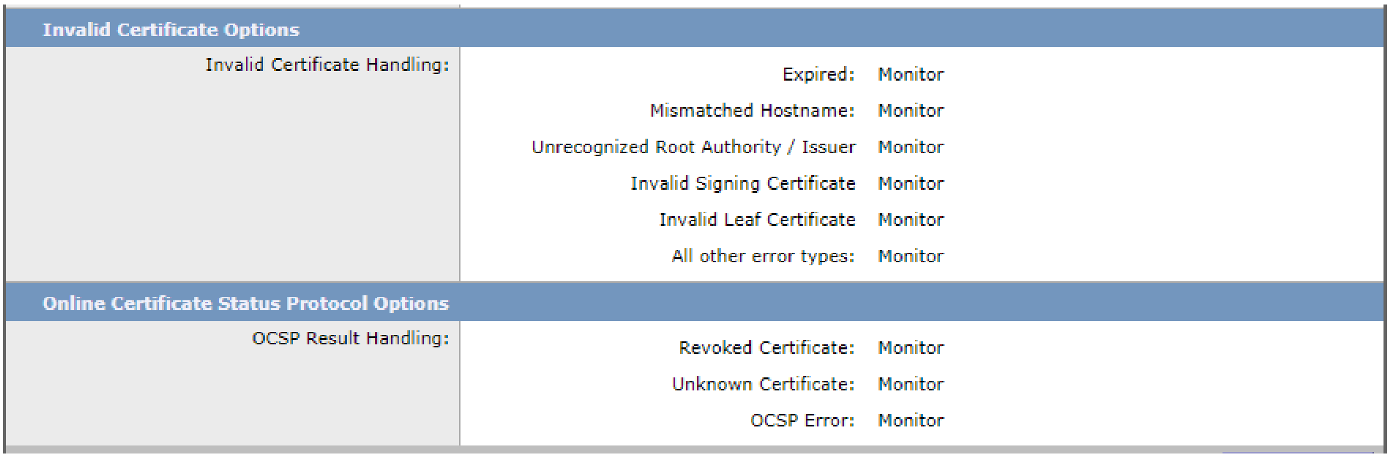
When enabled, range requests will be forwarded to the destination server. This can save bandwidth, but may result in reduced efficacy for Application Visibility and Control.

When range request forwarding is enabled and the Application Visibility and Control service is in use, additional settings related to range request handling for AVC are available in Access Policies (see Web Security Manager > Access Policies > Applications).

HTTPS Proxy

Security best practices suggest that private keys must be generated on the appliance where they are used and never be transported elsewhere. The HTTPS proxy wizard allows the creation of the key pair and certificate used for decryption of **Transport Layer Security (TLS)** connections. The **Certificate Signing Request (CSR)** can then be downloaded and signed by an in-house **Certificate Authority (CA)**. In an **Active Directory (AD)** environment, this is the best method because an AD-integrated CA is trusted automatically by all members of the domain and is not require additional steps to deploy the certificate.

One security function of the HTTPS proxy is to validate server certificates. Best practices suggest that invalid certificates require that the connection is dropped. Enabling Decrypt for EUN allows the SWA to present a block page explaining the reason for the block. Without this enabled, any HTTPS sites that are blocked, result in a browser error. This lead to increased help desk tickets and an assumption on the part of the user that something is broken, rather than the knowledge that the SWA has blocked the connection. All invalid certificate options must be set to at least Decrypt. Leaving any of these options as Monitor cannot log useful error messages in the event that certificate problems prevent a site from loading.



Invalid Certificate Options		
Invalid Certificate Handling:	Expired:	Monitor
	Mismatched Hostname:	Monitor
	Unrecognized Root Authority / Issuer	Monitor
	Invalid Signing Certificate	Monitor
	Invalid Leaf Certificate	Monitor
	All other error types:	Monitor
Online Certificate Status Protocol Options		
OCSP Result Handling:	Revoked Certificate:	Monitor
	Unknown Certificate:	Monitor
	OCSP Error:	Monitor

Similarly, **Online Certificate Services Protocol (OCSP)** checks must be left enabled and Monitor must not be used for any option. Revoked certificates must be dropped and all others must be at least set to Decrypt to allow logging of relevant error messages. **Authority Information Access Chasing (AIA chasing)** is a means by which a client can glean the signer of the certificate, and a URL from which additional certificates can be fetched. For example, if a certificate chain received from a server is incomplete (it is missing an intermediate or root certificate), the SWA can check the AIA field and use it to fetch the missing certificates and verify authenticity. This setting is only available in the CLI from these commands:

```
SWA_CLI> advancedproxyconfig

Choose a parameter group:
- AUTHENTICATION - Authentication related parameters
- CACHING - Proxy Caching related parameters
- DNS - DNS related parameters
- EUN - EUN related parameters
- NATIVEFTP - Native FTP related parameters
- FTPOVERHTTP - FTP Over HTTP related parameters
- HTTPS - HTTPS related parameters
- SCANNING - Scanning related parameters
- PROXYCONN - Proxy connection header related parameters
- CUSTOMHEADERS - Manage custom request headers for specific domains
- MISCELLANEOUS - Miscellaneous proxy related parameters
- SOCKS - SOCKS Proxy parameters
- CONTENT-ENCODING - Block content-encoding types
- SCANNERS - Scanner related parameters
[]> HTTPS
...
Do you want to enable automatic discovery and download of missing Intermediate Certificates?
[Y]>
...
```

Note: This setting is enabled by default and must not be disabled, as many modern servers rely on this mechanism to provide a full trust chain to clients.

Layer 4 Traffic Monitor (L4TM)

The L4TM is a highly effective way to extend the reach of the SWA to include malicious traffic that does not traverse the proxy, as well as to include traffic on all TCP and UDP ports. The T1 and T2 ports are intended to be connected to either a network tap, or switch monitor session. This allows SWA to passively monitor all traffic from clients. If traffic destined for a malicious IP address is seen, the SWA can terminate TCP sessions by sending an RST while spoofing the server IP address. For UDP traffic, it can send a Port Unreachable message. When configuring the monitor session, it is best to exclude any traffic destined for the management interface of the SWA to prevent the feature from potentially interfering with access to the device.

In addition to monitoring for malicious traffic, the L4TM also snoops DNS queries in order to update the bypass settings list. This list is used in WCCP deployments to return certain requests back to the WCCP router for direct routing to the web server. Packets that match the bypass settings list are not processed by the proxy. The list can contain IP addresses or server names. The SWA resolves any entries in the bypass settings list every 30 minutes, regardless of the record’s TTL. However, if the L4TM feature is enabled, the SWA can use snooped DNS queries to update these records more frequently. This reduces the risk of a false negative in a scenario where the client has resolved a different address from the SWA.

Policy Configuration

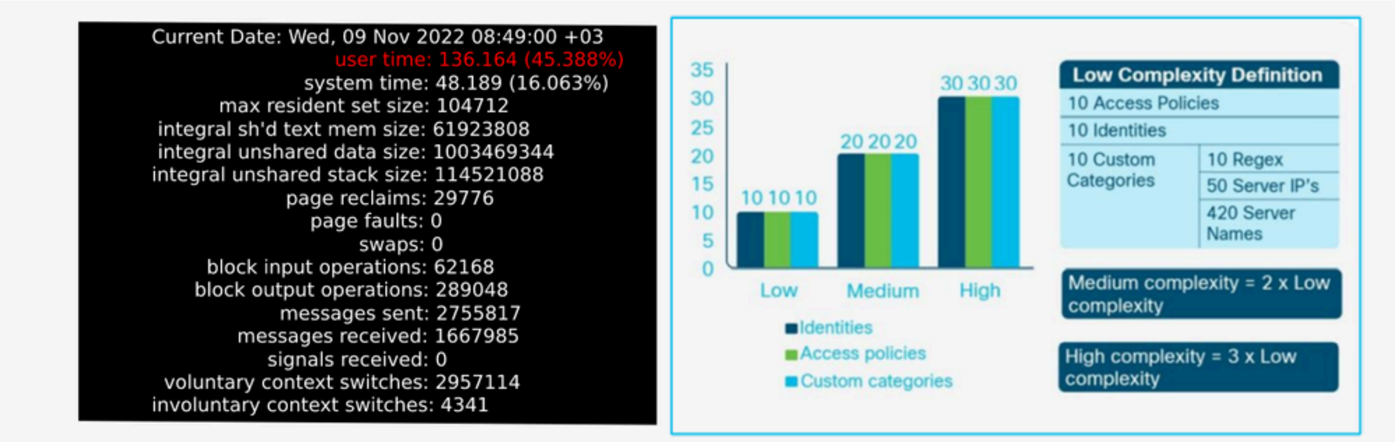
Correct policy configuration is central to the performance and scalability of the SWA. This is true not just because of the effectiveness of the policies themselves in protecting clients and enforcing company requirements, but also because which policies are configured has a direct impact on resource usage and the overall health and performance of the SWA. An overly complex or poorly designed set of policies can cause instability and slow responsiveness from the appliance.

Complexity

Various policy elements are used in the construction of SWA policies. The XML file that is generated from the configuration is used to create a number of back-end configuration files and access rules. The more complex the configuration, the more time the proxy process has to spend evaluating the various rule sets for each transaction. In benchmarking and sizing the SWA, a basic set of policy elements is created that represent three tiers of configuration complexity. Ten Identity profiles, Decryption policies, and Access policies, along with ten custom categories containing ten regex entries, fifty server IP addresses, and 420 server hostnames, is considered a Low Complexity configuration. Multiplying each of those figures by two and three result in a Medium Complexity and High Complexity configuration, respectively.

When a configuration becomes too complex, the first symptoms usually include slow response in the web interface and CLI. There cannot be a significant impact to users at first. But the more complex the configuration is, the more time the proxy process must spend in user mode. Because of this, checking the percentage of time spent in this mode can be a useful way to diagnose an overly complex configuration as the cause of a slow SWA.

The CPU time, in seconds, is logged in the track_stats log every five minutes. This means that the user time percentage can be calculated as (user time + system time)/300. As the user time approaches 270, the process is spending too many CPU cycles in user mode, and this is almost always because the configuration is too complex to be parsed efficiently.



```
      Current Date: Wed, 09 Nov 2022 08:49:00 +03
                          user time: 136.164 (45.388%)
                        system time: 48.189 (16.063%)
                max resident set size: 104712
         integral sh'd text mem size: 61923808
       integral unshared data size: 1003469344
      integral unshared stack size: 114521088
                     page reclaims: 29776
                        page faults: 0
                              swaps: 0
             block input operations: 62168
            block output operations: 289048
                     messages sent: 2755817
                 messages received: 1667985
                   signals received: 0
       voluntary context switches: 2957114
     involuntary context switches: 4341
```

Low Complexity Definition

10 Access Policies

10 Identities

10 Custom Categories: 10 Regex; 50 Server IP's; 420 Server Names

Medium complexity = 2 x Low complexity

High complexity = 3 x Low complexity

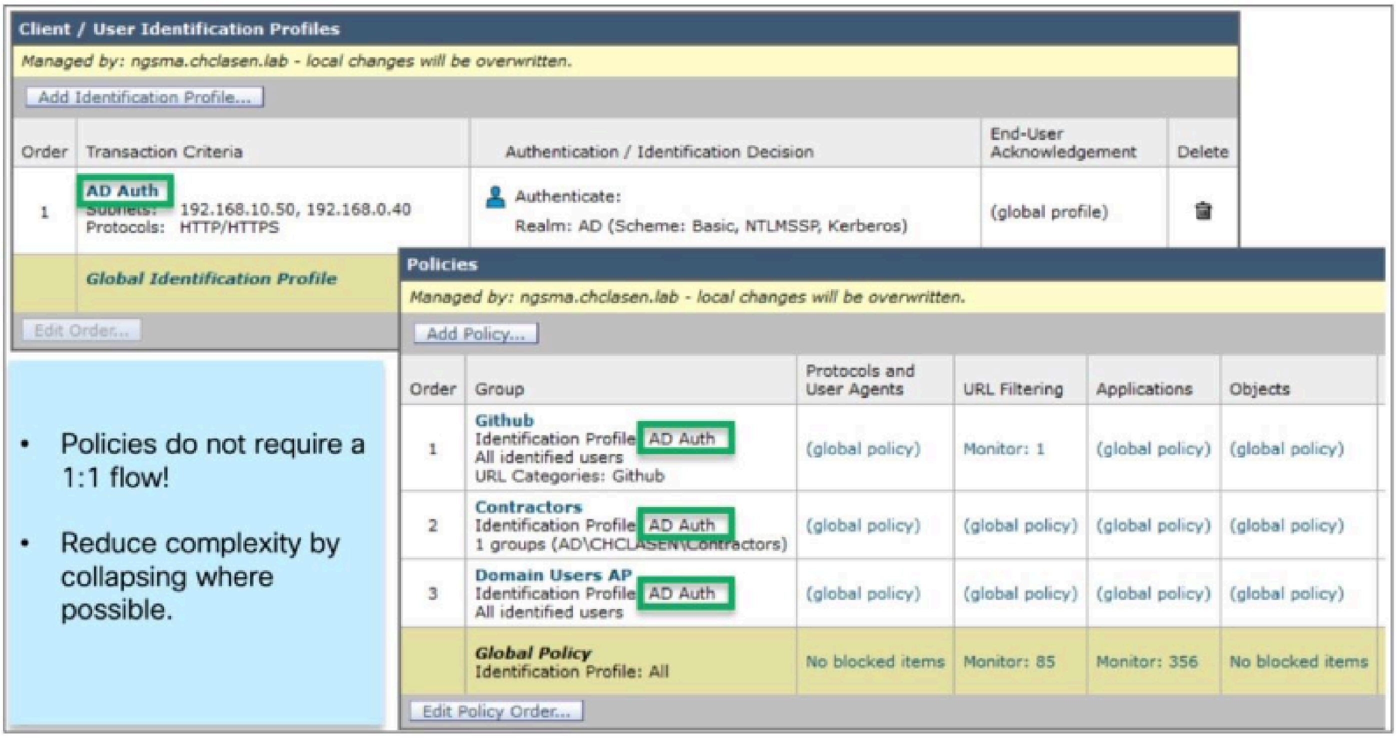
Note: So far, SWA has the maximum limitation of 60,000 concurrent client connections and 60,000 concurrent server connections.

Identification Profiles

The Identification (ID) profiles are the first policy elements that are evaluated when a new request is received. All the information configured in the first section of the ID profile is evaluated with a logical AND. This means that all criteria must match for the request to match the profile. When creating a policy, it must only be as specific as absolutely necessary. Profiles that include individual host addresses are almost never necessary, and can lead to sprawling configurations. Leveraging the user-agent string found in the HTTP headers, custom category list, or subnet is generally a better strategy to limit the scope of a profile.

In general, policies that require authentication are configured at the bottom and exceptions added on top of them. When ordering policies that do not require authentication, the most used policies must be the closest to the top as possible. Do not rely on failed authentication to restrict access. If a client on the network is known to be unable to authenticate to a proxy, it must be exempted from authentication and blocked in the access policies. Clients who cannot authenticate repeatedly send unauthenticated requests to the SWA, which use resources and can cause excessive CPU and memory utilization.

A common misconception for administrators is that there must be a unique ID profile and corresponding decryption policy and access policy. This is an inefficient strategy for policy configuration. When possible, policies must be "collapsed" so that a single ID profile can be associated with multiple decryption and access policies. This is possible because all of the criteria in a given policy must match in order for traffic to match the policy. Being more general in the authentication policy and more specific in the resulting policies allows for fewer policies as a whole.



Client / User Identification Profiles

Managed by: ngsma.chclasen.lab - local changes will be overwritten.

Order	Transaction Criteria	Authentication / Identification Decision	End-User Acknowledgement	Delete
1	AD Auth Subnets: 192.168.10.50, 192.168.0.40 Protocols: HTTP/HTTPS	Authenticate: Realm: AD (Scheme: Basic, NTLMSSP, Kerberos)	(global profile)	🗑
	Global Identification Profile			

Policies

Managed by: ngsma.chclasen.lab - local changes will be overwritten.

Order	Group	Protocols and User Agents	URL Filtering	Applications	Objects
1	Github Identification Profile AD Auth All identified users URL Categories: Github	(global policy)	Monitor: 1	(global policy)	(global policy)
2	Contractors Identification Profile AD Auth 1 groups (AD\CHCLASEN\Contractors)	(global policy)	(global policy)	(global policy)	(global policy)
3	Domain Users AP Identification Profile AD Auth All identified users	(global policy)	(global policy)	(global policy)	(global policy)
	Global Policy Identification Profile: All	No blocked items	Monitor: 85	Monitor: 356	No blocked items

- Policies do not require a 1:1 flow!
- Reduce complexity by collapsing where possible.

Decryption Policies

As with the ID profile, the criteria set in the decryption policy is also evaluated as a logical AND, with one important exception when information from the ISE is used. Here is how policy-matching works, depending on what elements are configured (AD group, user, or SGT):

- AD groups and users—No change to previous behavior; the policy is matched if the user is a member of group, OR the user is specified in the policy.
- SGT and AD groups and users—The policy is matched if the user is associated with the SGT AND is

a member of the AD group, OR the user is specified in the policy.

- SGT and users—The policy is matched if the user is associated with the SGT or the user is specified in the policy.

Of all the services performed by the SWA, evaluation of HTTPS traffic is the most significant from a performance standpoint. The percentage of decrypted traffic has a direct impact on how the appliance must be sized. An administrator can count on at least 75% of web traffic to be HTTPS.

After initial installation, the percentage of decrypted traffic must be determined to ensure that the expectations for future growth are accurately set. After deployment, this number must be checked once a quarter. Finding the percentage of HTTPS traffic that is decrypted by the SWA is easy to do with a copy of the access_logs, even without additional log management software. Simple Bash or PowerShell commands can be used to obtain this number. Here are the steps that are described for each environment:

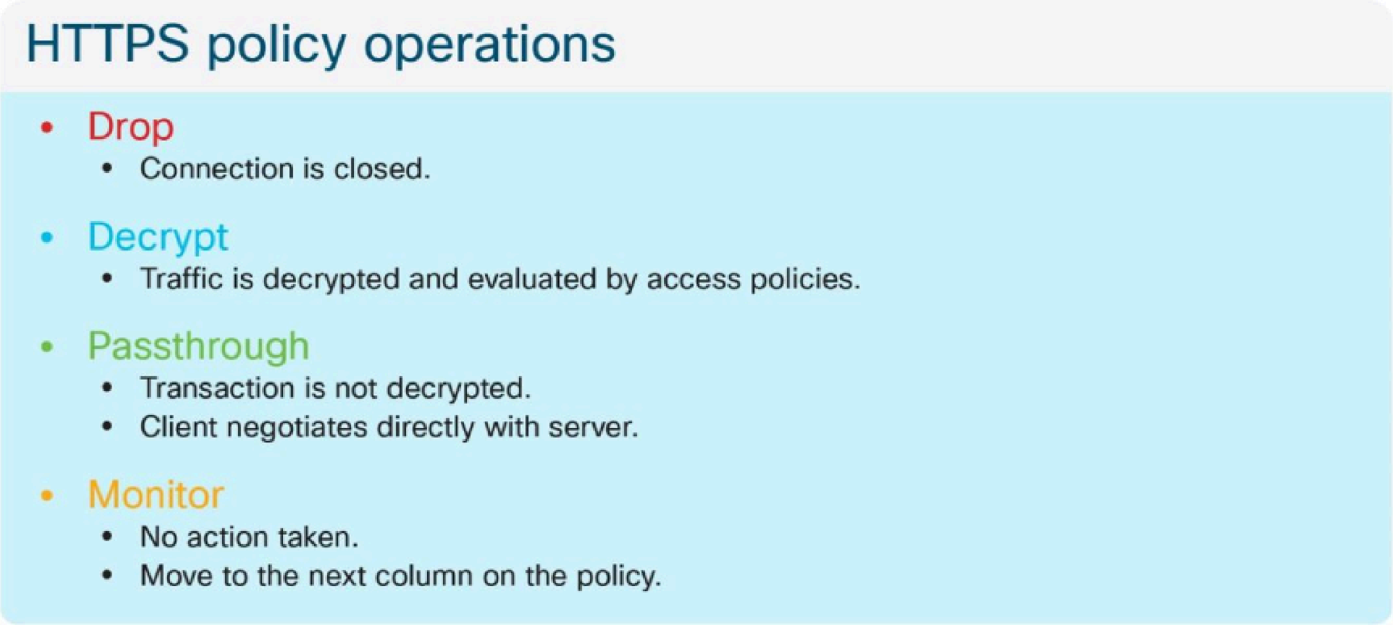
1. Linux Command:

```
cat aclog.current | grep -Ev "\/407|\/401" | awk 'BEGIN { total=0; decrypt=0; ssl=0;} {total++; if($0 ~
```

2. Powershell Command:

```
$lines = Get-Content -Path "aclog.current" | Where-Object { $_ -notmatch "/407|/401" }; $total = 0; $de
```

When designing decryption policies, it is important to understand how the various actions listed in the policy cause the appliance to evaluate HTTPS connections. The passthrough action is used when the client and server must be allowed to terminate each end of their TLS session without the SWA decrypting every packet. Even if a site is set to passthrough, the SWA must still be required to complete one TLS handshake with the server. This is because the SWA must choose to block a connection based on the certificate validity, and it must initiate a TLS connection with the server to obtain the certificate. If the certificate is valid, the SWA closes the connection and allows the client to continue setting up the session directly with the server.

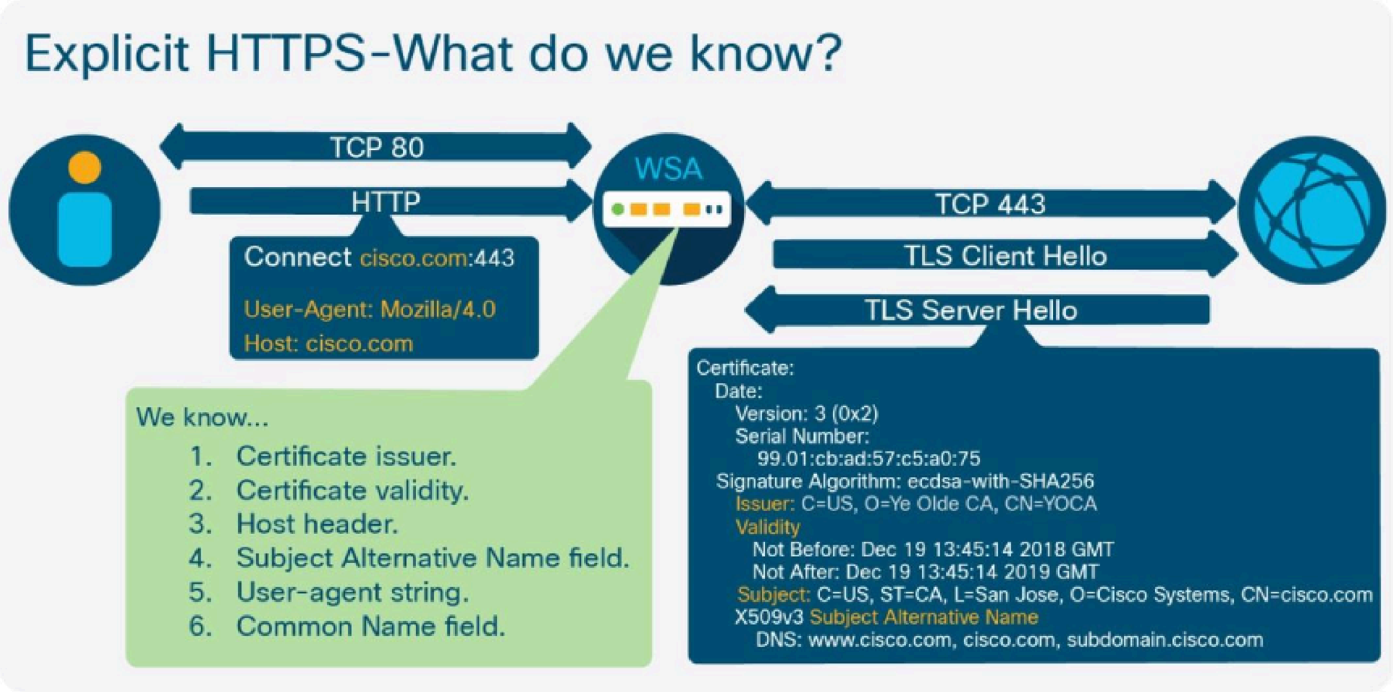


HTTPS policy operations

- Drop
  - Connection is closed.
- Decrypt
  - Traffic is decrypted and evaluated by access policies.
- Passthrough
  - Transaction is not decrypted.
  - Client negotiates directly with server.
- Monitor
  - No action taken.
  - Move to the next column on the policy.

The only case in which the SWA does not perform any TLS handshake is when the server name or IP address is present in a custom category, which is set to passthrough, and the server name is available in either an HTTP CONNECT or a TLS Client Hello. In an explicit scenario, the client provides the hostname of the server to the proxy prior to the TLS session initiation (in the host header), so this field is checked against the custom category. In a transparent deployment, the SWA check the **Server Name Indication (SNI)** field in the TLS Client Hello message and evaluate it against the custom category. If the host header or SNI is not present, the SWA must continue the handshake with the server in order to check the **Subject Alternative Name (SAN)** and **Common Name (CN)** fields on the certificate, in that order.

What this behavior means for policy design is that the number of TLS handshakes can be reduced by determining well-known and internally trusted servers and setting them to passthrough from custom category list, rather than relying on web category and reputation score, which still require the SWA to complete a TLS handshake with the server. However, it is important to note that this also prevents certificate validity checks.



Explicit HTTPS-What do we know?

TCP 80

HTTP

WSA

TCP 443

TLS Client Hello

TLS Server Hello

Connect cisco.com:443

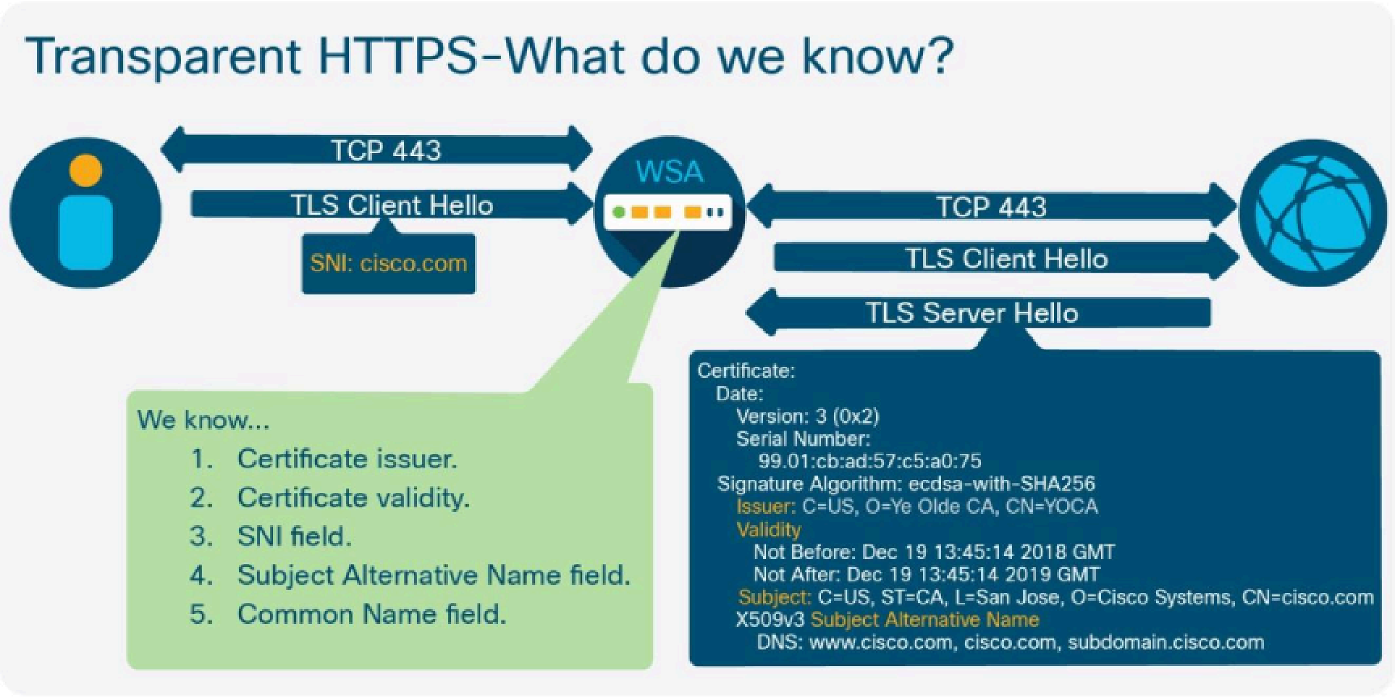
User-Agent: Mozilla/4.0

Host: cisco.com

We know...

1. Certificate issuer.
2. Certificate validity.
3. Host header.
4. Subject Alternative Name field.
5. User-agent string.
6. Common Name field.

```
Certificate:
  Date:
    Version: 3 (0x2)
    Serial Number:
      99.01:cb:ad:57:c5:a0:75
  Signature Algorithm: ecdsa-with-SHA256
    Issuer: C=US, O=Ye Olde CA, CN=YOCA
    Validity
      Not Before: Dec 19 13:45:14 2018 GMT
      Not After: Dec 19 13:45:14 2019 GMT
    Subject: C=US, ST=CA, L=San Jose, O=Cisco Systems, CN=cisco.com
    X509v3 Subject Alternative Name
      DNS: www.cisco.com, cisco.com, subdomain.cisco.com
```



We know...

1. Certificate issuer.
2. Certificate validity.
3. SNI field.
4. Subject Alternative Name field.
5. Common Name field.

```
Certificate:
  Date:
    Version: 3 (0x2)
    Serial Number:
      99.01:cb:ad:57:c5:a0:75
  Signature Algorithm: ecdsa-with-SHA256
    Issuer: C=US, O=Ye Olde CA, CN=YOCA
    Validity
      Not Before: Dec 19 13:45:14 2018 GMT
      Not After: Dec 19 13:45:14 2019 GMT
    Subject: C=US, ST=CA, L=San Jose, O=Cisco Systems, CN=cisco.com
    X509v3 Subject Alternative Name
      DNS: www.cisco.com, cisco.com, subdomain.cisco.com
```

Given the speed at which new sites appear on web, it is likely that a number of sites are found uncategorized by the web reputation and categorizations databases used by the SWA. This does not indicate that the site is necessarily more likely to be malicious, and additionally all of these sites still be subjected to AV scanning, AMP file reputation and analysis, and any object blocking or scanning that is configured. For these reasons, it is not recommended to drop uncategorized sites in most circumstances. It is best to set them to be decrypted and scanned by the AV engines and evaluated by AVC, AMP, access policies, and so on. There is more information about uncategorized sites in the **Access Policies** section.

Access Policies

As with the ID profile, the criteria set in the decryption policy is also evaluated as a logical AND with one important exception when information from the ISE is used. The policy-matching behavior is explained next, based on what elements are configured (AD group, user, or SGT):

- AD groups and users—No change to previous behavior; the policy is matched if the user is a member of group, OR the user is specified in the policy.
- SGT and AD groups and users—The policy is matched if the user is associated with the SGT AND is a member of the AD group, OR the user is specified in the policy.
- SGT and users—The policy is matched if the user is associated with the SGT OR the user is specified in the policy.

HTTP traffic is evaluated against the access policies immediately after being authenticated. HTTPS traffic is evaluated after being authenticated, and if the decrypt action is applied per the matching decryption policy. For decrypted requests, there are two access_log entries. The first log entry shows the action applied to the initial TLS connection (decrypt), and a second log entry shows the action applied by the access policy to the decrypted HTTP request.

As explained in the **Web Proxy** section, range request headers are used to request a specific byte range of a file and are commonly used by OS and application update services. The SWA, by default, strips these headers from outbound requests, because without the entire file, it is impossible to perform malware scanning or to utilize AVC features. If many hosts on the network are requesting small byte ranges frequently to retrieve updates, this can trigger the SWA to download the entire file several times simultaneously. This can quickly exhaust available internet bandwidth and cause service outages. The most common causes of this failure scenario are Microsoft Windows update and Adobe software update daemons.

To mitigate this, the best solution is to steer this traffic around the SWA altogether. This is not always feasible for transparently deployed environments, and in these cases, the next best option is to create dedicated access policies for the traffic and enable range request header forwarding on those policies. It must be considered that AV scanning and AVC is not possible for these requests, and so the policies must be carefully designed to only target the intended traffic. Often, the best way to accomplish this is by matching the user-agent string found in the request header. The user-agent string for common update daemons can be found online, or the requests can be captured by an administrator and examined. Most update services, including Microsoft Windows and Adobe software updates, are not using HTTPS.

As is described in the **Decryption Policies** section, it is not recommended to drop uncategorized sites in the decryption policies. For the same reasons, it is not recommended to block them in the access policies. The Dynamic Content Analysis (DCA) engine can use the content of a given site, along with other heuristic data to categorized sites that otherwise would be marked uncategorized by URL database lookups. Enabling this feature reduces the number of uncategorized verdicts in the SWA.

The Object Scanning settings of an access policy provide the ability to inspect several types of archive files. If the network regularly downloads archive files as part of application updates, enabling archive file inspection can significantly increase CPU usage. This traffic must be identified ahead of time and exempted if the intention is to inspect all archive files. The first place to investigate possible methods to identify this traffic is the user-agent string, as this can help avoid IP allowed-lists that can become cumbersome to maintain.

Custom and External URL Categories

The Custom category lists are used to identify a server by IP address or hostname. It is possible to use regular expressions (regex) to specify patterns by which server names can be matched. It is much more resource-intensive to use a regex pattern to match a server name than it is to use a substring match, and so they must only be used when absolutely necessary. A "." can be added to the beginning of a domain name in order to match a subdomain without the need for regex. For example, ".cisco.com" also match "www.cisco.com."

As explained in the **Complexity** section, low complexity is defined as ten custom category lists, medium complexity as twenty, and high complexity as thirty. It is recommended to keep this number under twenty, especially if the lists use regex patterns or contain a large number of entries. Refer to the **Access Policies** section for additional details on the number of entries for each type.

External URL feeds are much more flexible than static custom category lists, and leveraging them can have a direct impact on security because they remove the need for an administrator to manually maintain them. Because this feature can be used to retrieve lists that are not maintained or controlled by the SWA administrator, the ability to add individual exceptions to the downloaded addresses was added in AsyncOS version 11.8.

The Office365 API is especially useful to make policy decisions on this commonly deployed service and can be leveraged for individual applications (PowerPoint, Skype, Word, and so on). Microsoft recommends bypassing proxies for all Office365 traffic to optimize performance. Microsoft documentation states:

Caution: “While SSL Break and Inspect creates the largest latency, other services such as proxy authentication and reputation lookup can cause poor performance and a bad user experience. Additionally, these perimeter network devices need enough capacity to process all of the network connection requests. We recommend bypassing your proxy or inspection devices for direct Office 365 network requests.” - https://learn.microsoft.com/en-us/microsoft-365/enterprise/managing-office-365-endpoints?view=o365-worldwide

It can be difficult to use this guidance in a transparent proxy environment. Beginning in AsyncOS version 11.8, it is possible to use the dynamic category list retrieved from the Office365 API to populate the bypass settings list. This list is used to send transparently redirected traffic back to the WCCP device for direct routing.

Bypassing all Office365 traffic creates a blind spot for administrators who require some basic security controls and reporting for this traffic. If Office365 traffic is not bypassed by the SWA, it is important to understand the specific technical challenges that can arise. One of these is the number of connections that are required by the applications. Sizing must be appropriately adjusted to accommodate the additional persistent TCP connections required by Office365 applications. This can increase the total connection count by between ten and fifteen persistent TCP sessions per user.

The decrypt and re-encrypt actions performed by the HTTPS proxy introduce a small amount of latency to the connections. Office365 applications can be very sensitive to latency, and if other factors such as slow WAN connection and disparate geographic location compound this, user experience can suffer.

Some Office365 applications employ proprietary TLS parameters that prevent the HTTPS proxy from completing a handshake with the application server. This is required to validate the certificate or retrieve the hostname. When this is combined with an application such as Skype for Business that does not send a **Server Name Indication (SNI)** field in its TLS Client Hello message, it becomes necessary to bypass this traffic entirely. AsyncOS 11.8 has introduced the ability to bypass traffic based on destination IP address only, without certificate checks to address this scenario.

Monitors and Alerts

CLI Monitors

The SWA CLI provides commands for real-time monitoring of important processes. Most useful are the commands that show statistics related to the prox process. The **status detail** command is a good source for a summary of resource usage and performance metrics, including uptime, bandwidth used, response latency, number of connections, and more. Here is example output from this command:

```
SWA_CLI>  status detail

Status as of:                  Fri Nov 11 14:06:52 2022 +03
Up since:                      Fri Apr 08 10:15:00 2022 +03 (217d 3h 51m 52s)
System Resource Utilization:
  CPU                                     3.3%
  RAM                                     6.2%
  Reporting/Logging Disk                 45.6%
Transactions per Second:
  Average in last minute                    55
  Maximum in last hour                     201
  Average in last hour                      65
  Maximum since proxy restart             1031
  Average since proxy restart               51
Bandwidth (Mbps):
  Average in last minute                 4.676
  Maximum in last hour                 327.258
  Average in last hour                  10.845
  Maximum since proxy restart         1581.297
  Average since proxy restart           11.167
Response Time (ms):
  Average in last minute                   635
  Maximum in last hour                  376209
  Average in last hour                     605
  Maximum since proxy restart          2602943
  Average since proxy restart              701
Cache Hit Rate:
  Average in last minute                     0
  Maximum in last hour                       2
  Average in last hour                       0
  Maximum since proxy restart               15
  Average since proxy restart                0
Connections:
  Idle client connections                  186
  Idle server connections                  184
  Total client connections                3499
  Total server connections                3632
SSLJobs:
  In queue Avg in last minute                4
  Average in last minute                 45214
  SSLInfo Average in last min               94
Network Events:
  Average in last minute                   0.0
  Maximum in last minute                    35
  Network events in last min            124502
```

The **rate** command shows real-time information about the percentage of CPU used by the prox process, as well as the number of Requests Per Second (RPS) and cache statistics. This command continues to poll and display new output until interrupted. This is an example of output from this command:

```
SWA_CLI>  rate

Press Ctrl-C to stop.
  %proxy  reqs                         client    server    %bw  disk  disk
     CPU  /sec   hits blocks misses    kb/sec    kb/sec  saved   wrs   rds
  5.00    51      1    147    370      2283      2268    0.6    48    37
  4.00    36      0    128    237     21695     21687    0.0    47    38
  4.00    48      2    179    307      8168      8154    0.2    65    33
  5.00    53      0    161    372      2894      2880    0.5    48    32
  6.00    52      0    198    328     15110     15100    0.1    63    33
  6.00    77      0    415    363      4695      4684    0.2    48    34
  7.00    85      1    417    433      5270      5251    0.4    49    35
  7.00    67      1    443    228      2242      2232    0.5    85    44
```

The **tcpservices** command displays information about selected process listening ports. An explanation of each process and the address and port combination is also displayed:

```
SWA_CLI> tcpservices

System Processes (Note: All processes may not always be present)
  ftpd.main    - The FTP daemon
  ginetd       - The INET daemon
  interface    - The interface controller for inter-process communication
  ipfw         - The IP firewall
  slapd        - The Standalone LDAP daemon
  sntpd        - The SNTP daemon
  sshd         - The SSH daemon
  syslogd      - The system logging daemon
  winbindd     - The Samba Name Service Switch daemon

Feature Processes
  coeuslogd    - Main WSA controller
  gui          - GUI process
  hermes       - Mail server for sending alerts, etc.
  java         - Processes for storing and querying Web Tracking data
  musd         - AnyConnect Secure Mobility server
  pacd         - PAC file hosting daemon
  prox         - WSA proxy
  trafmon      - L4 Traffic Monitor
  uds          - User Discovery System (Transparent Auth)
  wccpd        - WCCP daemon

COMMAND      USER         TYPE NODE   NAME
connector    root         IPv4 TCP    127.0.0.1:8823
java         root         IPv6 TCP    [::127.0.0.1]:18081
hybridd      root         IPv4 TCP    127.0.0.1:8833
gui          root         IPv4 TCP    172.16.40.80:8443
ginetd       root         IPv4 TCP    172.16.40.80:ssh
nginx        root         IPv6 TCP    *:4431
nginx        root         IPv4 TCP    127.0.0.1:8843
nginx        nobody       IPv6 TCP    *:4431
nginx        nobody       IPv4 TCP    127.0.0.1:8843
nginx        nobody       IPv6 TCP    *:4431
nginx        nobody       IPv4 TCP    127.0.0.1:8843
api_serve    root         IPv4 TCP    172.16.40.80:6080
api_serve    root         IPv4 TCP    127.0.0.1:60001
api_serve    root         IPv4 TCP    172.16.40.80:6443
chimera      root         IPv4 TCP    127.0.0.1:6380
nectar       root         IPv4 TCP    127.0.0.1:6382
redis-ser    root         IPv4 TCP    127.0.0.1:6383
redis-ser    root         IPv4 TCP    127.0.0.1:6379
prox         root         IPv4 TCP    127.0.0.1:http
prox         root         IPv6 TCP    [::1]:http
prox         root         IPv4 TCP    172.16.11.69:http
prox         root         IPv4 TCP    172.16.11.68:http
prox         root         IPv4 TCP    172.16.11.252:http
prox         root         IPv4 TCP    127.0.0.1:3128
prox         root         IPv6 TCP    [::1]:3128
prox         root         IPv4 TCP    172.16.11.69:3128
prox         root         IPv4 TCP    172.16.11.68:3128
prox         root         IPv4 TCP    172.16.11.252:3128
prox         root         IPv4 TCP    127.0.0.1:https
prox         root         IPv6 TCP    [::1]:https
prox         root         IPv4 TCP    172.16.11.69:https
prox         root         IPv4 TCP    172.16.11.68:https
prox         root         IPv4 TCP    172.16.11.252:https
prox         root         IPv4 TCP    127.0.0.1:http
prox         root         IPv6 TCP    [::1]:http
prox         root         IPv4 TCP    172.16.11.69:http
prox         root         IPv4 TCP    172.16.11.68:http
prox         root         IPv4 TCP    172.16.11.252:http
prox         root         IPv4 TCP    127.0.0.1:3128
prox         root         IPv6 TCP    [::1]:3128
prox         root         IPv4 TCP    172.16.11.69:3128
prox         root         IPv4 TCP    172.16.11.68:3128
prox         root         IPv4 TCP    172.16.11.252:3128
prox         root         IPv4 TCP    127.0.0.1:https
prox         root         IPv6 TCP    [::1]:https
prox         root         IPv4 TCP    172.16.11.69:https
prox         root         IPv4 TCP    172.16.11.68:https
prox         root         IPv4 TCP    172.16.11.252:https
prox         root         IPv4 TCP    127.0.0.1:25255
prox         root         IPv4 TCP    127.0.0.1:socks
prox         root         IPv6 TCP    [::1]:socks
prox         root         IPv4 TCP    172.16.11.69:socks
prox         root         IPv4 TCP    172.16.11.68:socks
prox         root         IPv4 TCP    172.16.11.252:socks
prox         root         IPv4 TCP    127.0.0.1:ftp-proxy
prox         root         IPv6 TCP    [::1]:ftp-proxy
prox         root         IPv4 TCP    172.16.11.69:ftp-proxy
prox         root         IPv4 TCP    172.16.11.68:ftp-proxy
prox         root         IPv4 TCP    172.16.11.252:ftp-proxy
prox         root         IPv4 TCP    127.0.0.1:http
prox         root         IPv6 TCP    [::1]:http
prox         root         IPv4 TCP    172.16.11.69:http
prox         root         IPv4 TCP    172.16.11.68:http
prox         root         IPv4 TCP    172.16.11.252:http
prox         root         IPv4 TCP    127.0.0.1:3128
prox         root         IPv6 TCP    [::1]:3128
prox         root         IPv4 TCP    172.16.11.69:3128
prox         root         IPv4 TCP    172.16.11.68:3128
prox         root         IPv4 TCP    172.16.11.252:3128
prox         root         IPv4 TCP    127.0.0.1:https
prox         root         IPv6 TCP    [::1]:https
prox         root         IPv4 TCP    172.16.11.69:https
prox         root         IPv4 TCP    172.16.11.68:https
prox         root         IPv4 TCP    172.16.11.252:https
prox         root         IPv4 TCP    127.0.0.1:25256
prox         root         IPv4 TCP    127.0.0.1:http
prox         root         IPv6 TCP    [::1]:http
prox         root         IPv4 TCP    172.16.11.69:http
prox         root         IPv4 TCP    172.16.11.68:http
prox         root         IPv4 TCP    172.16.11.252:http
prox         root         IPv4 TCP    127.0.0.1:3128
prox         root         IPv6 TCP    [::1]:3128
prox         root         IPv4 TCP    172.16.11.69:3128
prox         root         IPv4 TCP    172.16.11.68:3128
prox         root         IPv4 TCP    172.16.11.252:3128
prox         root         IPv4 TCP    127.0.0.1:https
prox         root         IPv6 TCP    [::1]:https
prox         root         IPv4 TCP    172.21.11.69:https
prox         root         IPv4 TCP    172.21.11.68:https
prox         root         IPv4 TCP    172.21.11.252:https
prox         root         IPv4 TCP    127.0.0.1:25257
smart_age    root         IPv6 TCP    [::127.0.0.1]:65501
smart_age    root         IPv6 TCP    [::127.0.0.1]:28073
interface    root         IPv4 TCP    127.0.0.1:domain
stunnel      root         IPv4 TCP    127.0.0.1:32137
```

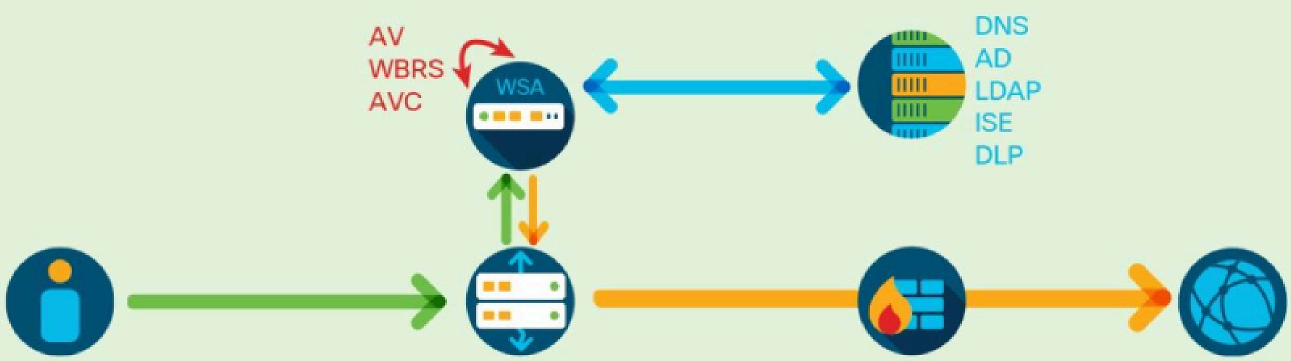
Logging

Web traffic is highly dynamic and varied. After a proxy deployment is complete, it is important to regularly reassess the quantity and makeup of traffic being passed through the appliance. You must check the percentage of decrypted traffic on a regular basis (once a quarter) to ensure the size is consistent with the expectations and specifications of the initial installation. This can be done with a log management product such as **Advanced Web Security Reporting (AWSR)** or with simple Bash or PowerShell commands with the access logs. The number of RPS must also be reassessed on a regular basis to ensure that the appliance has enough overhead to account for spikes in traffic and possible failover in a high-availability, load-balanced configuration.

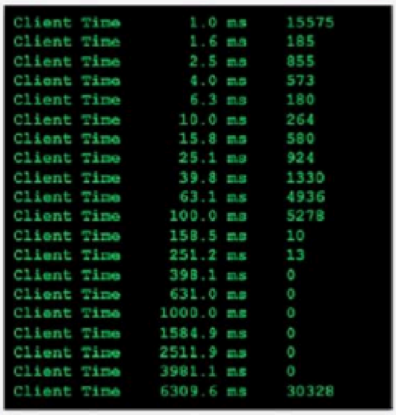
The track_stats log is appended every five minutes and includes several sections of output directly related to the prox process and its objects in memory. Most useful in performance monitoring are the sections that show the average latency for various request processes, include DNS lookup time, AV engine scan time, and many more useful fields. This log is not configurable from the GUI or the CLI and is only accessible through Secure Copy Protocol (SCP) or File Transfer Protocol (FTP). This is the most important log to have when troubleshooting performance so it must be polled frequently.

Where can latency be introduced?

- Client Side
- External Services
- Internal Services
- Server Side



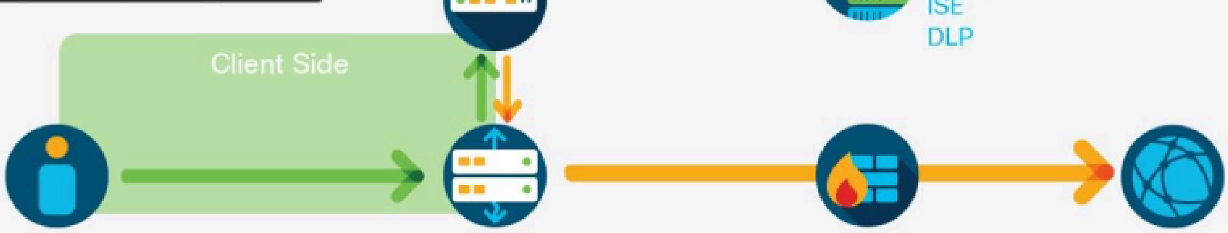
Client side latency



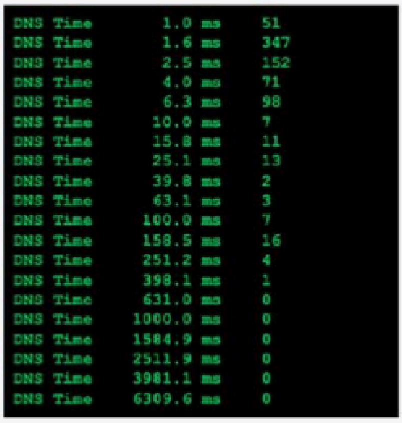
```
Client Time       1.0 ms    15575
Client Time       1.6 ms    185
Client Time       2.5 ms    855
Client Time       4.0 ms    573
Client Time       6.3 ms    180
Client Time      10.0 ms    264
Client Time      15.8 ms    580
Client Time      25.1 ms    924
Client Time      39.8 ms    1330
Client Time      63.1 ms    4936
Client Time     100.0 ms    5278
Client Time     158.5 ms    10
Client Time     251.2 ms    13
Client Time     398.1 ms    0
Client Time     631.0 ms    0
Client Time    1000.0 ms    0
Client Time    1584.9 ms    0
Client Time    2511.9 ms    0
Client Time    3981.1 ms    0
Client Time    6309.6 ms    30328
```

- **“Client Time”** in **track_stats** log.
- The amount of time in milliseconds that the client was waiting for a response.
- May indicate an upstream issues-keep investigating!
- Access logs can show this in custom field **% : 1>**

%:1>	x-p2c-first-byte-time	Wait-time for first byte written to client



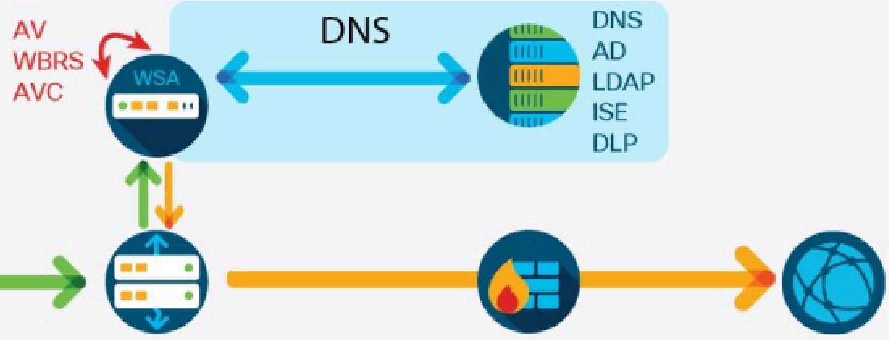
DNS latency



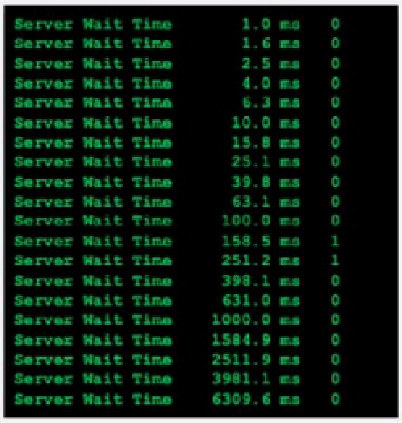
```
DNS Time       1.0 ms    51
DNS Time       1.6 ms    347
DNS Time       2.5 ms    152
DNS Time       4.0 ms    71
DNS Time       6.3 ms    98
DNS Time      10.0 ms    7
DNS Time      15.8 ms    11
DNS Time      25.1 ms    13
DNS Time      39.8 ms    2
DNS Time      63.1 ms    3
DNS Time     100.0 ms    7
DNS Time     158.5 ms    16
DNS Time     251.2 ms    4
DNS Time     398.1 ms    1
DNS Time     631.0 ms    0
DNS Time    1000.0 ms    0
DNS Time    1584.9 ms    0
DNS Time    2511.9 ms    0
DNS Time    3981.1 ms    0
DNS Time    6309.6 ms    0
```

- The amount of time in milliseconds that the WSA waited for a DNS response.
- Calls for investigation for your DNS resolvers (or path to them).
- **access logs** can show this in custom field **% : >d**

%:>d	x-p2p-dns-svc-time	Time taken by the Web Proxy DNS Process to send a DNS result to ehe Web proxy.



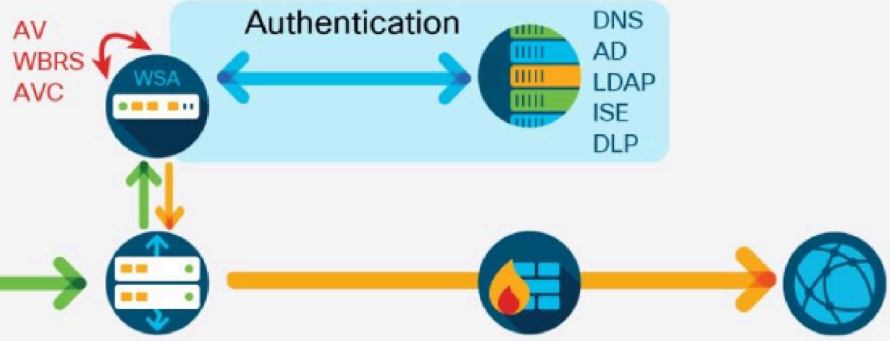
Authentication latency



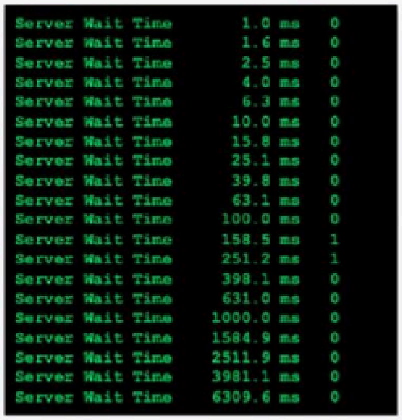
```
Server Wait Time       1.0 ms    0
Server Wait Time       1.6 ms    0
Server Wait Time       2.5 ms    0
Server Wait Time       4.0 ms    0
Server Wait Time       6.3 ms    0
Server Wait Time      10.0 ms    0
Server Wait Time      15.8 ms    0
Server Wait Time      25.1 ms    0
Server Wait Time      39.8 ms    0
Server Wait Time      63.1 ms    0
Server Wait Time     100.0 ms    0
Server Wait Time     158.5 ms    1
Server Wait Time     251.2 ms    1
Server Wait Time     398.1 ms    0
Server Wait Time     631.0 ms    0
Server Wait Time    1000.0 ms    0
Server Wait Time    1584.9 ms    0
Server Wait Time    2511.9 ms    0
Server Wait Time    3981.1 ms    0
Server Wait Time    6309.6 ms    0
```

- There are two metrics: **"Auth Helper Wait Time"** and **"Auth Helper Service Wait Time."**
- Use the first to get pure auth time without the request time added.
- **access logs** can show this in custom field **% : >a**

%:<a	x-p2p-auth-wait-time	Wait-time to receive the response from the Web Proxy authentication process, after the Web Proxy sent the request.



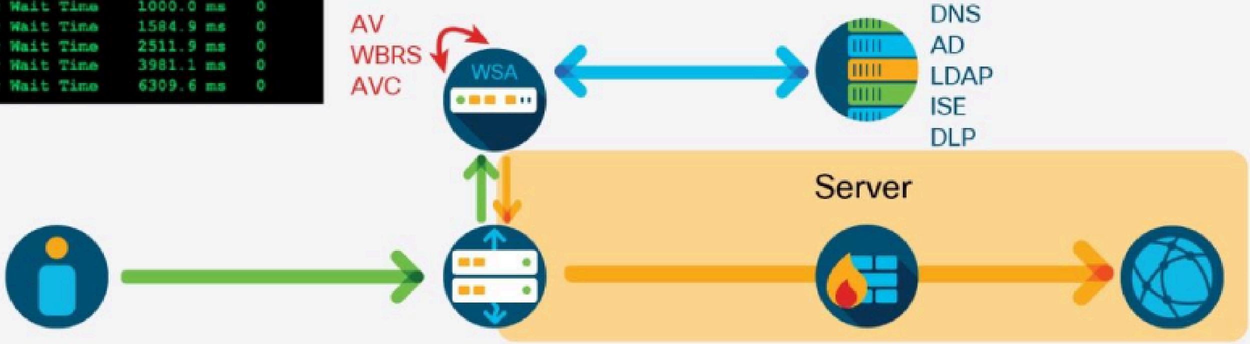
Server latency-wait time



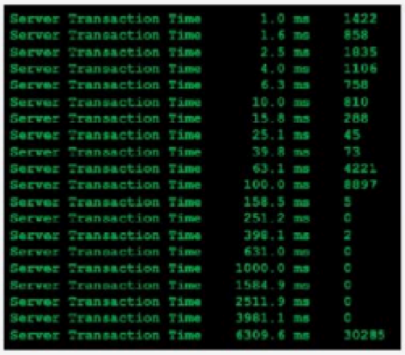
```
Server Wait Time       1.0 ms   0
Server Wait Time       1.6 ms   0
Server Wait Time       2.5 ms   0
Server Wait Time       4.0 ms   0
Server Wait Time       6.3 ms   0
Server Wait Time      10.0 ms   0
Server Wait Time      15.8 ms   0
Server Wait Time      25.1 ms   0
Server Wait Time      39.8 ms   0
Server Wait Time      63.1 ms   0
Server Wait Time     100.0 ms   0
Server Wait Time     158.5 ms   1
Server Wait Time     251.2 ms   1
Server Wait Time     398.1 ms   0
Server Wait Time     631.0 ms   0
Server Wait Time    1000.0 ms   0
Server Wait Time    1584.9 ms   0
Server Wait Time    2511.9 ms   0
Server Wait Time    3981.1 ms   0
Server Wait Time    6309.6 ms   0
```

- The amount of time in milliseconds that the WSA waited for the first byte of the server response.
- Calls for investigation of your upstream devices and WAN connection.
- **access logs** can show this in custom field **% : >1**

%:>1	x-s2p-first-byte-time	Wait-time for first response byte from server

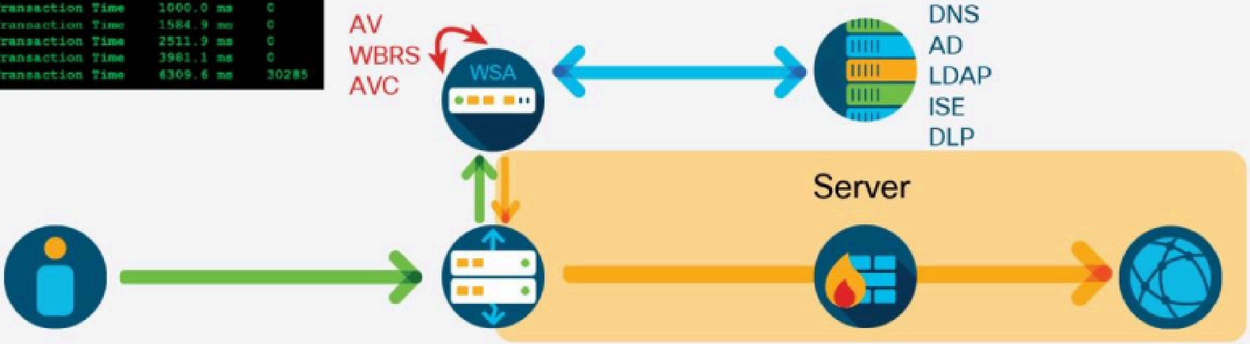


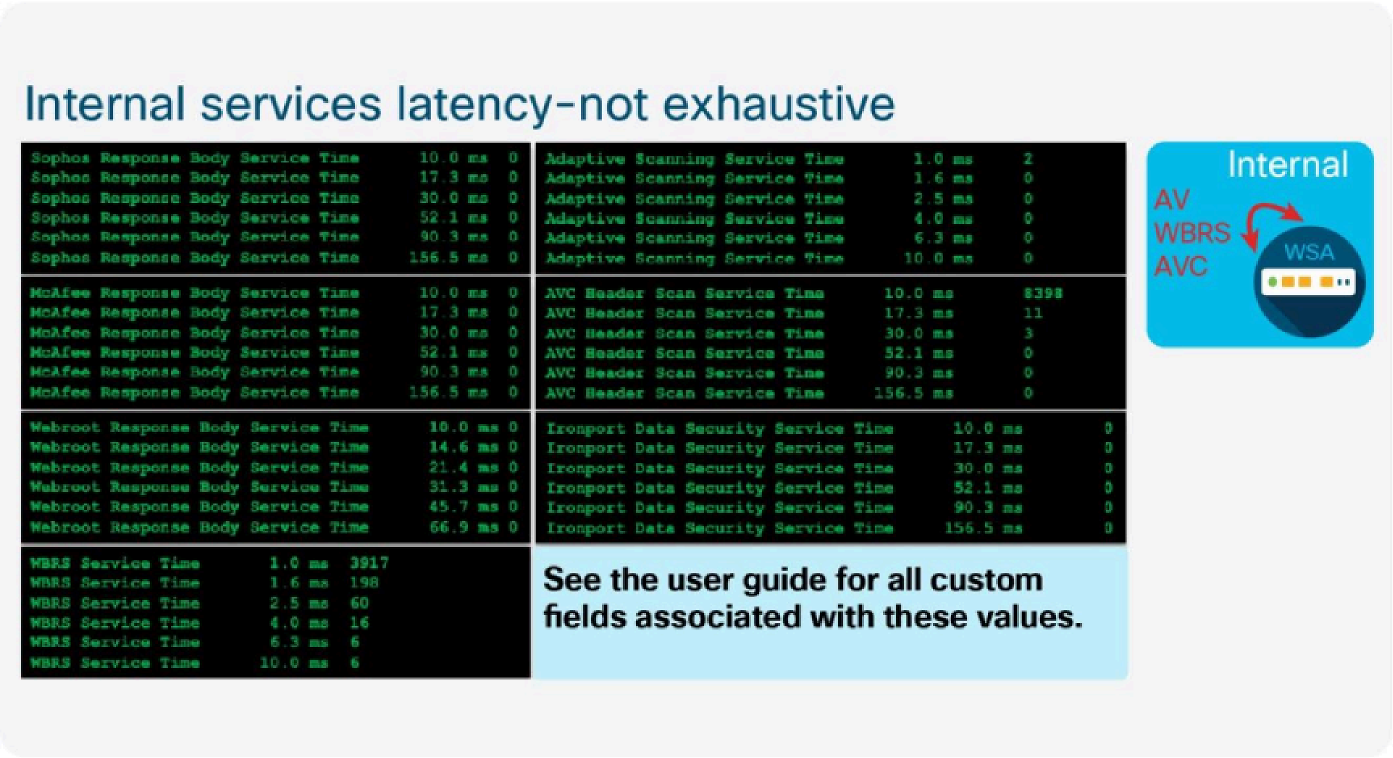
Server latency-transaction time



```
Server Transaction Time       1.0 ms    1422
Server Transaction Time       1.6 ms    858
Server Transaction Time       2.5 ms    1035
Server Transaction Time       4.0 ms    1106
Server Transaction Time       6.3 ms    758
Server Transaction Time      10.0 ms    810
Server Transaction Time      15.8 ms    288
Server Transaction Time      25.1 ms    45
Server Transaction Time      39.8 ms    73
Server Transaction Time      63.1 ms    4221
Server Transaction Time     100.0 ms    8897
Server Transaction Time     158.5 ms    5
Server Transaction Time     251.2 ms    0
Server Transaction Time     398.1 ms    2
Server Transaction Time     631.0 ms    0
Server Transaction Time    1000.0 ms    0
Server Transaction Time    1584.9 ms    0
Server Transaction Time    2511.9 ms    0
Server Transaction Time    3981.1 ms    0
Server Transaction Time    6309.6 ms    30285
```

- The amount of time in milliseconds for the entire server-side transaction to complete.
- Calls for investigation of your upstream devices and WAN connection.
- No **access logs** custom field, but can be determined by a combinationof them.





Internal services latency-not exhaustive

```
Sophos Response Body Service Time       10.0 ms  0
Sophos Response Body Service Time       17.3 ms  0
Sophos Response Body Service Time       30.0 ms  0
Sophos Response Body Service Time       52.1 ms  0
Sophos Response Body Service Time       90.3 ms  0
Sophos Response Body Service Time      156.5 ms  0
```

```
Adaptive Scanning Service Time       1.0 ms     2
Adaptive Scanning Service Time       1.6 ms     0
Adaptive Scanning Service Time       2.5 ms     0
Adaptive Scanning Service Time       4.0 ms     0
Adaptive Scanning Service Time       6.3 ms     0
Adaptive Scanning Service Time      10.0 ms     0
```

```
McAfee Response Body Service Time       10.0 ms  0
McAfee Response Body Service Time       17.3 ms  0
McAfee Response Body Service Time       30.0 ms  0
McAfee Response Body Service Time       52.1 ms  0
McAfee Response Body Service Time       90.3 ms  0
McAfee Response Body Service Time      156.5 ms  0
```

```
AVC Header Scan Service Time     10.0 ms      8398
AVC Header Scan Service Time     17.3 ms      11
AVC Header Scan Service Time     30.0 ms      3
AVC Header Scan Service Time     52.1 ms      0
AVC Header Scan Service Time     90.3 ms      0
AVC Header Scan Service Time    156.5 ms      0
```

```
Webroot Response Body Service Time      10.0 ms 0
Webroot Response Body Service Time      14.6 ms 0
Webroot Response Body Service Time      21.4 ms 0
Webroot Response Body Service Time      31.3 ms 0
Webroot Response Body Service Time      45.7 ms 0
Webroot Response Body Service Time      66.9 ms 0
```

```
Ironport Data Security Service Time      10.0 ms        0
Ironport Data Security Service Time      17.3 ms        0
Ironport Data Security Service Time      30.0 ms        0
Ironport Data Security Service Time      52.1 ms        0
Ironport Data Security Service Time      90.3 ms        0
Ironport Data Security Service Time     156.5 ms        0
```

```
WBRS Service Time       1.0 ms  3917
WBRS Service Time       1.6 ms  198
WBRS Service Time       2.5 ms  60
WBRS Service Time       4.0 ms  16
WBRS Service Time       6.3 ms  6
WBRS Service Time      10.0 ms  6
```

See the user guide for all custom fields associated with these values.

An individual SHD log line is written every 60 seconds and contains many fields that are important for performance monitoring, including latency, RPS, and total client-side and server-side connections. This is an example of an SHD log line:

```
Fri Nov 11 14:16:42 2022 Info: Status: CPULd 2.4 DskUtil 45.7 RAMUtil 6.7 Reqs 62 Band 11383 Latency 61
Fri Nov 11 14:17:42 2022 Info: Status: CPULd 2.6 DskUtil 45.7 RAMUtil 6.7 Reqs 55 Band 10532 Latency 77
Fri Nov 11 14:18:43 2022 Info: Status: CPULd 1.9 DskUtil 45.7 RAMUtil 6.6 Reqs 48 Band 7285 Latency 579
Fri Nov 11 14:19:43 2022 Info: Status: CPULd 2.3 DskUtil 45.7 RAMUtil 6.6 Reqs 52 Band 34294 Latency 79
Fri Nov 11 14:20:43 2022 Info: Status: CPULd 2.4 DskUtil 45.7 RAMUtil 6.7 Reqs 55 Band 8696 Latency 691
Fri Nov 11 14:21:43 2022 Info: Status: CPULd 2.3 DskUtil 45.7 RAMUtil 6.7 Reqs 49 Band 7064 Latency 140
Fri Nov 11 14:22:43 2022 Info: Status: CPULd 1.9 DskUtil 45.7 RAMUtil 6.8 Reqs 41 Band 5444 Latency 788
Fri Nov 11 14:23:43 2022 Info: Status: CPULd 2.2 DskUtil 45.7 RAMUtil 6.8 Reqs 48 Band 6793 Latency 820
Fri Nov 11 14:24:44 2022 Info: Status: CPULd 2.3 DskUtil 45.7 RAMUtil 6.7 Reqs 44 Band 8735 Latency 673
Fri Nov 11 14:25:44 2022 Info: Status: CPULd 2.4 DskUtil 45.7 RAMUtil 6.7 Reqs 53 Band 8338 Latency 731
```

Additional custom fields can be added to the access_logs that denote latency information for individual requests. These fields include server response, DNS resolution, and AV scanner latency. The fields must be added to the log to glean valuable information to be used for troubleshoot. This is the recommended custom field string for use:

```
[ Request Details: ID = %I, User Agent = %u, AD Group Memberships = ( %m ) %g ] [ Tx Wait Times (in ms)
```

Performance information derived from these values is as follow:

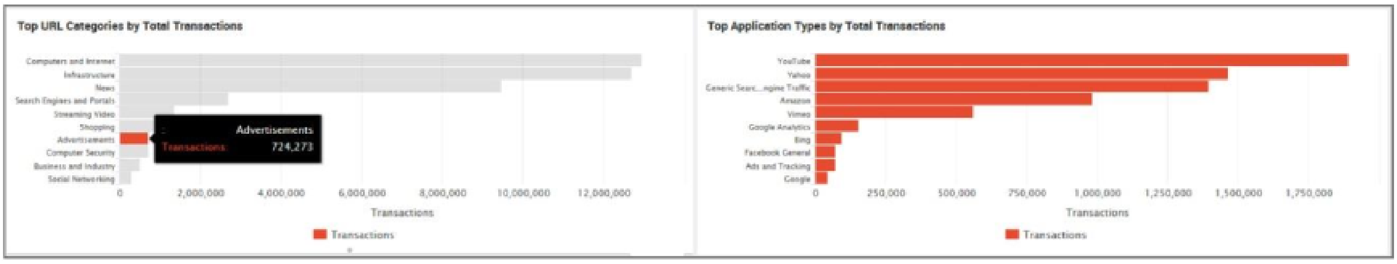
Custom field	Description
%:<a	Wait time to receive the response from the web proxy authentication process, after the Web Proxy sent the request.

%:<b	Wait time to write request body to server after header.
%:<d	Wait time to receive the response from the web proxy DNS process, after the Web Proxy sent the request.
%:<h	Wait time to write request header to server after first byte.
%:<r	Wait time to receive the response from the web reputation filters, after the web proxy sent the request.
%:<s	Wait time to receive the verdict from the web proxy antispyware process, after the web proxy sent the request.
%:>	Wait time for first response byte from server.
%:>a	Wait time to receive the response from the web proxy authentication process, includes the time required for the web proxy to send the request.
%:>b	Wait time for complete response body after header received.
%:>c	Time required for the web proxy to read a response from the disk cache.
%:>d	Wait time to receive the response from the web proxy DNS process, includes the time required for the web proxy to send the request.
%:>h	Wait time for server header after first response byte.
%:>r	Wait time to receive the verdict from the web reputation filters, includes the time required for the web proxy to send the request.
%:>s	Wait time to receive the verdict from the web proxy antispyware process, includes the time required for the web proxy to send the request.
%:1<	Wait time for first request byte from new client connection.
%:1>	Wait time for first byte written to client.
%:b<	Wait time for complete client body.
%:b>	Wait time for complete body written to client.
%:e>	Wait time to receive the response from the AMP scanning engine, after web proxy sent the request.
%:e<	Wait time to receive the verdict from the AMP scanning engine, includes the time required for the web proxy to send the request.
%:h<	Wait time for complete client header after first byte.
%:h>	Wait time for complete header written to client.
%:m<	Wait time to receive the verdict from the McAfee scanning engine, includes the time required for the web proxy to send the request.
%:m>	Wait time to receive the response from the McAfee scanning engine, after the web proxy sent the request.
%F	Client source port.
%p	Web Server port.
%k	Data source IP address (Web Server IP Address).
%:w<	Wait time to receive the verdict from the Webroot scanning engine, includes the time required for the web proxy to send the request.
%:w>	Wait time to receive the response from the Webroot scanning engine, after the web proxy sent the request.

The SWA licensing model allows for the reuse of physical appliance licenses for virtual appliances. You can take advantage of this and deploy test SWAv appliances for use in lab environment. New features and configurations can be piloted this way to ensure stability and reliability without and at the same time not violate licensing terms.

Advanced Web Security Reporting (AWSR)

AWSR must be leveraged to make the most of reporting data from the SWA. Especially in environments where many SWAs are deployed, this solution is many times more scalable than utilizing centralized reporting on a **Security Management Appliance (SMA)** as well as provides custom reporting attributes that add an immense amount of depth and customization to the data. Reports can be grouped and customized to meet any organization’s need. Cisco Advanced Services group must be leveraged in sizing for AWSR.



Email Alerting

The built-in email alert system on the SWA is best leveraged as a base-line alert system. It must be tweaked appropriately to meet the needs of the administrator, as it can be very noisy if all informational events are enabled. It is more important to limit the alerts and actively monitor them than it is to alert on everything and ignore them as spam.

Alert Settings	**Configuration**
From Address to use when sending alerts	Automatically generated
Initial number of seconds to wait before sending a duplicate alert	300 Seconds
Maximum number of seconds to wait before sending a duplicate alert	3600 Seconds

Availability Monitoring

There are two methods that can be employed to monitor availability of a web proxy.

1. The first is **Layer 3 (L3)** monitoring, which tests whether the appliance IP address is reachable on the network. The simplest way to test this is to send an **ICMP Echo (ping)** request to the address at regular intervals and check for a Reply packet. The attributes of the reply, such as TTL, and latency can be parsed to determine the health of the network layer.
2. It is possible that a device can be responsive to pings but that the proxy processes are unresponsive or intermittent. Because of this, it is advisable to employ a **Layer 7 (L7)** monitor, which sends an explicit proxy request to the appliance and expects a **200 OK** HTTP response code. This tests not only the reachability of the network interface, but also the responsiveness of the proxy services and the viability of upstream services if an external resource is requested. This type of monitoring typically takes the form of an explicit HTTP **HEAD** request that asks the proxy to connect to a resource. The **HEAD** method requests the headers that would be returned must the client send a **GET** request, but includes only the response headers and no data.
   - If you use an **L7** monitoring tool or script, it is important to ensure that the traffic is exempted

from authentication. Otherwise this result in regular authentication failures and consumption of resources. When you use a custom user-agent string in the monitoring tool must be employed to identify the traffic. Even though the traffic is exempted from authentication, it can still be restricted from unnecessary internet access through the access policies.

When you use one or more of these methods, an administrator must establish a baseline of acceptable metrics around the proxy response and use that to build alert thresholds. You must dedicate time to gather the responses of such checks and before you decide how you decide how to configure the thresholds and the alert.

SNMP Monitoring

The **Simple Network Management Protocol (SNMP)** is the principle method for monitoring the health of the appliance. It can be used to receive alerts from the appliance (traps) or to poll various **Object Identifiers (OIDs)** to gather information. There are many OIDs available on the SWA that cover everything from hardware to resource usage to individual process information and request statistics.

There are a number of specific **Machine Information Base (MIB)** that must be monitored for both hardware and performance related reasons. The full list of MIBs can be found here: https://www.cisco.com/web/ironport/tools/web/asyncosweb-mib.txt.

This is a list of the recommended MIBs to monitor and not an exhaustive list:

Hardware OID	Name
1.3.6.1.4.1.15497.1.1.1.18.1.3	raidID
1.3.6.1.4.1.15497.1.1.1.18.1.2	raidStatus
1.3.6.1.4.1.15497.1.1.1.18.1.4	raidLastError
1.3.6.1.4.1.15497.1.1.1.10	fanTable
1.3.6.1.4.1.15497.1.1.1.9.1.2	degreesCelsius

This is OIDs map directly to the output of the **status detail** CLI command:

OID	Name	Status detail field
System Resources		
1.3.6.1.4.1.15497.1.1.1.2.0	perCentCPUUtilization	CPU
1.3.6.1.4.1.15497.1.1.1.1.0	perCentMemoryUtilization	RAM
Transactions per Second		
1.3.6.1.4.1.15497.1.2.3.7.1.1.0	cacheThruputNow	Average transactions per second in last minute.
1.3.6.1.4.1.15497.1.2.3.7.1.2.0	cacheThruput1hrPeak	Maximum transactions per second in last hour.
1.3.6.1.4.1.15497.1.2.3.7.1.3.0	cacheThruput1hrMean	Average transactions per second in last hour.
1.3.6.1.4.1.15497.1.2.3.7.1.8.0	cacheThruputLifePeak	Maximum transactions per second since proxy restart.

1.3.6.1.4.1.15497.1.2.3.7.1.9.0	cacheThruputLifeMean	Average transactions per second since proxy restart.
Bandwidth		
1.3.6.1.4.1.15497.1.2.3.7.4.1.0	cacheBwidthTotalNow	Average bandwidth in last minute.
1.3.6.1.4.1.15497.1.2.3.7.4.2.0	cacheBwidthTotal1hrPeak	Maximum bandwidth in last hour.
1.3.6.1.4.1.15497.1.2.3.7.4.3.0	cacheBwidthTotal1hrMean	Average bandwidth in last hour.
1.3.6.1.4.1.15497.1.2.3.7.4.8.0	cacheBwidthTotalLifePeak	Maximum bandwidth since proxy restart.
1.3.6.1.4.1.15497.1.2.3.7.4.9.0	cacheBwidthTotalLifeMean	Average bandwidth since proxy restart.
Response time		
1.3.6.1.4.1.15497.1.2.3.7.9.1.0	cacheHitsNow	Average cache hit rate in last minute.
1.3.6.1.4.1.15497.1.2.3.7.9.2.0	cacheHits1hrPeak	Maximum cache hit rate in last hour.
1.3.6.1.4.1.15497.1.2.3.7.9.3.0	cacheHits1hrMean	Average cache hit rate in last hour.
1.3.6.1.4.1.15497.1.2.3.7.9.8.0	cacheHitsLifePeak	Maximum cache hit rate since proxy restart.
1.3.6.1.4.1.15497.1.2.3.7.9.9.0	cacheHitsLifeMean	Average cache hit rate since proxy restart.
Cache hit rate		
1.3.6.1.4.1.15497.1.2.3.7.5.1.0	cacheHitsNow	Average cache hit rate in last minute.
1.3.6.1.4.1.15497.1.2.3.7.5.2.0	cacheHits1hrPeak	Maximum cache hit rate in last hour.
1.3.6.1.4.1.15497.1.2.3.7.5.3.0	cacheHits1hrMean	Average cache hit rate in last hour.
1.3.6.1.4.1.15497.1.2.3.7.5.8.0	cacheHitsLifePeak	Maximum cache hit rate since proxy restart.
1.3.6.1.4.1.15497.1.2.3.7.5.9.0	cacheHitsLifeMean	Average cache hit rate since proxy restart.
Connections		
1.3.6.1.4.1.15497.1.2.3.2.7.0	cacheClientIdleConns	Idle client connections.
1.3.6.1.4.1.15497.1.2.3.3.7.0	cacheServerIdleConns	Idle server connections.
1.3.6.1.4.1.15497.1.2.3.2.8.0	cacheClientTotalConns	Total client connections.
1.3.6.1.4.1.15497.1.2.3.3.8.0	cacheServerTotalConns	Total server connections.

Conclusion

This guide seeks to describe the most important aspects of SWA configuration, deployment, and monitoring. As a reference guide, its goal is to provide valuable information for those who wanted to ensure the most effective use of the SWA. The best practices described here are important for the stability, scalability, and efficacy of the device as a security tool. It also seeks to remain as a relevant resource goes forward and thus must be updated frequently to reflect changes in network environments and product feature sets.