Draft Standard

MEF 138 Draft (R1)

Security Functions for IP Services

November 2023

This draft represents MEF work in progress and is subject to change.

Table of Contents

List of Figures

List of Tables

1 List of Contributing Members

The following members of the MEF participated in the development of this Standard and have requested to be included in this list.

Editor Note 1: This list will be put together just before going to Letter Ballot

2 Abstract

This Standard specifies the requirements needed to add Security Functions to an IP Service [1].

This Standard defines the set of parameters that need to be agreed between the Subscriber and Service Provider for each Security Function. It also defines Security Functions that, when enabled, enforce the Policy on a per-Service Flow basis by performing any of the following actions: IP, Port and Protocol Filtering, DNS Protocol Filtering, Domain Name Filtering, URL Filtering, Malware Detection and Removal, Data Loss Prevention, Protective DNS or decryption and re-encryption by a Middlebox Security Function. The capabilities required to support these Security Functions are also defined.

This document supersedes and replaces MEF 88, Application Flow Security for SD-WAN Services [3] by generalizing Security Functions to support any type of IP Service, including SD-WAN.

The key changes from MEF 88 are detailed in Appendix F.

3 Release Notes

This document represents the results of Call for Comments Ballot #2 for MEF W138, with most of the comments received on the Call for Comments Ballot having been resolved. The comments on CfC#2 that have not been resolved, have editor's notes inserted in W138, Working Draft 3.

Call for Comments Ballot #3 is planned to be the last CfC Ballot for this project and should be initiated soon. The contents may change subject to comments received during future Call for Comments Ballots.

This section briefly summarizes the unresolved comments from CfC#2 in the bullet list below:

- Figure 8 (Protective DNS): The plan is to update this figure and related text to make it more in the context of Service Flows. Contribution is planned for early in the next CfC#3 cycle.
- Protective DNS Supported and Protective DNS Unsupported Lists: discussion needed re: tying DNS encryption methods and DNS resource records into one list or two. Discussion needed in next cycle. See Editor Note 9:
- Protective DNS Security Function Policy parameters: The information in the four rows (Supported and Unsupported Lists) in Table 15 needs further discussion and may change. Related to previous bullet item.
- Appendix A (Service Flows and Security Functions): Plan is to a) add Protective DNS to the list of Security Functions, b) update the legend of Security Functions (Figure 9), and c) add a use case (or two) featuring DNS Service Flows and the application of the DNS security functions. Contribution is planned for early in the next CfC#3 cycle.
- Appendix E (Examples of Security Function Policies): the example for Protective DNS is just a placeholder and needs to be fleshed out. A contribution is planned for early in the next cycle.

4 Terminology and Abbreviations

This section defines the terms used in this document. In many cases, the normative definitions to terms are found in other documents. In these cases, the third column is used to provide the reference that is controlling, in other MEF or external documents.

In addition, terms defined in MEF 61.1 [1], MEF 70.1 [2], and MEF 117 [4] are included in this document by reference and are not repeated in Table 1.

Term	Definition	Reference
Allow	An action taken by a Security Function that permits a Service Flow or a subset of a Service Flow to pass.	This document
Allow List	A list of criteria entries that when applied to a Service Flow permits the subset of the Service Flow that contains a match to one of the entries.	This document
Block	An action taken by a Security Function that does not permit a Service Flow or a subset of a Service Flow to pass.	This document
Block List	A list of criteria entries that when applied to a Service Flow denies the subset of the Service Flow that contains a match to one of the entries.	This document
Certificate	An electronic document that uses a digital signature to bind a public key and an identity.	CA Browser Forum [64]
Confidential or Proprietary Information	Information that could harm an organization if made public or used against them.	This document
DNS Protocol Filtering	The Security Function that determines whether a Service Flow, or subset of a Service Flow, contains Domain Name System (DNS) messages that are to be Allowed or Blocked.	This document
Domain Name Filtering	The Security Function that determines whether a Service Flow, or subset of a Service Flow, contains domain names that are to be Allowed or Blocked.	This document
IP, Port and Protocol Filtering	The Security Function that determines whether a Service Flow's source or destination IP addresses, source or destination port numbers, or IP protocols are to be Allowed or Blocked.	This document
IP Service	A connectivity service that carries IP Packets irrespective of the underlying Layer 2 technology.	Adapted from MEF 61.1 [1]
Key Material	Data that is represented as a binary string such that any non-overlapping segments of the string with the required lengths can be used as secret keys, secret initialization vectors, and other secret parameters (also known as keying material).	Adapted from NIST SP 800-56B Rev. 2 [60]
Malware	Software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system.	This document
Malware Detection and Removal	The Security Function that determines whether a Service Flow, or subset of a Service Flow, contains Malware, and removes the Malware or Blocks the subset of the Service Flow containing the Malware.	This document

Middlebox Security Function	A function used to decrypt and re-encrypt secured sessions, e.g., IPsec or TLS, in a Service Flow that allows other Security Functions to apply to the unencrypted Service Flow.	This document
Personally Identifiable Information	Information about an individual maintained by an agency, including information that can be used to distinguish or trace an individual's identity, and other information that is linked or linkable to an individual.	Adapted from NIST SP 800-122 [63]
Policy	A set of rules used to manage and control the changing or maintaining of the state of one or more managed objects.	MEF 95.0.1
Protective DNS	The Security Function that analyzes DNS queries and takes action to mitigate threats, leveraging the existing DNS protocol and architecture.	NSA and CISA [67]
Quarantine List	A list of criteria entries that are not on the Block List but are deemed suspicious.	This document
Security Admin Notification	A notification to the agreed upon list of Subscriber personnel of a change to a Security Function Policy.	This document
Security Admin Notification Policy	An Atomic Policy that specifies the details related to the Security Admin Notification (i.e., recipient list).	This document
Security Event	An incident associated with a Service Flow detected by a Security Function that triggers a notification to the Subscriber.	This document
Security Event Notification	A communication to the agreed upon list of Subscriber personnel of a Security Event.	This document
Security Event Notification Policy	An Atomic Policy that specifies the details related to the Security Event Notification (i.e., recipient list).	This document
Security Function	The component that, when included in a Service Policy, makes a decision to Allow or Block a subset of a Service Flow.	This document
Security Function Policy	A set of parameters for a given Security Function.	This document
Service End Point	A logical construct at an IP-based UNI where Policies are associated with Ingress and Egress Service Flows.	Adapted from MEF 70.1 [2]
Service Flow	A sequence of IP Packets between one or more Subject / Target Actor pairs associated with a given Subscriber that traverse a UNI.	This document
Service Provider	An organization that provides services to Subscribers.	MEF 61.1 [1]
Subject Actor	An Actor requesting access to a Target Actor	MEF 118 [5]
Subscriber's Security Administrator	The person (or persons) in the Subscriber's organization responsible for establishing Security Function Policies and communicating those policies to the Service Provider	This document
Target Actor	An Actor that a Subject Actor wants to access	MEF 118 [5]
URL Filtering	The Security Function that determines whether a Service Flow, or subset of a Service Flow, contains a URL that is to be Allowed or Blocked.	This document

Table 1 - Terminology

The abbreviations used in this document are listed in Table 2.

Abbreviation	Definition	Reference
CA	Certificate Authority	CA Browser Forum [64]
CVE	Common Vulnerabilities and Exposures	MITRE [65]
CPI	Confidential or Proprietary Information	This document
DPF	DNS Protocol Filtering	This document
DNF	Domain Name Filtering	This document
HTTP	Hypertext Transfer Protocol	RFC 7230 [40]
HTTPS	Hypertext Transfer Protocol Secure	RFC 2818 [11]
IOC	Indicators of Compromise	NIST SP 800-53 [59]
IPPF	IP, Port and Protocol Filtering	This document
IPsec	IP Security	RFC 6071 [33]
MD+R	Malware Detection and Removal	This document
MBSF	Middlebox Security Function	This document
PII	Personally Identifiable Information	Adapted from NIST SP 800-122 [63]
PKI	Public Key Infrastructure	NIST SP 800-53 [59]
SAN	Security Admin Notification	This document
SEN	Security Event Notification	This document
TLS	Transport Layer Security	RFC 5246 [28]
URI	Uniform Resource Identifier	RFC 3986 [17]
URL	Uniform Resource Locator	RFC 3986 [17]
UUID	Universally Unique Identifier	RFC 4122 [22]
URLF	URL Filtering	This document
UTC	Coordinated Universal Time	RFC 3339 [13]

Table 2 - Abbreviations

5 Compliance Levels

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 (RFC 2119 [9], RFC 8174 [47]) when, and only when, they appear in all capitals, as shown here. All key words must be in bold text.

Items that are **REQUIRED** (contain the words **MUST** or **MUST NOT**) are labeled as **[Rx]** for required. Items that are **RECOMMENDED** (contain the words **SHOULD** or **SHOULD NOT**) are labeled as **[Dx]** for desirable. Items that are **OPTIONAL** (contain the words **MAY** or **OPTIONAL**) are labeled as **[Ox]** for optional**.**

6 Introduction

With the migration of workloads to the cloud, enterprises need to secure more than just their data centers. Services now span multiple clouds, edge compute, supply chain and distributed workforces which drives the need for the security of IP Services delivered to the enterprise. This has greatly expanded the attack surface. Threats can and do occur everywhere in this ecosystem. The number, scope and impact of such threats and their sophistication is increasing rapidly. The rapid pace and the urgency of implementing security functions to combat these threats is therefore critical. The loss of revenue, ransomware, damaged reputation, and confidence are well-known and cannot be overstated.

This document specifies requirements for security functions used to detect and prevent threats carried across Service Provider IP services. A Service Provider that provides Security Functions to mitigate such threats can help a Subscriber maintain and improve their security posture over time.

This document focuses on specifying Security Functions which protect a Subscriber's Service Flow. It is assumed that Service Flows from different Subscribers are isolated from each other, as would be expected for a network service delivered over a shared infrastructure. This isolation is provided as part of the IP Service to which these Security Functions are added. The Security Functions defined herein can also be used to protect the Service Provider's network infrastructure, but this use case is beyond the scope of this document.

The term Service Flow is defined as a sequence of IP Packets between one or more Actors associated with a given Subscriber that traverse a UNI. MEF 118 [5] defines Actor as a User, Device, or Application. At a minimum, all IP packets are either transmitted or received from Devices, which are one type of Actor. The granularity of the Actor and whether a given IP address can be associated with a User, Application or another Device is dependent on the capabilities of the IP Service. The detailed specification of the Service Flow is left up to the relevant IP service standard, e.g., an Internet access service or an SD-WAN service standard. A Service End Point is defined as a logical construct at an IP Service UNI where policies are associated with Ingress and Egress Service Flows.

A Security Function is defined in this document as the component that, when included in a Service Policy, makes a decision to Allow or Block a subset of a Service Flow. It is intended that any IP Service Standard can reference this Standard when there is a need to apply Security Functions to that Service.

Figure 1 depicts an example of two Service Flows that are identified on ingress at the UNI, one of which has Security Functions applied.

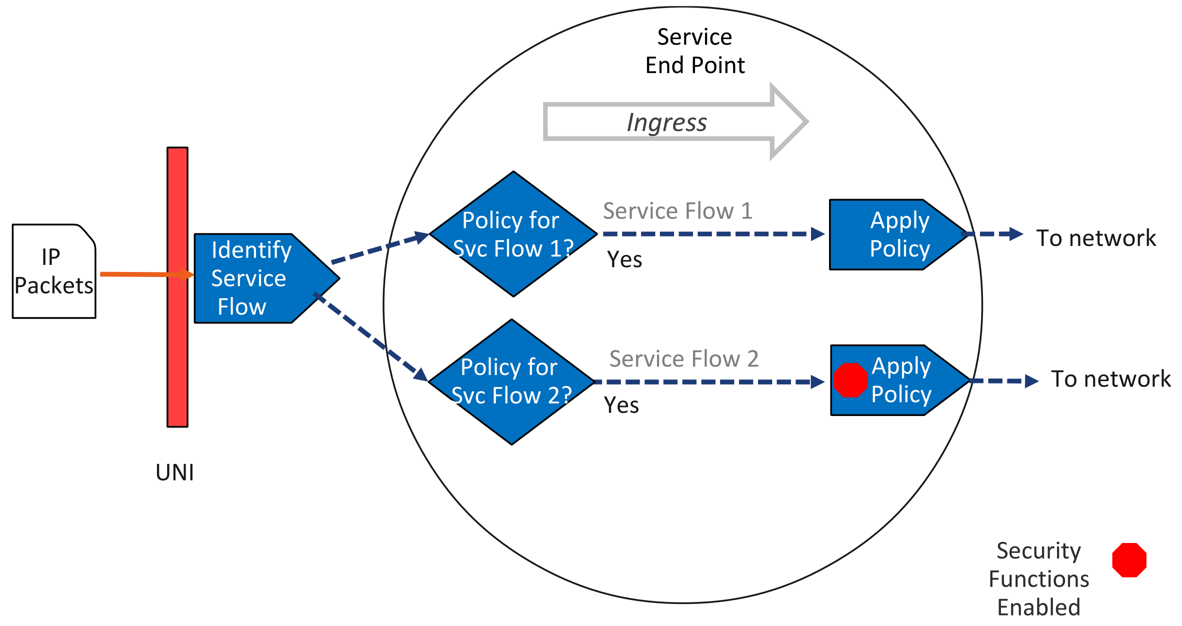


Figure 1 - Example of Service Flow Security for an Ingress Service Flow

In this example, Service Flow 1 carries traffic that does not require additional Security Functions defined in this document. Service Flow 2 carries traffic that requires a set of Security Functions to be applied to the traffic.

Figure 2 depicts a high-level perspective on the positioning of Security Functions supported by the IP Service in the context of Service Flow classification and forwarding. Security Functions can be enabled for Ingress Service Flows and/or Egress Service Flows.

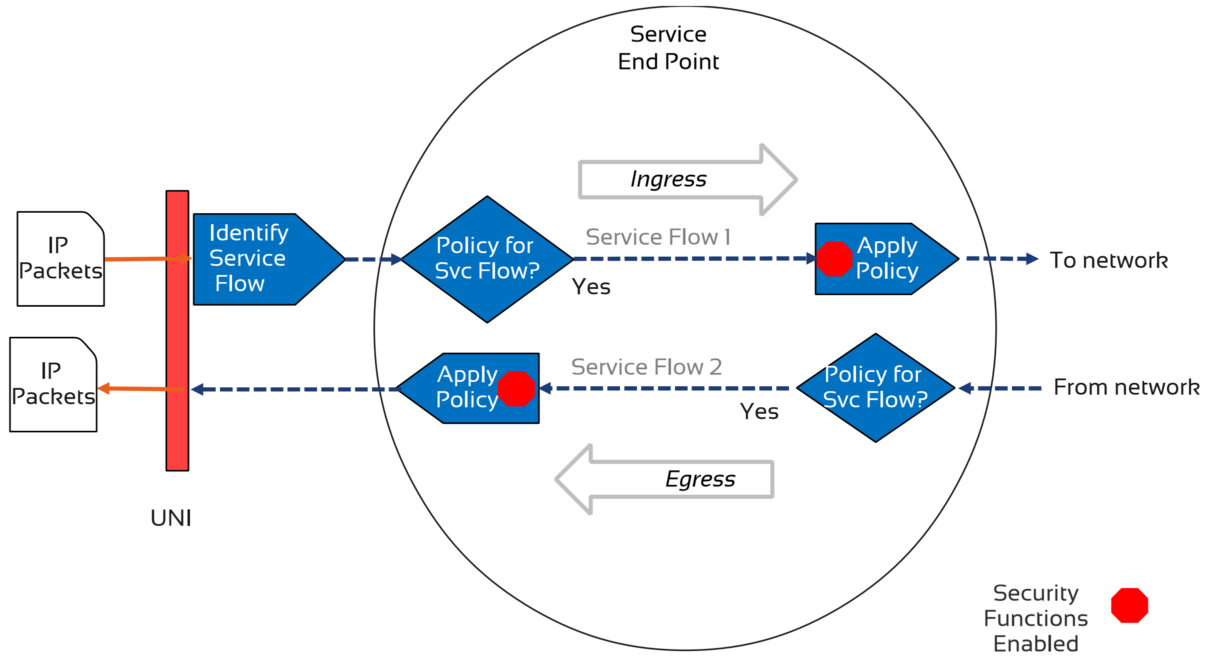


Figure 2 - Example of the Relationship of Security Functions and Service Flows

In the example shown in Figure 2, Security Functions are shown between classification (identifying the Service Flow) and forwarding for both the Ingress Service Flow and the Egress Service Flow. The precise positioning of each Security Function is an implementation choice.

The Service Flows depicted in Figure 2 could be stateless (e.g., both Service Flow 1 and Service Flow 2 are independent) or stateful (e.g., both Service Flow 1 and Service Flow 2 are considered to be one Service Flow). It is up to the Service to determine the relationship. When an IP Security Function is stateful, a security action is applied for both the ingress and egress directions. When an IP Security Function is stateless, a security action needs to be applied for each direction, independently. How state is tracked by the Security Function is outside the scope of this document.

Required capabilities in support of Security Functions are specified in Section 7. These include:

- Security Action Lists: when a Security Function is enabled, a specific Block List, Allow List, and Quarantine List for that Security Function is required. The generic requirements related to the lists are specified.
- Security Event Notification (SEN): notifies the agreed upon list of Subscriber personnel when security actions are taken.
- Security Admin Notification (SAN): notifies the agreed upon list of Subscriber personnel when a significant change to a Security Function Policy is made.

When the term Subscriber is used in the context of Security Function Policy agreements, it means anyone the Subscriber delegates the authority to. Often it is the Subscriber's Security Administrator that makes those decisions. The Subscriber's Security Administrator is defined as the person (or persons) in the Subscriber's organization responsible for establishing Security Function Policies and communicating those policies to the Service Provider.

The Middlebox Security Function, when enabled for a given Service Flow, can decrypt encrypted subsets of the Service Flow to allow for scanning and then re-encrypts. In this document, use of the term TLS refers to TLS 1.2, as specified in IETF 5246 [28], unless another version is specified. The Middlebox Security Function is a special case, i.e., it not only can take on the role of a Security Function, but it is required by many of the other Security Functions to be able to fully do their work. The description and requirements related to the Middlebox Security Function are detailed in Section 8.

Additional Security Functions are listed here and further detailed in Section 9.

- IP, Port and Protocol Filtering, when enabled for a given Service Flow, can Allow or Block a subset of the Service Flow based on source or destination IP address, source or destination port number and/or IP protocol.
- Domain Name Filtering, when enabled for a given Service Flow, can Allow or Block a subset of the Service Flow based on domain name. Wildcard entries are supported.
- URL Filtering, when enabled for a given Service Flow, can Allow or Block a subset of the Service Flow based on the URL. Wildcard entries are supported.

- Malware Detection and Removal, when enabled for a given Service Flow, scans Objects for Malware and takes appropriate action when Malware is detected.
- Data Loss Prevention, when enabled can protect the Service Flow from possible exfiltration of Personally Identifiable Information (PII) and Confidential and Proprietary Information (CPI).
- DNS Protocol Filtering, when enabled for a given Service Flow, can Allow or Block a subset of the Service Flow based on DNS message type and source/destination IP addresses.
- Protective DNS, when enabled for a given Service Flow, provides for enhanced security for DNS, i.e., checking whether each DNS request/response is allowed.

In this document, the words Allow and Block are used when describing the possible action of a Security Function, and when used in this context, the terms are always capitalized. Allow is defined as an action taken by a Security Function that permits a Service Flow or a subset of a Service Flow to pass. Block is defined as an action taken by a Security Function that does not permit a Service Flow or a subset of a Service Flow to pass. Note that these actions apply to each Security Function independently, i.e., for a given Service Flow, one Security Function might Allow, while another Security Function might Block. In this case, the overall effect is to Block. See Appendix A for use case examples of how these Security Functions interact with different Service Flows.

Section 10 defines an atomic policy for each Security Function. A set of parameters are agreed between the Subscriber and Service Provider for each Security Function Policy. Each IP Service that references this Standard for adding Security Functions can incorporate the policy parameters for each Security Function into either a Composite Policy [4] or an Atomic Policy [4], as required by the Service.

The remainder of this document is organized as follows:

- References are listed in Section 11.
- Use cases relating Service Flows and Security Functions are described in Appendix A.
- Examples of Security Event Notifications are provided in Appendix B.
- Examples of Malware Detection and Removal are described in Appendix C.
- A summary description of a detailed threat modeling report is provided in Appendix D, together with a link to the threat modeling tool output.
- Examples of Atomic Security Function Policies are provided in Appendix E.
- Major changes from MEF are provided in Appendix F.
- Acknowledgments are provided in Appendix G.

Readers should review the related threat model document, referenced in Appendix D, to better familiarize themselves with the overriding threat model of IP service implementations. Further threats will likely exist beyond the scope of this document and would require additional agreement between the Service Provider and Subscriber to address these threats. A selection of potential threats to supporting services, such as DNS, can be found in the accompanying threat model document. The conditions for how remaining threats from the threat model are treated, which must be treated within the Middlebox Security Function context, can be found in Section 8. Readers should familiarize themselves with [R52], which covers constraints on how data should be processed if it leaves the Middlebox Security Function.

This document does not impose any explicit constraints on where in the network Security Functions can be applied - it allows for flexibility in various deployment options. It is assumed that when a Security Function is provided on a Service Flow, all possible paths have the same security scanning. A Service Provider may distribute Security Functions, e.g., in a cloud or at the customer premises, depending on the functionality and where it can be optimally placed. This document also does not address requirements to secure the IP Services network itself. The Service Provider is responsible for securing its own network.

Note that when the term *support* is used in a normative context in this document, it means that the Service Provider can enable the functionality upon agreement with the Subscriber.

When the term *enabled* is used in the context of a Security Function, it means that Security Function is included in the policy for a given service. And when the term *disabled* is used in the context of a Security Function, it means that Security Function is not included in the policy for a given service.

7 Required Capabilities in Support of Security Functions

The Security Functions are specified in Sections 8 and 9 of this document. Required capabilities in support of these Security Functions are specified in this section, including:

- Security Action Lists, i.e., generic requirements for the Block List, Allow List, Quarantine List, Supported List, and Unsupported List are specified in Section 7.1.
- Security Event Notification (SEN) requirements are specified in Section 7.2.
- Security Admin Notification (SAN) requirements are specified in Section 7.3.
- DNS Resolution is described at a high level in Section 7.4.

7.1 Security Action Lists

Security Action Lists can be used by each Security Function to take an action. The five lists specified in this section are: Block List (Section 7.1.1), Allow List (Section 7.1.2), Quarantine List (Section 7.1.3), Supported List (Section 7.1.4), and Unsupported List (Section 7.1.5). In addition, Requirements Pertaining to All of the Lists (Section 7.1.6) and Action Modifiers (Section 7.1.7) are specified, as indicated.

The term "criteria entry" is used to indicate an entry in a list. Depending on the Security Function to which it applies, a criteria entry could be a specific port number, protocol identifier, URL, domain name, IP address, hash, pattern, or some other identifier within the context of a Service Flow. A given Security Function can have three types of lists in which different criteria entries are placed, based on user preferences, the criteria entry's reputation, or other factors. The criteria entry would be used by a Subscriber's policy management system to specify the policy criterion and what action to take if a match occurs.

The term *Action* is defined as an operation taken by a Security Function on a Service Flow or a subset of a Service Flow. In this document, there are two Actions defined: *Block* or *Allow*. A Service Flow, or a subset of a Service Flow, can pass through a given Security Function unmodified (Allowed) or be Blocked. The *Action* could be used by a Subscriber's policy management system to specify the policy action to take when a match occurs to the policy criterion.

The term *Action Modifier* is defined as an influencer on how an Action is performed and is further elaborated in Section 7.1.7. One or more Acton Modifiers can be attached to criteria entries on a Security Action List, as determined by agreement between the Subscriber and Service Provider.

The following provides a brief description of these lists and their behaviors.

7.1.1 Block List

A Block List is a list of criteria entries that when applied to a Service Flow denies the subset of the Service Flow that contains a match to one of the entries. There is a different Block List for each of the Security Functions. For each Security Function, the name of the Block List is prefixed with the name of the Security Function, e.g., the URL Filtering Block List. The Service Provider is responsible for enforcing the Block List.

The list of criteria entries in the Block List include:

- Criteria entries that are deemed a security threat by the Subscriber
- Criteria entries that are deemed a security threat by the Service Provider
- Criteria entries, not related to security threats, that conform to the Subscriber's policies, including applicable business or regulatory compliance requirements

A Block List can consist of thousands of criteria entries. It is not practical for the Subscriber to provide complete/exhaustive criteria entries for the Block List. For any given Security Function, it is up to the Service Provider and Subscriber to agree on the format used by the Subscriber to add criteria entries to the Block List.

The Service Provider has access to a security threat database for each Security Function that provides an up-to-date comprehensive list of criteria entries to guard against security threats. The Subscriber might also have a security threat database. It is expected that the Service Provider and Subscriber will merge those lists to guard against security threats. The security threat database is dynamic, i.e., it changes often with new entries being added and older entries being deleted. Specifications on the threat database, the method for maintaining each of the lists, and the method for providing this interaction for the Subscriber's use, e.g., an Application Programming Interface (API), are beyond the scope of this document.

As for Subscriber policy, the Subscriber typically provides categories that need to be Blocked, e.g., gambling, pornography, movies, home shopping, etc. The Service Provider then fills in the detailed criteria entries for each category. The database of detailed criteria entries within each category is also dynamic. The details of how a category is specified are beyond the scope of this document.

The Block List is a combination of all these entries, and dynamically changes over time. The following requirements apply to the Block List for each Security Function.

[R1] For each Security Function, the Service Provider **MUST** maintain a list of criteria entries in the Block List.

Once the service is turned up, the criteria entries on the Block List may be modified at the request of the Subscriber. This is normally a dynamic process. This could be via a web portal or API. Such implementation methods are beyond the scope of this document.

[R2] For each Security Function, the Service Provider **MUST** allow the Subscriber to add criteria entries to the Block List.

[R3] For each Security Function, the Service Provider **MUST** allow the Subscriber to remove a criteria entry on the Block List that was added by one of the following methods:

- Specified explicitly by the Subscriber
- Specified by the Service Provider to conform to a category specified by the Subscriber

It is important that the Subscriber not be allowed to modify an entry on the Block List that the Service Provider added because of a perceived security threat.

[R4] For each Security Function, the Service Provider **MUST NOT** allow the Subscriber to remove a criteria entry from the Block List that the Service Provider specified due to a security threat.

Since the security threat database can get out of sync, it is important that the Service Provider provide the frequency of updates, e.g., hourly, daily, weekly, per change.

[R5] For each Security Function, the Service Provider **MUST** provide to the Subscriber the frequency of updates to the security threat database.

There needs to be a process in place to deal with issues that can occur with the security threat database or with the Block List. For example, a security threat that was on the Block List is no longer a security threat (problem was fixed), or there was a mistake in a criteria entry in the list (e.g., a typo).

[R6] For each Security Function, the Service Provider **MUST** provide a documented process to allow the Subscriber to request a change to an entry on the Block List that the Service Provider specified due to a security threat.

7.1.2 Allow List

An Allow List is a list of criteria entries that when applied to a Service Flow permits the subset of the Service Flow that contains a match to one of the entries. There is a different Allow List for each of the Security Functions. For each Security Function, the name of the Allow List is prefixed with the name of the Security Function, e.g., the URL Filtering Allow List. The Service Provider is responsible for enforcing the criteria entries on the Allow List. The criteria entries in the Allow List include:

- Criteria entries that are explicitly identified by the Subscriber

The Subscriber can provide explicit criteria entries for a given Allow List, or categories that are permitted, e.g., all URLs within the corporate domain. For any given Security Function, it is up to the Service Provider and Subscriber to agree on the format used by the Subscriber to add criteria entries to the Allow List.

If a criteria entry on the Allow List becomes a possible security threat, e.g., one URL in the corporate domain has been compromised, that criteria entry would be removed from the Allow List and added to either the Block List or the Quarantine List.

The following requirements apply to the Allow List for each Security Function.

[R7] For each Security Function, the Service Provider **MUST** maintain a list of criteria entries in the Allow List.

[R8] For each Security Function, the Service Provider **MUST** allow the Subscriber to add or remove criteria entries in the Allow List.

7.1.3 Quarantine List

A Quarantine List is a list of criteria entries that are not on the Block List but are deemed suspicious. The subset of the Service Flow that contains a match to one of the criteria entries on the Quarantine List is Blocked. There is a different Quarantine List for each of the Security Functions. For each Security Function, the name of the Quarantine List is prefixed with the name of the Security Function, e.g., the URL Filtering Quarantine List. The criteria entries in the Quarantine List include:

- Criteria entries that are deemed suspicious by the Subscriber
- Criteria entries that are deemed suspicious by the Service Provider

For any given Security Function, it is up to the Service Provider and Subscriber to agree on the format used by the Subscriber to add criteria entries to the Quarantine List. One example helps to illustrate the use of the Quarantine List. When the service is activated and the URL Filtering Security Function is *Enabled*, the URL Filtering Quarantine List is empty. Later, if the Service Provider detects any suspicious activity associated with a specific criteria entry (in this case a specific URL) that was on the URL Filtering Allow List, the Service Provider removes that criteria entry from the URL Filtering Allow List and places it on the URL Filtering Quarantine List, for further investigation by the Subscriber.

While monitoring the Security Functions in a given service, the Service Provider might determine that a particular Service Flow is compromised. If the Service Provider can qualify the compromise into a set of criteria entries, the Service Provider would add those criteria entries to the Quarantine List. In the case that such a criteria entry had been on the Allow List, then the Service Provider would remove that criteria entry from the Allow List to comply with [R14].

The Subscriber's own monitoring might determine that a particular Service Flow is compromised. If the Subscriber can qualify the compromise into a set of criteria entries, the Subscriber can either add those criteria entries to the Block List or to the Quarantine List, removing criteria entries from the Allow List in cases of conflict to comply with [R14]. However, given that the Block List might possibly contain many entries, and the Quarantine List may be smaller, the Subscriber might utilize the Quarantine List as an investigative list. This could be easier to search and easier to trigger questions as to why certain entries are on the Quarantine List. This choice is purely up to the Subscriber.

In both cases, sets of criteria entries were added to the Quarantine List. The Service Provider may not be able to determine the exact nature of exploitation. However, if the Service Provider determines that the compromise is such that it needs to be included in the threat intelligence feeds (e.g., vulnerabilities and IOCs), then the Service Provider would remove the set of criteria entries from the Quarantine List and add them to the Block List. However, if the set of criteria entries is not deemed severe enough to be added to the threat intelligence feeds, then the Service Provider would not remove the set of criteria entries from the Quarantine List, unless there is agreement with the Subscriber to do so.

While it is true that traffic may not flow for a certain amount of time and it may appear that the vulnerability has been mitigated, there is no way for the Service Provider to positively determine the difference from mitigation and temporary stoppage of data transmission. Therefore, it is security best practice that the set of criteria entries remain on the Quarantine List until the Subscriber removes those entries.

Suspicious entries are added to the Quarantine List dynamically, and the Subscriber makes the final decision to remove them from the Quarantine List with only one exception, elevation to the threat intelligence feeds (and hence to the Block List) from the Service Provider.

The Quarantine List enables the Subscriber to take one of the following actions:

- Move the criteria entry from the Quarantine List to the Block List, or
- Move the criteria entry to the Allow List, or
- Ignore the criteria entry and leave it in the Quarantine List.

The following requirements apply to the Quarantine List for each Security Function.

[R9] For each Security Function, the Service Provider **MUST** maintain a list of criteria entries in the Quarantine List.

[R10] For each Security Function, the Service Provider **MUST** allow the Subscriber to do each of the following:

- add criteria entries to the Quarantine List
- remove criteria entries from the Quarantine List

7.1.4 Supported List

A Supported List is a list of criteria entries that specifies the capability of a given Security Function. This is needed for some of the Security Functions specified in this Standard, i.e., the Middlebox Security Function (Section 8) and the Protective DNS Security Function (Section 9.6.2).

[R11] Based on agreement with the Subscriber, the Service Provider **MUST** maintain a list of criteria entries in the Supported List for each Security Function that uses a Supported List.

7.1.5 Unsupported List

An Unsupported List is a list of criteria entries that specifies that a given Security Function is not able to process such entries. This is needed for some of the Security Functions, i.e., the Middlebox Security Function (Section 8) and the Protective DNS Security Function (Section 9.6.2).

[R12] Based on agreement with the Subscriber, the Service Provider **MUST** maintain a list of criteria entries in the Unsupported List for each Security Function that uses an Unsupported List.

[R13] The Service Provider **MUST** ensure that a criteria entry on the Supported List for a given Security Function cannot also appear on the Unsupported List for that Security Function.

7.1.6 Requirements Pertaining to All of the Lists

To ensure conflict is minimized, a criteria entry cannot be on more than one list.

[R14] For each Security Function, the Service Provider **MUST** ensure that each criteria entry is on at most one of the following lists: Allow List, Block List, or Quarantine List.

There are at least two cases where potential conflicts could occur.

Example 1: A criteria entry on one list could be generalized, e.g., all URLs in www.qaz.com/world are to be permitted, and so the criteria entry on the Allow List uses a wildcard to denote this generalization, www.qaz.com/world/*. Due to a security threat, the URL www.qaz.com/world/badpage needs to be denied, and so it is placed on the Block List. In this case, the more specific criteria entry overrules the more general criteria entry, i.e., all access to www.qaz.com/world is Allowed, except for www.qaz.com/world/badpage, which is Blocked.

Example 2: The Service Provider deems suspicious a criteria entry that is on the Allow List. The Service Provider removes it from the Allow List and places it on the Quarantine List. The Subscriber might have good reason to Allow the criteria entry and decide to place it back on the Allow List.

For a Security Function that uses Supported and Unsupported Lists, each criteria entry on the Supported List and the Unsupported List need to also be included in one of the following lists: the Security Function Allow List, the Security Function Block List, and the Security Function Quarantine List. The following requirements apply.

[R15] For a given Security Function that uses Supported List and Unsupported Lists, the Service Provider **MUST** ensure that each criteria entry on the Supported List is also on exactly one of the following lists for that Security Function: the Allow List, the Block List, or the Quarantine List.

[R16] For a given Security Function that uses Supported List and Unsupported Lists, the Service Provider **MUST** ensure that each criteria entry on the Unsupported List is also on exactly one of the following lists for that Security Function: the Allow List, the Block List, or the Quarantine List.

[R17] For a given Security Function that uses Supported List and Unsupported Lists, the Service Provider **MUST** ensure that each criteria entry on the Allow List is also on exactly one of the following lists for that Security Function: Supported List or the Unsupported List.

[R18] For a given Security Function that uses the Supported List and Unsupported List, the Service Provider **MUST** ensure that each criteria entry on the Block List is also on exactly one of the following lists for that Security Function: Supported List or the Unsupported List.

[R19] For a given Security Function that uses the Supported List and Unsupported List, the Service Provider **MUST** ensure that each criteria entry on the Quarantine List is also on exactly one of the following lists for that Security Function: Supported List or the Unsupported List.

It is the responsibility of the Service Provider to implement these lists in such a way as to match the intent of the Subscriber.

7.1.7 Action Modifiers

An Action Modifier is defined as an influencer on how an Action is performed. When a subset of the Service Flow matches a criteria entry on one of the Lists, that subset of the Service Flow is either Blocked or Allowed, depending on which list it appears on. An Action Modifier indicates any additional actions that need to be taken. Note that the Action Modifier does not change the operation applied to the subset of the Service Flow, i.e., Block or Allow. The following Action Modifier is defined in this section, including the specification of any related requirements: Notification. Other Action Modifiers may also be agreed, but these are not specified here.

Notification - a notification is sent to the Subscriber. The Subscriber indicates when the Notification Action Modifier is used. The Subscriber typically uses it to ensure that events that are important to the Subscriber trigger a notification. In this document, the notification is a Security Event Notification (SEN), as specified in Section 7.2.

Since there can be thousands of criteria entries on a List, and hence thousands of possible Action Modifiers, the Subscriber typically provides the Service Provider with a Policy that provides categories of criteria entries for a given Action Modifier. For example, a given Security Function Policy could use the following categories of criteria entries for the Block List: Security Threat (high-risk), suspicious (low risk) and Other (all other). And, by policy, the Notification Action Modifier can be enabled in the Block List for the high-risk security threat category to immediately send a Security Event Notification whenever a subset of a Service Flow is Blocked due to a high-risk security threat. And the policy could also state that whenever a subset of a Service Flow is

Blocked due to other reasons, a Security Event Notification would be archived by the Service Provider for possible further inspection by the Subscriber. The Service Provider and Subscriber are free to agree on the categories associated with the Notification Action Modifier. The details of how a category is specified are beyond the scope of this document.

The Service Provider is responsible for maintaining the criteria entries for each List, as well as any associated Action Modifiers derived from the Policy. The Subscriber can change any of the Action Modifiers at any time. The method for maintaining the Action Modifiers for a given Security Function, and the method for providing this interaction for the Subscriber's use, e.g., an Application Programming Interface (API), are beyond the scope of this document.

The following requirements apply to the Notification Action Modifier.

[R20] The Service Provider **MUST** support the use of the Notification Action Modifier, for each Security Function independently, applicable to the Security Function's Block List, Allow List, or Quarantine List.

[R21] All criteria entries on a Security Function's Block List and Quarantine List **MUST** contain the Notification Action Modifier, unless modified by the Subscriber.

[D1] The Service Provider **SHOULD** support the use of more than one notification level for the Notification Action Modifier, as agreed by the Subscriber and Service Provider.

[CR1]<[D2] If multiple notification levels are used, the Subscriber and Service Provider **MUST** agree on the timing of issuing each SEN per the notification level.

Multiple alert levels can be used to moderate the number of Security Event Notifications sent immediately to the Subscriber. For example, the highest alert level could trigger an immediate Security Event Notification, whereas a lower-level Alert could result in a log and daily/weekly report to the Subscriber.

[R22] The Service Provider **MUST** allow the Subscriber to change the Action Modifiers on any of the Lists.

7.2 Security Event Notification (SEN)

A Security Event is defined as an incident associated with a Service Flow detected by a Security Function that triggers a notification to the Subscriber due to either a) a Notification Action Modifier associated with the criteria entry on any of the Security Function's Lists, or b) the Service Flow does not match a criteria entry on any of the Security Function's Lists but is Blocked due to the application of the Security Function Policy.

One example of a Security Event is when the Malware Detection and Removal Security Function blocks a subset of a Service Flow due to a known security threat, and the signature for that Malware matches a criteria entry on the Block List with the Notification Action Modifier. A second example is when the Data Loss Prevention Security Function detects Personally Identifiable Information in a Service Flow that matches a criteria entry on the Allow List with the Notification Action Modifier. A third example is when there is no match on any of the Lists, and the IP, Port and Protocol Filtering Security Function Blocks the subset of the Service Flow because the Subscriber's Policy is to Block the subset of the Service Flow that has no match.

A Security Event Notification (SEN) is defined as a communication to the agreed upon list of Subscriber personnel of a Security Event. The SEN typically includes an Indicator of Compromise (IOC) and other information about the Security Event.

IOCs provide organizations with valuable information on objects or information systems that have been compromised. Typical IOCs are virus signatures, IP addresses, hashes of Malware files, or URLs or domain names of botnet command and control servers.

Per NIST SP 800-53 [59], "Indicators of compromise (IOC) are forensic artifacts from intrusions that are identified on organizational information systems (at the host or network level). IOCs provide organizations with valuable information about objects or information systems that have been compromised. IOCs for the discovery of compromised hosts can include, for example, the creation of registry key values. IOCs for network traffic include, for example, Universal Resource Locator (URL) or protocol elements that indicate Malware command and control servers. The rapid distribution and adoption of IOCs can improve information security by reducing the time that information systems and organizations are vulnerable to the same exploit or attack."

[R23] A SEN **MUST** be issued whenever a subset of a Service Flow:

- matches a criteria entry that contains the Notification Action Modifier on a Security Function's Block List, Allow List, or Quarantine List, or
- does not match a criteria entry on any of the Security Function's Lists but is Blocked due to the application of the Security Function Policy.

[R24] When multiple notification levels are used, the SEN **MUST** be issued based on the agreement between the Subscriber and Service Provider, per [CR1]<[D1].

[R25] The Service Provider **MUST** store each SEN in a secure repository for future reference and security auditing purposes.

The amount of time that a SEN needs to be stored is agreed between the Subscriber and Service Provider, e.g., it could be in accordance with the Subscriber's data retention policy.

To improve the User experience, the Service might decide to immediately inform the client when the client initiated the Service Flow and the Security Function Blocked a subset of that Service Flow. However, this notification could provide Users, who have malicious intent, information about what is blocked by the Service. Therefore, the service might deem it more secure not to inform the client about the circumstances surrounding the blocking of the subset of the Service Flow. In cases where the Service Flow originates from the Server and is destined for the client, it is highly probable that the Service would inform neither the client nor the Server due to a Security Policy.

The Service definition would determine if, when, and under what circumstances a client would be informed due to a subset of the Service Flow being blocked by a Security Function. In all instances, a Security Event Notification would be created for each subset of the Service Flow that was Blocked as per [R23] .

A SEN could get generated by the Service or by the Security Function. It is outside the scope of this document to mandate exactly how the SEN is generated. However, this document does mandate items that need to be included in a SEN (see [R26]), as well as recommended items (see [D2]).

[R26] A SEN **MUST** include the items listed in Table 3.

Item	**Value**	**Comments**
Issuer	UTF-8 [16] String[1]	Examples: Service Provider Name, Security Vendor Name, etc.
Timestamp of SEN	date-time	RFC 3339 [13], Section 4.1 UTC
SEN ID	UUID	RFC 4122 [22] Universally Unique Identifier
Security Function	UTF-8 [16] String	Security Function Name
Security Function Policy ID	UTF-8 [16] String	The identifier of the Security Function Policy
Type of SEN	UTF-8 [16] String	IOC, ATT&CK, or OTHER
Type of Security Event	UTF-8 [16] String	A description of the Security Event. Includes the CVE [65] ID (for an IOC event) or the TA number [66] (for an ATT&CK event). Examples: reconnaissance, credential access, command and control, exfiltration.
Security Event Source IP address	Human readable IPv4 dotted decimal IPv6 hexadecimal strings	IANA Number Resources [54]
Security Event details	UTF-8 [16] String	The details re: the Security Event, e.g., the rule violation (for an IOC event) or the T number [66] (for an ATT&CK event).
Action Taken	UTF-8 [16] String	Examples: informational, quarantined or Blocked, Malware removed

Table 3 - Items to be included in a SEN

The UNI ID associated with the Security Event might not always be known by the Security Function. When it is known, it is recommended that the UNI ID be included in the SEN.

[D2] A SEN **SHOULD** include the Security Event UNI ID, expressed with a UTF-8 [16] String.

[1] UTF-8, Unicode Transformation Format 8-bit

The format of the SEN is not specified in this document.

This document mandates that URLs and domains listed in the SEN be neutralized. It also recommends the use of square brackets, which are reserved characters in RFC 3986 [17], to neutralize a domain or URL in a SEN. For example, if the compromised detail includes www.domain.tld, the SEN will send it as www[.]domain[.]tld.

[R27] Any domain name or URL in a SEN **MUST** be neutralized.

[D3] The method for neutralizing the domain name or URL in a SEN **SHOULD** use square brackets around each period.

Other items could also be included in the SEN, if agreed, e.g., hostname, username, owner of public IP address, place, contact, etc.

[R28] The timestamp of the SEN **MUST** be expressed in UTC [13].

[R28] means that the SEN is time stamped in UTC, but it does not prohibit the Service Provider from also presenting a specific time zone to a specific Subscriber.

A Security Event Notification Policy is defined as an Atomic Policy that specifies the details related to the Security Event Notification (i.e., recipient list). Note that it is up to the Service to determine the number of SEN Policies.

[R29] A Security Event Notification Policy **MUST** contain the following:

- *List of Recipients* and one or more contact methods for each recipient authorized to receive the SEN

[R29] means that for each recipient in the *List of Recipients* there needs to be one or more associated contact method(s). For example, the *List of Recipients* could include an e-mail distribution list for the Security Operations Center, a Network Operations Center Slack channel, or e-mail address of an individual.

Additional parameters can be agreed between the Subscriber and Service Provider.

Appendix B provides examples of two types of SENs – IOC and ATT&CK.

7.3 Security Admin Notification (SAN)

A Security Admin Notification (SAN) is a notification to the agreed upon list of Subscriber personnel of a change to a Security Function Policy. A SAN is issued when one or more parameter values are changed for a given Security Function Policy. If parameter values for multiple Security Function Policies are changed at the same time, a separate SAN is issued for each of those Security Function Policies. The comprehensive list of parameter values for each Security Function Policy is available to the Subscriber.

The SAN includes only those Security Function Policy parameter values that have changed. The Service Provider can send the SAN to the Subscriber, as needed, and archives each SAN for later inspection by the Subscriber.

Note that a change in the mapping of a Security Function Policy to a Service Flow is controlled by the Service definition, and whether a service change notification is sent to the Subscriber is beyond the scope of this document.

[R30] A SAN **MUST** be issued whenever parameter element values for a given Security Function Policy are changed.

[R31] The Service Provider **MUST** store each SAN in a secure repository for future reference and auditing purposes.

The amount of time that a SAN needs to be stored is agreed between the Subscriber and Service Provider, e.g., it could be in accordance with the Subscriber's data retention policy.

[R32] A SAN **MUST** include the items listed in Table 4:

Item	Value	Comments
Issuer	UTF-8 [16] String[2]	Example: Service Provider Name
Timestamp of SAN	date-time	RFC 3339 [13] Section 4.1 UTC
SAN ID	UUID	RFC 4122 [22] Universally Unique Identifier
Source IP address	Human readable IPv4 dotted decimal IPv6 hexadecimal strings	IANA Number Resources [54] (Identifier of the individual making the change)
Security Function Policy Name	UTF-8 [16] String	The name of the Security Function Policy that was changed. Example: *URL Filtering Policy Red*.
SAN details	UTF-8 [16] String	Example: If the URL Filtering Block List was changed, the list of criteria entries that were added or removed in the URL Filtering Block List would be listed. Username of person making the change.

Table 4 - Items to be included in a SAN

For example, in the URL Filtering Security Function Policy a change is made to the list of criteria entries for the URL Filtering Block List parameter that deletes some of the entries in the URL Filtering Block List. The Subscriber needs to be informed of such a change.

The format of the SAN is not specified in this document.

This document mandates that URLs and domains listed in the SAN be neutralized. It also recommends the use of square brackets, which are reserved characters in RFC 3986 [17], to

[2] UTF-8, Unicode Transformation Format 8-bit

neutralize a domain or URL in a SAN. For example, if the compromised detail includes www.domain.tld, the SAN will send it as www[.]domain[.]tld.

[R33] Any domain name or URL in a SAN that is associated with a Block List or Quarantine List **MUST** be neutralized.

The method for neutralizing the domain name or URL in a SAN **SHOULD** use square brackets around each period.

Other items could also be included in the SAN, if agreed, e.g., the name of the person authorizing the change.

[R34] The timestamp of the SAN **MUST** be expressed in UTC [13].

[R34] means that the SAN is time stamped in UTC, but it does not prohibit the Service Provider from also presenting a specific time zone to a specific Subscriber.

A SAN is used to notify interested parties of a change to a Security Function Policy.

[R35] A Security Admin Notification Policy **MUST** contain the following:

- *List of Recipients* and one or more contact methods for each recipient authorized to receive the SAN

[R35] means that for each recipient in the *List of Recipients* there needs to be one or more associated contact method(s)., e.g., email distribution list, or messaging service channel.

[D4] The *List of Recipients* referred to in [R35] **SHOULD** include those responsible for creating a Subscriber's Security Function Policies.

Additional parameters can be agreed between the Subscriber and Service Provider.

7.4 DNS Resolution

A DNS resolution function, referred to as a DNS resolver in RFC 1035 [6], is necessary to provide the capability to offer a service responding to customer DNS requests with either upstream DNS responses (from the legitimate DNS servers for a given resource) or to examine the request and insert a different custom response based on the security policy applied by the various security functions. It is important to resolve DNS queries securely. See the DNS Protocol Filtering Security Function (Section 9.6.1) and the Protective DNS Security Function (Section 9.6.2).

8 Middlebox Security Function

The Middle Box Function, as defined in MEF 88, considers only TLS wrapped connections, however there are other ways in which traffic can be encrypted/signed for secure transmission. The process of looking at other cryptographic protocols necessitates generalizing the approach to Key Materials and ciphers in a way that will be applicable no matter what protocol is in use. Ultimately, this standard aims to ensure that there is no degradation of security properties, no matter what protocol is intercepted.

The Middlebox Security Function provides for the decryption and re-encryption[3] of a Service Flow that uses Transport Layer Security (TLS), Internet Protocol Security (IPsec), or other cryptographic protocols that may be supported.

Many Security Functions can only work when the Service Flow is unencrypted. An encrypted Service Flow must be decrypted for those Security Functions to inspect the packets, and then re-encrypted after the Security Function actions are taken.

In the context of this document, the Middlebox Security Function is used to decrypt and re-encrypt one or more secured sessions (e.g., TLS, IPsec) in a Service Flow, and that allows other Security Functions to apply to the unencrypted Service Flow. This may be required by specific Security Functions that need to see unencrypted traffic to detect, remove, filter, etc. In this document, for convenience, when we say that the Service Flow is decrypted using a Middlebox Security Function, it means that both decryption and re-encryption are performed on the Service Flow.

If the Middlebox Security Function is applied to a Service Flow containing unencrypted packets, such unencrypted packets would be either Allowed or Blocked. The action applied to such packets are determined based on the agreement of the Subscriber and Service Provider.

Editor Note 2: *a) Consider whether to add a requirement that the Service Provider MUST support the ability for the Subscriber to choose which action to apply for unencrypted packets. Such a requirement would tighten things up. b) should this be added to the policy section?*

The following terms are used when the Middlebox Security Function decrypts encrypted connections:

- For TLS, 'client' is used to reference "the application entity that initiates a TLS connection to a server", and 'server' is used to reference "the application entity that responds to requests for connections from clients", per RFC 5246 [28].

[3] Note that the Middlebox Security Function does not take unencrypted traffic coming into the Service Provider's network and send it through encrypted, nor does it take encrypted traffic coming into the Service Provider's network and send it through unencrypted.

- For IPsec, RFC 7296 [41] (IKEv2) uses the term ‘endpoint’ to indicate the end of an IPsec Security Association (SA). An endpoint can be an ‘initiator’, i.e., the one requesting an SA to be set up, or a responder (i.e., the one responding to the request).
- For other protocols, other terms could be used.

The Middlebox Security Function can be used as part of a Service Policy, based upon agreement between the Subscriber and Service Provider.

Four lists (Middlebox Security Function Supported List, Middlebox Security Function Unsupported List, Middlebox Security Function Allow List and Middlebox Security Function Block List) are maintained by the Service Provider for each Middlebox Security Function Policy, with each criteria entry on a list including the following information.

[R36] When the Middlebox Security Function supports TLS, each Middle-box Security Function List **MUST** contain the following criteria entry parameters for TLS.

- Type of encryption protocol: TLS
- Protocol Version, e.g., 1.1, 1.2, 1.3…
- List of Cipher Suites for a given TLS protocol version (any can be used to indicate that any cipher suite for a given TLS protocol version is on the list)

[R37] When the Middlebox Security Function supports IPsec, each Middle-box Security Function List **MUST** contain the following criteria entry parameters for IPsec.

- Type of encryption protocol: IPsec
- List of Internet Key Exchange v2 (IKEv2) parameters, per Table 1 of NIST SP 800-77 Rev. 1 [62]
- List of IPsec parameters, per Table 1 of NIST SP 800-77 Rev. 1 [62]

Additional criteria entries can be agreed between the Subscriber and Service Provider.

Other encryption protocols can be specified using a similar structure for the criteria entries.

As for TLS connections, a cipher suite includes the protocol name (i.e., TLS), the Key Exchange algorithm (e.g., DHE_RSA), with the encryption algorithm (e.g., AES_128_GCM) and the Message Authentication algorithm (e.g., SHA256). The following is an example of a cipher suite for TLS 1.2: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x00,0x9E). The IANA TLS Cipher Suites Registry [53] provides a full list of cipher suites.

As for IPsec, the IKEv2 parameters include the encryption algorithm (e.g., AES-GCM), the integrity/pseudorandom function (e.g., HMAC-SHA256), the Diffie-Hellman group (e.g., DH 21), the peer authentication algorithm (e.g., RSA with 3072-bit or larger key), and the lifetime (e.g., 24 hours). The IPsec parameters include the protocol (ESP), version (e.g., IPsec-v3), the encryption algorithm (e.g., AES-CCM), the integrity algorithm (e.g., HMAC-SHA512), the perfect forward secrecy algorithm (e.g., DH 21), and the lifetime (e.g., 8 hours).

The Middlebox Security Function Supported List is a list of criteria entries that specifies that the Middlebox Security Function can decrypt the subset of a Service Flow that contains a match to one of the criteria entries.

Based on agreement with the Subscriber, the Service Provider needs to maintain a list of criteria entries in the Middlebox Security Function Supported List (see [R11]).

The Middlebox Security Function Unsupported List is a list of criteria entries that specifies that the Middlebox Security Function is not capable of decrypting the subset of a Service Flow that contains a match to one of the criteria entries.

Based on agreement with the Subscriber, the Service Provider needs to maintain a list of criteria entries in the Middlebox Security Function Unsupported List (see [R12] and [R13]).

The Middlebox Security Function Block List is a list of criteria entries, which specifies that the Middlebox Security Function Blocks the subset of the Service Flow that contains a match to one of the entries.

The Middlebox Security Function Allow List is a list of criteria entries, which specifies that the Middlebox Security Function Allows the subset of the Service Flow that contains a match to one of the entries.

Requirements related to the Middlebox Security Function Block List, the Middlebox Security Function Allow List, Middlebox Security Function Supported List, and the Middlebox Security Function Unsupported List are found it Section 7.1 of this document.

[R38] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the Middlebox Security Function Block List, the Middlebox Security Function Allow List, the Middlebox Security Function Supported List, and the Middlebox Security Function Unsupported List.

The criteria entries on the Middlebox Security Function Supported List and the Middlebox Security Function Unsupported List need to be included in the Middlebox Security Function Allow List or the Middlebox Security Function Block List, and vice versa. See requirements [R15], [R16], [R17], and [R18].

When a Middlebox Security Function Policy is included in a Policy for a given Service Flow, there are four cases to consider.

- Case 1: A subset of the Service Flow matches a criteria entry on the Middlebox Security Function Block List. The Middlebox Security Function Blocks this subset.
- Case 2: A subset of the Service Flow matches a criteria entry on the Middlebox Security Function Allow List and on the Middlebox Security Function Supported List. The Middlebox Security Function decrypts this subset, allowing for other Security Functions to be applied to the unencrypted subset, and then re-encrypts the subset for transport in the Service Flow.
- Case 3: A subset of the Service Flow matches a criteria entry on the Middlebox Security Function Allow List and on the Middlebox Security Function Unsupported List. The Middlebox Security Function passes this subset through, unchanged.
- Case 4: A subset of the Service Flow does not match an entry on any of the lists. This is the 'no-match' case, and this is covered by the agreement of the Subscriber and Service Provider to either pass this through the Middlebox Security Function unchanged or to Block it.

The following requirements specify the Middlebox Security Function behavior for these cases.

[R39] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Middlebox Security Function Block List **MUST** be Blocked.

[R40] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Middlebox Security Function Allow List and on the Middlebox Security Function Supported List **MUST** be decrypted.

[R41] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Middlebox Security Function Allow List and on the Middlebox Security Function Unsupported List **MUST** be passed through the Middlebox Security Function without change.

It is possible that a subset of the Service Flow does not match a criteria entry on any of these lists. Requirements [R42] and [R43] cover this gap.

[R42] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the Middlebox Security Function lists:

- Block the subset of the Service Flow
- Pass the subset of the Service Flow through the Middlebox Security Function without change

[R43] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Middlebox Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the Middlebox Security Function lists:

- Block the subset of the Service Flow
- Pass the subset of the Service Flow through the Middlebox Security Function without change

When a Service Flow is carrying multiple protocols, it is possible that a subset of the Service Flow cannot be decrypted by the Middlebox Security Function, e.g., a subset of the Service Flow may be unencrypted, or encrypted with an encryption protocol not supported by the Middlebox Security Function. Furthermore, in some cases, the Middlebox Security Function may not be able to re-encrypt an encryption protocol even if the CA is trusted. This could be the case when the CA of the Middlebox Security Function is trusted by the Subscriber's client, and the Middlebox Security Function is able to successfully decrypt the encryption protocol from the Subscriber's client in the decryption phase of the Middlebox Security Function process, however, the target server CA may have pinned the client certificate to the Subscriber's client, prohibiting the use of a client certificate by the Middlebox Security Function for the re-encryption phase of the Middlebox Security Function process.

Some protocols present challenges for a Middlebox Security Function to support. For example, the HTTP Strict Transport Security (HSTS) protocol, as specified in IETF RFC 6797 [37], can run over TLS and provides a mechanism implemented at the server to expressly prevent a Middlebox Security Function. Another example is IPsec, as specified in RFC 7296 [41], that uses perfect forward secrecy for secure transport.

[D5] The Middlebox Security Function **SHOULD** state the cryptographic standards, e.g., FIPS-140-2 [57], with which it is aligned with respect to the cryptographic processing that is performed.

[R44] The Middlebox Security Function **MUST** support at least one of the following secure transport protocols:

- Transport Layer Security (TLS)
- IPsec

[R45] Based on agreement with the Subscriber, the Service Provider **MUST** maintain a list of secure transport protocols that are supported by the Middlebox Security Function.

[R46] The Middlebox Security Function **MUST** use one or more of the protocols identified in [R45].

[R47] When TLS is supported, the Middlebox Security Function **MUST** meet the mandatory requirements of TLS 1.2, per RFC 5246 [28].

Editor Note 3: There's a question about the need for a requirement (e.g., recommendation) re: TLS 1.3. A related question - do we need a brief description of how TLS 1.3 (and 1.2) work, i.e., what are the key differences from a Middlebox perspective? All good contributions are welcome.

[R48] When TLS is supported, the Middlebox Security Function **MUST** meet the mandatory requirements of Section 9.3 of RFC 8446 [48] (Protocol Invariants).

[R49] When IPsec is supported, the Middlebox Security Function **MUST** meet the mandatory requirements of IKEv2, per RFC 7296 [41].

This document does not require the Middlebox Security Function to support any specific cryptographic standards. For the case where a protocol and/or cipher suite is not supported, but where it can be identified (e.g., TLS 1.3 per RFC 8446 [48]), these should be processed as per the Middlebox Security Function Unsupported List. If the Subscriber wants to use to use such an unsupported protocol and/or cipher suite, the Subscriber will need to ensure that the criteria entries related to aspects of the protocol and/or cipher suite that can be identified, are also on the Middlebox Security Function Allow List.

[R50] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Middlebox Security Function **MUST NOT** change the protocol implementation or cryptographic suite of the session as compared to the client (TLS) or initiator endpoint (IPsec) request.

[R51] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Middlebox Security Function **MUST NOT** choose a weaker cipher suite in the negotiation for a session as compared to the session without the Middlebox Security Function.

NIST SP 800-52 [58] and NIST SP 800-77 [62] are useful references that provide guidelines for the selection, configuration, and use of common protocols and the selection of cryptographic suites for US federal systems.

Figure 3 depicts a Middlebox Security Function used between a Subscriber's client (Subject Actor) and a server (Target Actor) - the server can be either a public Internet host or an internal private host in the Subscriber's network. The terms Subject Actor and Target Actor are appropriate terms per MEF 118 [5]. For simplicity, the Service network is not shown here. As depicted in Figure 3, a certificate chain consists of all the certificates needed to certify the Middlebox Security Function's certificate. This includes the Middlebox Security Function's proposed certificate, the certificate(s) of any intermediate CAs, and the certificate of a root CA trusted by all parties in the chain.

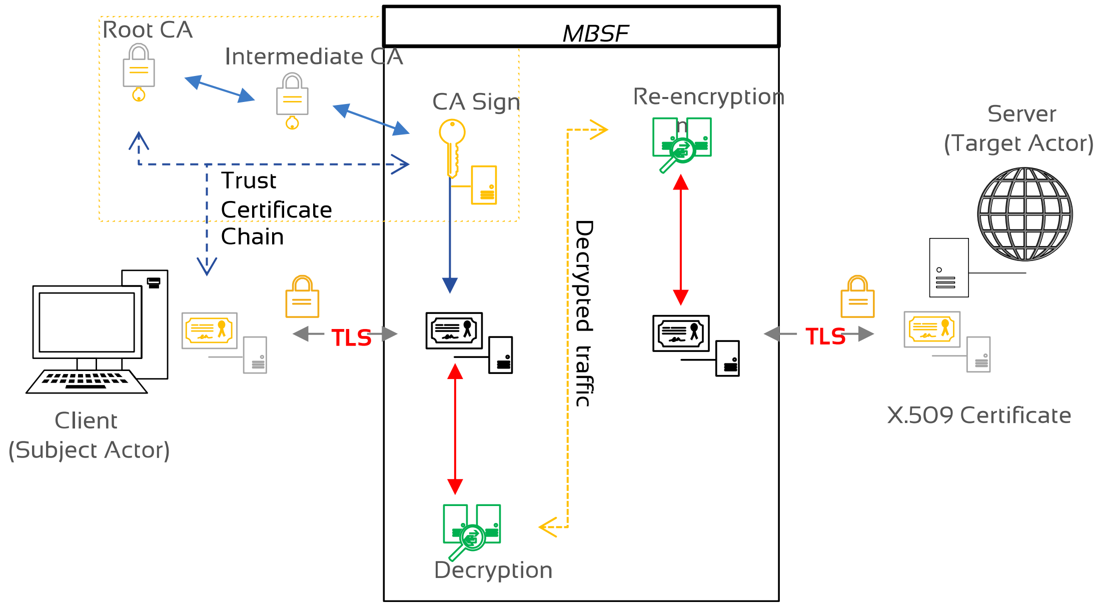


Figure 3 - Example of Middlebox Security Function for an encrypted flow between a Subscriber's client and a Server

The attack surface presented by the Middlebox Security Function from which the decrypted traffic can be intercepted and modified/analyzed prior to re-encryption should be minimized to limit attacks. In particular, decrypted traffic should remain within the customer's designated tenancy and should not be distributed across multiple physical nodes (via shared compute, shared memory and/or shared network fabric) in a manner that presents opportunities for unauthorized access, e.g., within a customer or between customers. This is in addition to the specific requirement ([R52]) that resultant data (logging or otherwise) must not be transported outside of the Middlebox Security Function with a lower level of protection than it was originally transmitted.

Once a given Service Flow is decrypted by the Middlebox Security Function, then other Security Functions can be applied. Figure 4 depicts an example of an implementation that applies Security Functions to an Ingress Service Flow that is encrypted. A similar approach could be used for an Egress Service Flow.

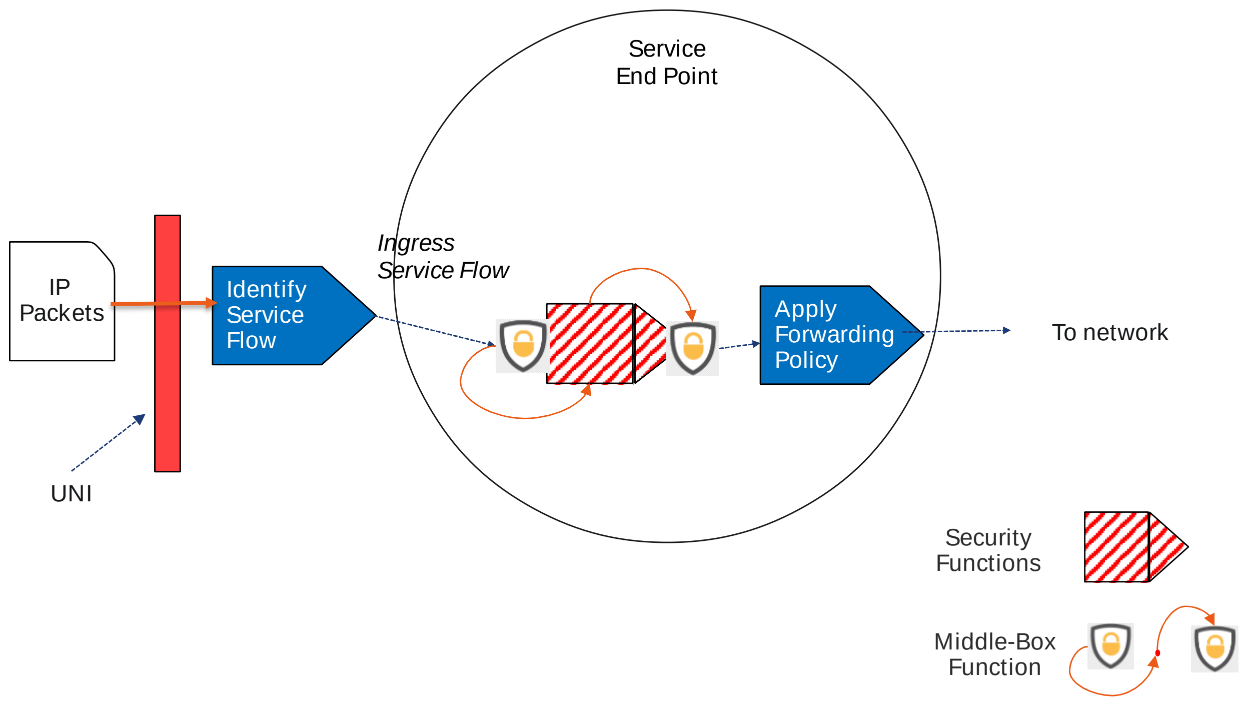


Figure 4 - Example of Middlebox Security Function applied to a Service Flow

Note: It is desirable that decryption and re-encryption happen within the same Middlebox Security Function to avoid further compromising the security of Subscriber's Service Flow during the application of Security Functions.

[R52] A Service Flow decrypted by the Middlebox Security Function **MUST NOT** be exposed outside the Service Provider's Security Functions in an unencrypted form or in an encrypted form that offers a lower level of confidentiality and integrity than the originally encrypted Service Flow.

The expectation is that the Middlebox Security Function is implemented within a secure perimeter within the Service Provider's network, and that unencrypted traffic cannot leave that secure perimeter. Ancillary traffic relating to an encrypted Service Flow needs to also be secured such that it meets the expectation of the Subscriber as to the confidentiality and integrity of the data being communicated.

8.1 Certificate Authority and Validation

The Certificate Authority (CA) and certificate validation are agreed between the Subscriber and the Service Provider to account for both public and private certificate usage.

[R53] When TLS or other protocols using PKI are used, the Middlebox Security Function **MUST** be capable of issuing valid, signed certificates for each encrypted session in a manner that is trusted by the Subscriber.

[R54] When TLS or other protocols using PKI are not used, the Middlebox Security Function **MUST** be capable of issuing Key Material for each encrypted session in a manner that is trusted by the Subscriber.

[R55] When TLS or other protocols using PKI are used and when performing inspection within the Middlebox Security Function, it **MUST** be possible to use certificates that are backed by a CA where the trust path and issuer can be validated by users within the Subscriber's network who have installed the full certificate chain on their computer.

[R56] When TLS or other protocols using PKI are used, the Service Provider **MUST** ensure that the certificate chain allowing for this validation to occur, as described in [R55], is made available to the Subscriber.

While this document does not prescribe the exact hierarchy of trust that should be leveraged when replacing complex certificate chains in-line, the following requirements have been deemed necessary.

[R57] Server certificates and/or other Key Material **MUST** be generated and regenerated with fresh, suitably random material per the requirements in FIPS-140-2 [57] for which the Middlebox Security Function processes Service Flows.

[R58] All replacement properties for each encrypted session, e.g., alternative certificate server names, certificate validity periods, and choices of cipher suites **MUST NOT** reduce the level of security functionality.

[R59] Where a CA is operated in support of the inspection within the Middlebox Security Function, it **MUST** clearly identify itself within the visible issuer properties (for TLS, within the certificates, as defined in Section 4.1.2.4 of RFC 5280 [29]) presented, for reasons of transparency, so that the Subscriber can identify when a Middlebox Security Function is in the path versus when they are connecting directly to the originating server.

[D6] Any TLS based interception being performed by a Middlebox Security Function as part of a managed service **SHOULD** use a Public Key Infrastructure (PKI) hierarchy that is rooted in a CA that is operated in line with the CA/Browser Forum baseline requirements [64] where certificates are securely created, used, revoked and destroyed.

[D7] Where possible, CAs **SHOULD** log all certificates that they issue using the standardized Certificate Transparency (CT) security standard, see RFC 6962 [39].

Per [D7], this creates a system of public logs that seek to eventually record all certificates issued by publicly trusted certificate authorities, allowing efficient identification of mistakenly or maliciously issued certificates.

[R60] The Middlebox Security Function **MUST** be capable of accepting a valid, signed Subscriber certificate for each TLS session between the Middlebox Security Function and the Subscriber's server.

[R61] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Middlebox Security Function **MUST** verify the validity/identity of the client and server (TLS), as illustrated in Figure 3.

[R62] When a Middlebox Security Function Policy is included in a Security Policy for a given Service Flow, the Middlebox Security Function **MUST** verify the validity/identity of the initiator and responder endpoints (IPsec).

[R63] When an invalid/unknown client and server (TLS) is detected, the Middlebox Security Function **MUST** be capable of performing a variety of behaviors including blocking, allowing, etc., and notifying the client (TLS) in the Subscriber's network of the discrepancy.

[R64] When an invalid/unknown initiator and responder endpoint (IPsec), is detected, the Middlebox Security Function **MUST** be capable of performing a variety of behaviors including blocking, allowing, etc., and notifying the initiator endpoint (IPsec) of the discrepancy.

Editor Note 4: Do we need to add this into the policy section: i.e., the behavior to apply is based on Subscriber - SP agreement? need some thoughtful feedback.

The method for notifying the client (TLS) in the Subscriber's network, or the initiator endpoint (IPsec) is beyond the scope of this document.

Note that this document does not impose any requirements on validating the client certificate.

Additional information on key management can be found in NIST SP 800-57 [61].

9 Security Functions

The Security Functions specified in this section can be applied to a Service Flow (Ingress or Egress) at a given Service End Point, and include:

- IP, Port and Protocol Filtering - detailed specification in Section 9.1
- Domain Name Filtering - detailed specification in Section 9.2
- URL Filtering - detailed specification in Section 9.3
- Malware Detection and Removal - detailed specification in Section 9.4
- Data Loss Prevention – detailed specification in Section 9.5
- DNS Security Functions - detailed specifications in Section 9.6
    - DNS Protocol Filtering - detailed specification in Section 9.6.1
    - Protective DNS – detailed specification in Section 9.6.2

9.1 IP, Port and Protocol Filtering

IP, Port and Protocol Filtering is defined as the Security Function that determines whether a Service Flow’s source or destination IP addresses, source or destination port numbers, or IP protocols are to be Allowed or Blocked. A Security Policy might disallow specific IP addresses, IP protocols and/or port numbers that are not used by the Subscriber, to mitigate possible attacks.

An IP, Port and Protocol Filtering Block List consists of a list of criteria entries used by the IP, Port and Protocol Filtering Security Function to Block the subset of the Service Flow that contains a match to one of the entries. An IP, Port and Protocol Filtering Allow List consists of a list of criteria entries used by the IP, Port and Protocol Filtering Security Function to Allow the subset of the Service Flow that contains a match to one of the entries. An IP, Port and Protocol Filtering Quarantine List is a list of criteria entries that are not on the IP, Port and Protocol Filtering Block List but are deemed suspicious. The subset of the Service Flow that contains a match to one of the entries on the IP, Port and Protocol Filtering Quarantine List is Blocked by the IP, Port and Protocol Filtering Security Function. The three lists are maintained by the Service Provider.

[R65] Each criteria entry on an IP, Port and Protocol Filtering List **MUST** include the fields listed in Table 5.

Additional fields can be agreed between the Subscriber and Service Provider.

Field	Description
SAV4	List of IPv4 source addresses. Any could be used to indicate that any source IPv4 address is on the list.
DAV4	List of IPv4 destination addresses. Any could be used to indicate that any destination IPv4 address is on the list.
PROTV4	List of IPv4 protocols; a list of integers in the range 0 to 255 or a list of keywords from IANA, Protocol Numbers [55], or a mix of integers and keywords.
SAV6	List of IPv6 source addresses. Any could be used to indicate that any source IPv6 address is on the list.
DAV6	List of IPv6 destination addresses. Any could be used to indicate that any destination IPv6 address is on the list.
NEXTHEADV6	List of IPv6 protocols; a list of integers in the range 0 to 255 or a list of keywords from IANA, Protocol Numbers [55], or a mix of integers and keywords.
SPORT	List of transport source ports - is a list of integers in the range 0 to 65535 or a list of service names from IANA, Service Name and Transport Protocol Port Number Registry [56] or a mix of integers and service names. Any could be used to indicate that any source port number is on the list.
DPORT	List of transport destination ports - is a list of integers in the range 0 to 65535 or a list of service names from IANA, Service Name and Transport Protocol Port Number Registry [56] or a mix of integers and service names. Any could be used to indicate that any destination port number is on the list.

Table 5 - Criteria entry fields for IP, Port and Protocol Filtering

The following are some examples of an entry on the IP, Port and Protocol Filtering Block List:

- A criteria entry in the IP, Port and Protocol Filtering Block List of <*Any, Any, TCP, Any, Any, TCP, Any, 53*> means that the subset of the Service Flow that contains TCP and destination port 53 would be Blocked for both IPv4 and IPv6.
- A criteria entry in the IP, Port and Protocol Filtering Block List of <*Any, Any, GRE, Any, Any, GRE, Any, Any*> means that the subset of the Service Flow that encapsulates GRE over IPv4 or IPv6 would be Blocked.
- A criteria entry in the IP, Port and Protocol Filtering Block List of <*Any, Any, [TCP, UDP], Any*, *Any, [TCP, UDP], Any,* [*137,138,139]*> means that the subset of the Service Flow that contains TCP or UDP destination ports 137, 138, or 139 (NetBIOS) would be Blocked.

Requirements related to the IP, Port and Protocol Filtering Block List, IP, Port and Protocol Filtering Allow List and the IP, Port and Protocol Filtering Quarantine List are found in Section 7.1 of this document.

[R66] When an IP, Port and Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the IP, Port and Protocol Filtering Block List, the IP, Port and Protocol Filtering Allow List and the IP, Port and Protocol Filtering Quarantine List.

[R67] When an IP, Port and Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the IP, Port and Protocol Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R68] When an IP, Port and Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the IP, Port and Protocol Filtering Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the IP, Port and Protocol Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R69] When an IP, Port and Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the IP, Port and Protocol Filtering Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the subset of the Service Flow that matches a criteria entry on the IP, Port and Protocol Filtering Allow List
- Allow the subset of the Service Flow that does not match a criteria entry on any of the IP, Port and Protocol Filtering lists, per the second bullet of [R68]
- Block the subset of the Service Flow that matches a criteria entry on the IP, Port and Protocol Filtering Block List
- Block the subset of the Service Flow that matches a criteria entry on the IP, Port and Protocol Filtering Quarantine List
- Block the subset of the Service Flow that does not match a criteria entry on any of the IP, Port and Protocol Filtering lists, per the first bullet of [R68]

[D8] The IP, Port and Protocol Filtering Security Function **SHOULD** use the same set of Block/Allow/Quarantine Lists for Service Flows including DNS messages and Service Flows not including DNS messages.

[D8] means that in the event there are different IP, Port and Protocol Filtering policies available for a given Subscriber's service,

- the criteria entries in each of the IP, Port and Protocol Filtering Block Lists in the different policies should be the same; and

- the criteria entries in each of the IP, Port and Protocol Filtering Allow Lists in the different policies should be the same; and
- the criteria entries in each of the IP, Port and Protocol Filtering Quarantine Lists in the different policies should be the same.

This would ensure the same behavior in the Subscriber's service whether the Service Flow contains DNS messages or not.

9.2 Domain Name Filtering

Domain Name Filtering is defined as the Security Function that determines whether a Service Flow, or subset of a Service Flow, contains domain names that are to be Allowed or Blocked. Domain Name Filtering provides a level of protection from attempting to access a malicious host.

A Domain Name Filtering Block List is a list of domains used by the Domain Name Filtering Security Function to Block the subset of the Service Flow that contains a match to one of the entries. The Subscriber may be notified of the Block, including the cause. A Domain Name Filtering Allow List is a list of domains used by the Domain Name Filtering Security Function to Allow the subset of the Service Flow that contains a match to one of the entries. A Domain Name Filtering Quarantine List is a list of domains that are deemed suspicious but have not been identified on the Domain Name Filtering Block List. Access to a criteria entry on the Domain Name Filtering Quarantine List is Blocked by the Domain Name Filtering Security Function until further security checks can be done. The Subscriber may be notified of the Block, including the cause. The Subscriber referred to in this paragraph can be informed either via a SEN (if the reason for the Block triggers a SEN), or via immediate communication to the client (Subject Actor) depending on the Service Policy. For notification by the SEN, see [R23] for conditions that trigger a SEN.

The lists are maintained by the Service Provider, with each criteria entry on a list being a domain name. Each domain name on a list can be either explicit, e.g., host.domain.tld or wildcarded, e.g., *.domain.tld.

Note that the process of qualifying a domain name is a process that happens outside of this Security Function. The Protective DNS Security Function, see section 9.6, can qualify domain names used in DNS messages and this can also be used by the Domain Name Filtering Security Function to Block or Allow domain names in a Service Flow that does not contain DNS messages.

[R70] The Domain Name Filtering Security Function **MUST** be capable of using wildcard criteria entries.

[R71] The Service Provider **MUST** inform the Subscriber of the options regarding the use of wildcards for the Domain Name Filtering Security Function.

The following are examples of wildcard options: *.domain.tld, host.*.tld and host. domain.*

There are five basic methods for identifying whether a subset of the Service Flow matches a criteria entry on the Domain Name Filtering Block List, Domain Name Filtering Allow List, or the Domain Name Filtering Quarantine List. All five methods may be used on a single Service Flow.

- Inspect the unencrypted DNS messages (TCP and UDP port 53) to identify the domain names.
- When the Protective DNS Security Function (see section 9.6) is used, encrypted DNS messages, e.g., DNS over HTTPS [49] using port 443 and DNS over TLS [44] using port 853, can be checked to identify the domain names.
- Inspect the subset of the Service Flow that is unencrypted to identify the domain names.
- For the subset of the Service Flow that is encrypted with TLS 1.2 or below and uses the Server Name Indication extension field of the handshake message per RFC 6066 [32], inspect the Server Name Indication extension field to identify the domain names.
- If a Middlebox Security Function is Enabled for the Service Flow and a subset of the Service Flow is encrypted, inspect the decrypted subset of the Service Flow to identify the domain names.

Note that the identification methods described in bullets 1, 3, and 4 above can also be used when the Middlebox Security Function is *Disabled*.

The Service Provider has a list of disreputable or undesirable domain names (e.g., those known to cause security threats or have illicit content) for the Domain Name Filtering Block List. This a)Domain Name Filtering Block List is maintained by the Service Provider and dynamically modified over time as additional domains to be Blocked are identified. The Subscriber may also provide a list of domain name categories and/or specific domain names to populate the Domain Name Filtering Block List.

An example of Domain Name Filtering using DNS messages is shown in Figure 5 .

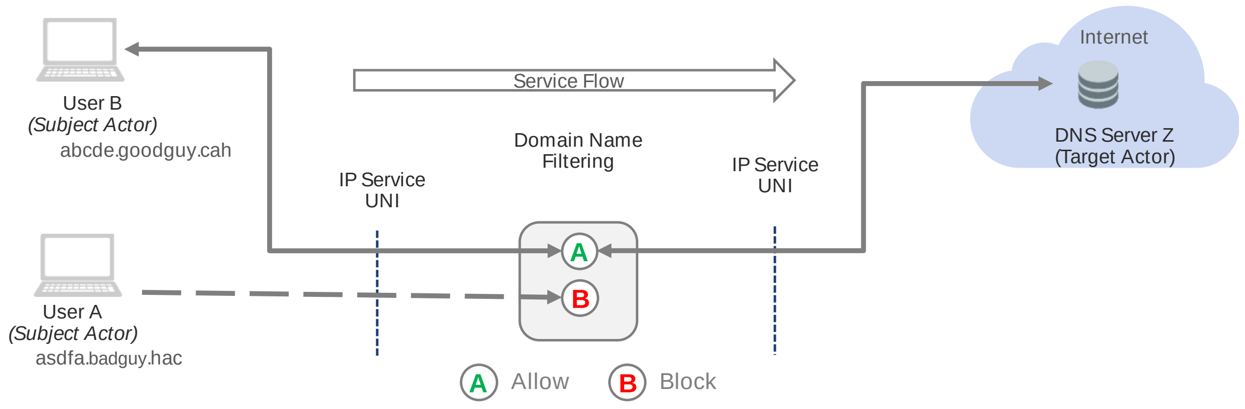


Figure 5 - Example of Domain Name Filtering using DNS Messages

As shown in Figure 5 , users A and B (Subject Actors) send unencrypted request messages to DNS Server Z (Target Actor), which is on the DNS Protocol Filtering Allow List. Domain Name Filtering is *Enabled*, and '*.badguy.hac' is on the Domain Name Filtering Block List, so the subset of the Service Flow containing 'asdfa.badguy.hac' is Blocked. Only DNS messages with Allowed domain names get forwarded to the DNS server.

Requirements related to the Domain Name Filtering Block List, the Domain Name Filtering Allow List and the Domain Name Filtering Quarantine List are found in Section 7.1 of this document.

[R72] When a Domain Name Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the Domain Name Filtering Block List, the Domain Name Filtering Allow List and the Domain Name Filtering Quarantine List.

[R73] When a Domain Name Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the Domain Name Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R74] When a Domain Name Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Domain Name Filtering Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the Domain Name Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R75] When a Domain Name Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Domain Name Filtering Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the subset of the Service Flow that matches a criteria entry on the Domain Name Filtering Allow List
- Allow the subset of the Service Flow that does not match a criteria entry on any of the Domain Name Filtering lists, per the second bullet of [R74]
- Block the subset of the Service Flow that matches a criteria entry on the Domain Name Filtering Block List
- Block the subset of the Service Flow that matches a criteria entry on the Domain Name Filtering Quarantine List
- Block the subset of the Service Flow that does not match a criteria entry on any of the Domain Name Filtering lists, per the first bullet of [R74]

[D9] The Domain Name Filtering Security Function **SHOULD** use the same set of Block/Allow/Quarantine Lists for Service Flows including DNS messages and Service Flows not including DNS messages.

[D9] means that in the event there are different Domain Name Filtering policies available for a given Subscriber's service,

- the criteria entries in each of the Domain Name Filtering Block Lists in the different policies should be the same; and
- the criteria entries in each of the Domain Name Filtering Allow Lists in the different policies should be the same; and
- the criteria entries in each of the Domain Name Filtering Quarantine Lists in the different policies should be the same.

This would ensure the same behavior in the Subscriber's service whether the Service Flow contains DNS messages or not.

9.3 URL Filtering

URL Filtering is defined as the Security Function that determines whether a Service Flow, or subset of a Service Flow, contains a URL that is to be Allowed or Blocked. URL is specified in IETF RFC 3986 [17]. URL Filtering applies to cases where the domain name is on the Domain Name Filtering Allow List, but one or more URLs associated with that domain have a security issue and need to be Blocked.

A URL Filtering Block List is a list of URLs used by the URL Filtering Security Function to Block the subset of the Service Flow that contains a match to one of the entries, i.e., access to that URL is Blocked. The Subscriber may be notified of the Block, including the cause. A URL Filtering Allow List is a list of URLs used by the URL Filtering Security Function to Allow the subset of the Service Flow that contains a match to one of the entries, i.e., access to that URL is Allowed. A URL Filtering Quarantine List is a list of URLs that are deemed suspicious but have not been identified on the URL Filtering Block List. Access to a URL on the URL Filtering Quarantine List is Blocked by the

URL Filtering Security Function until further security checks can be done. The Subscriber may be notified of the Block, including the cause. The Subscriber referred to in this paragraph can be informed either via a SEN (if the reason for the Block triggers a SEN), or via immediate communication to the client (Subject Actor) depending on the Service Policy. For notification by the SEN, see [R23] for conditions that trigger a SEN.

The lists are maintained by the Service Provider, with each criteria entry on a list using a URL. Each URL on a list can be either explicit, e.g., host.domain.tld/section/world/europe or wildcarded, e.g., host.domain.tld/section/*.

[R76] The URL Filtering Security Function **MUST** be capable of using wildcard criteria entries.

[R77] The Service Provider **MUST** inform the Subscriber of the options regarding the use of wildcards for the URL Filtering Security Function.

There are two basic methods for identifying whether a subset of the Service Flow matches a criteria entry on one of the URL Filtering lists:

- If the Service Flow is unencrypted, inspect the Service Flow to identify the URL.
- If the Service Flow is encrypted, enable the Middlebox Security Function for the Service Flow, and inspect the unencrypted packets to identify the URL.

An example of URL Filtering applied to a TLS session is shown in Figure 6 .

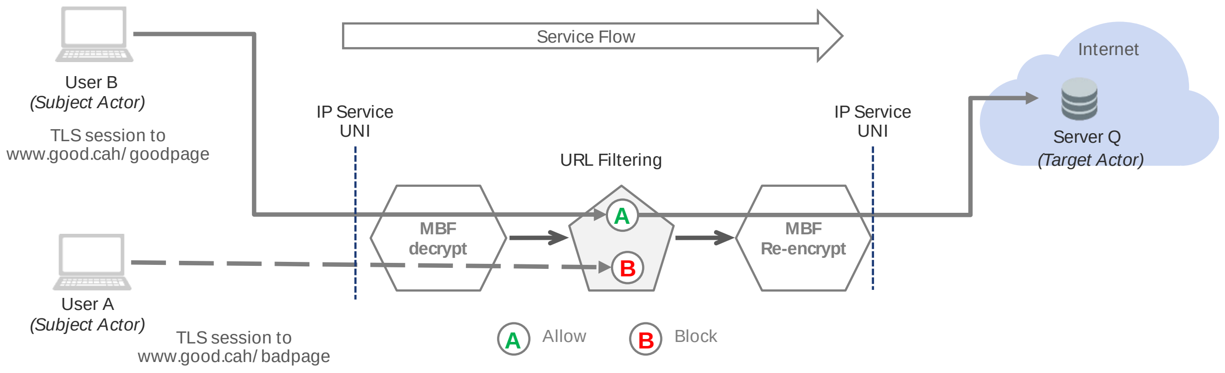


Figure 6 - Example of URL Filtering applied to TLS session

In the example shown in Figure 6, Server Q (Target Actor) hosts several domains. URL Filtering is enabled for the Service Flow and since the Service Flow is encrypted, a Middlebox Security Function is also enabled. User A (Subject Actor) tries to connect to Server Q for a URL that is on the URL Filtering Block List, and that connection attempt is Blocked. User B (Subject Actor) connects to Server Q for a URL that is on the URL Filtering Allow List, and that connection is Allowed. Only appropriate URLs get connected.

The Service Provider may have a list of disreputable or undesirable URLs associated with good domains (e.g., those URLs known to cause security threats or have illicit content) for the URL Filtering Block List.

Requirements related to the URL Filtering Block List, the URL Filtering Allow List and the URL Filtering Quarantine List are found in Section 7.1 of this document.

[R78] When a URL Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the URL Filtering Block List, the URL Filtering Allow List, and the URL Filtering Quarantine List.

[R79] When a URL Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the URL Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R80] When a URL Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the URL Filtering Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the URL Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R81] When a URL Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the URL Filtering Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the subset of the Service Flow that matches a criteria entry that is on the URL Filtering Allow List
- Allow the subset of the Service Flow that does not match a criteria entry on any of the URL Filtering lists, per the second bullet of [R80]
- Block the subset of the Service Flow that matches a criteria entry on the URL Filtering Block List
- Block the subset of the Service Flow that matches a criteria entry on the URL Filtering Quarantine List
- Block the subset of the Service Flow that does not match a criteria entry on any of the URL Filtering lists, per the first bullet of [R80]

9.4 Malware Detection and Removal

Malware is defined as software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system.

Malware Detection and Removal is defined as the Security Function that determines whether a Service Flow, or subset of a Service Flow, contains Malware, and removes the Malware or Blocks the subset of the Service Flow containing the Malware. A typical use case is where a Subscriber wants all web e-mails and downloads to be checked, and, when Malware is detected, it is removed.

If a given Service Flow is encrypted, Malware Detection and Removal can only be *Enabled* if a Middlebox Security Function is also *Enabled*.

A Malware Detection and Removal Block List is a list of criteria entries used by the Malware Detection and Removal Security Function that Blocks the subset of the Service Flow that contains a match to one of the entries, i.e., Malware has been detected and the appropriate action taken (see [R86]). When this happens, the Subscriber is notified via the SEN. A Malware Detection and Removal Allow List[4] is a list of criteria entries used by the Malware Detection and Removal Security Function that Allows the subset of the Service Flow that contains a match to one of the entries. A Malware Detection and Removal Quarantine List is a list of criteria entries that are deemed suspicious, but the criteria entry has not been identified on the Malware Detection and Removal Block List. For a criteria entry on the Malware Detection and Removal Quarantine List, either the Malware is removed or the subset of the Service Flow containing the Malware is Blocked by the Malware Detection and Removal Security Function until further security checks can be done (see [R86]). When this happens, the Subscriber is notified via the SEN.

The lists are maintained by the Service Provider, with each criteria entry on a list being a hash.

Requirements related to the Malware Detection and Removal Block List, the Malware Detection and Removal Allow List and the Malware Detection and Removal Quarantine List are found in Section 7.1 of this document.

[4] There could be cases where a specific Subscriber might need to see certain malware coming through. One example is a company that is working on malware detection and removal technology.

[R82] When a Malware Detection and Removal Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the Malware Detection and Removal Block List, the Malware Detection and Removal Allow List and the Malware Detection and Removal Quarantine List.

Malware Detection: A Service Flow is scanned, using known signatures, to detect any Malware hidden in the Service Flow (e.g., antivirus scan on an e-mail). The Subscriber decides on the behavior to take when Malware is detected. A decision is made to either:

a) Block the Service Flow, or
b) Block the subset of the Service Flow containing the Malware and Allow the remainder of the Service Flow, or
c) Quarantine the Service Flow, or
d) Quarantine the subset of the Service Flow containing the Malware and Allow the remainder of the Service Flow, or
e) Remove the Malware from the Service Flow and Allow the remainder of the Service Flow.

For a limited set of cases determined by the Service Provider as potentially suspicious, behavioral analysis may be done. This consists of a simulated but monitored environment where a subset of the Service Flow is scanned, executed in an isolated and secure environment (referred to in the cybersecurity industry as a sandbox) and monitored for suspicious activities. Behavioral analysis can detect many of the yet unknown attacks, but it can be very compute intensive and is usually done when no known signature exists. If something dangerous is detected, a signature is created.

[R83] When a Malware Detection and Removal Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** describe which kind of detection (e.g., signature scan, behavioral analysis, or both) is performed.

For a given Service Flow, Malware Detection and Removal assembles the packets into the intended files, analyzes the files, removes any Malware, and then releases the files for packet forwarding. When the Malware Detection and Removal Security Function is applied to a given Service Flow, performance could be impacted.

If a subset of a Service Flow is encrypted and cannot be decrypted by the Middlebox Security Function, then Malware Detection and Removal cannot be performed for that subset of the Service Flow. The Service Provider and the Subscriber need to agree on the action to be taken when these kinds of events occur. See Requirements [R84], [R85], and [R86].

[R84] When a Malware Detection and Removal Security Function is included in a Service Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the Malware Detection and Removal Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R85] When a Malware Detection and Removal Security Function is included in a Service Policy for a given Service Flow, the Malware Detection and Removal Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the Malware Detection and Removal lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R86] When a Malware Detection and Removal Security Function is included in a Service Policy for a given Service Flow, and when a subset of the Service Flow is determined to either have Malware or look suspicious that it may have Malware, the Malware Detection and Removal Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Block the Service Flow
- Block the subset of the Service Flow containing the Malware and Allow the remainder of the Service Flow
- Quarantine the Service Flow
- Quarantine the subset of the Service Flow containing the Malware and Allow the remainder of the Service Flow
- Remove Malware from the Service Flow and Allow the Service Flow

Note that if removing Malware from the Service Flow is attempted and fails, then one of the other actions applies.

[R87] If a SEN is issued ([R23]), the Service Provider **MUST** report which action is taken for detected Malware and make it available to the Subscriber via the SEN (see Section 7.2).

Appendix C describes four examples of Malware Detection and Removal.

9.5 Data Loss Prevention

Data Loss Prevention (DLP) is a Security Function that determines whether a Service Flow, or subset of a Service Flow, contains confidential, sensitive, or important data, and prevents such data from being exfiltrated by people or systems either intentionally or unintentionally. This Security Function does not cover the case where the Subscriber no longer has access to the data.

A typical use case is where a Subscriber (e.g., Subject Actor) is concerned about unauthorized disclosure of information and wants a Service Flow checked for Personally Identifiable Information (PII) and Confidential or Proprietary Information (CPI). As an example, when detected such information can either be removed or the subset of the Service Flow containing such information could be Blocked. Data Loss Prevention may be required for compliance to a Subscriber's information security policy.

Personally Identifiable Information (PII) is defined in NIST SP 800-122 [63] as: "any information about an individual maintained by an agency, including (1) any information that can be used to distinguish or trace an individual's identity, such as name, social security number, date and place of birth, mother's maiden name, or biometric records; and (2) any other information that is linked or linkable to an individual, such as medical, educational, financial, and employment information."

In addition, NIST SP 800-122 [63] lists the following examples of PII:

- "Name, such as full name, maiden name, mother's maiden name, or alias
- Personal identification number, such as social security number (SSN), passport number, driver's license number, taxpayer identification number, patient identification number, and financial account or credit card number
- Address information, such as street address or email address
- Asset information, such as Internet Protocol (IP) or Media Access Control (MAC) address or other host-specific persistent static identifier that consistently links to a particular person or small, well-defined group of people
- Telephone numbers, including mobile, business, and personal numbers
- Personal characteristics, including photographic image (especially of face or other distinguishing characteristic), x-rays, fingerprints, or other biometric image or template data (e.g., retina scan, voice signature, facial geometry)
- Information identifying personally owned property, such as vehicle registration number or title number and related information
- Information about an individual that is linked or linkable to one of the above (e.g., date of birth, place of birth, race, religion, weight, activities, geographical indicators, employment information, medical information, education information, financial information)."

Confidential or Proprietary Information (CPI) is defined as any information that could harm an organization if made public or used against them.

Basic information about an individual (e.g., Subject Actor) could be required in some contexts, e.g., as an authentication factor when logging into certain web sites. The same information could be deemed a higher risk in other contexts (e.g., an e-mail or e-mail attachment). The Subscriber determines: a) the details of each Data Loss Prevention Security Function Policy, and b) which Service Policy includes the Data Loss Prevention Security Function. Context is normally considered when making this determination.

When the Data Loss Prevention Security Function is included in a Service Policy for a given Service Flow, the Subscriber and Service Provider agree on the details, i.e., which techniques are used to check the Service Flow and which information to look for. An example of a technique could include checking all text in an e-mail for PII and opening an attached image and scanning it using Optical Character Recognition (OCR) technology and then inspecting the derived characters to determine if the image contains any PII or CPI.

The Data Loss Prevention Security Function needs to be able to identify PII and CPI in a Service Flow. Both structured (information in fixed fields) and unstructured data could be included within a Service Flow. Information could be embedded in a text file or an image file so each would require different scanning techniques to forward only those files that that are deemed low risk. The technique that is used for checking a Service Flow is determined by agreement between the Subscriber and Service Provider as part of the Service and is not in scope for this Standard.

When the Data Loss Prevention Security Function is applied to a given Service Flow, performance could be impacted.

The Subscriber and Service Provider agree on the format of the criteria entries used to identify PII or CPI; often, this uses a text string pattern. The Service Provider typically provides the standard set of criteria entries, and the Subscriber adds or modifies that set for a given Data Loss Prevention Security Function Policy. A criteria entry could be simple (e.g., a single, structured identifier for a 9-digit Social Security number), or more complex (e.g., a combination of information such as bank account number, social security number and username). Note that a criteria entry could include a Boolean combination of criteria entries, e.g., A AND X. The criteria entry for the Data Loss Prevention Block List consists of characters from a character set agreed between the Subscriber and Service Provider.

A Data Loss Prevention Block List is a list of criteria entries used by the Data Loss Prevention Security Function to Block the subset of the Service Flow that contains a match to one of the entries. The Subscriber may be notified of the Block, including the cause. An example is when a Subscriber wants to Block certain Service Flows that contain PII or CPI. A Data Loss Prevention Allow List is a list of criteria entries used by the Data Loss Prevention Security Function to Allow the subset of the Service Flow that contains a match to one of the entries. The Subscriber may be notified of such events. An example is when a Subscriber wants to get notified when certain Service Flows contain PII or CPI but Allow the subset of the Service Flow through. A SEN could be used to alert the Subscriber, for example with the Notification Action Modifier (see Section 7.1.7) set, as part of the criteria entry in the Data Loss Prevention Allow List. A Data Loss Prevention Quarantine List is a list of criteria entries that are deemed suspicious, e.g., an inexact, but close match to a criteria entry on the Data Loss Prevention Block List, or a suspicious context, e.g., a list of passport numbers sent in an e-mail from a specific Subject Actor, as defined in Zero Trust Framework for MEF Services (MEF 118 [5]) to a Target Actor, as defined in Zero Trust Framework for MEF Services (MEF 118 [5]). The Data Loss Prevention Security Function Blocks the subset of the Service Flow containing a match to a criteria entry on the Data Loss Prevention Quarantine List, and possibly also due to the Subject Actor and/or Target Actor. The Subscriber may be notified of the Block, including the cause, and then could possibly Allow the subset of the Service Flow (e.g., third party approval might be needed to override the Block). When PII or CPI has been detected in a subset of the Service Flow, the Data Loss Prevention Security Function takes the appropriate action - see [R91].

The Subscriber referred to in the above paragraph can be informed either via a SEN (if the reason for the Block triggers a SEN), or via immediate communication to the client (Subject Actor) depending on the Service Policy. For notification by the SEN, see [R23] for conditions that trigger a SEN.

Requirements related to the Data Loss Prevention Block List, the Data Loss Prevention Allow List, and the Data Loss Prevention Quarantine List are found in Section 7.1 of this document.

[R88] When a Data Loss Prevention Security Function is included in a Service Policy, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the Data Loss Prevention Block List, the Data Loss Prevention Allow List, and the Data Loss Prevention Quarantine List.

[R89] When a Data Loss Prevention Security Function Policy is included in a Service Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the Data Loss Prevention lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R90] When a Data Loss Prevention Security Function Policy is included in a Service Policy for a given Service Flow, the Data Loss Prevention Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the Data Loss Prevention lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R91] When a Data Loss Prevention Security Function Policy is included in a Service Policy for a given Service Flow, and when a subset of the Service Flow is determined to either have PII or CPI, the Data Loss Prevention Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the subset of the Service Flow that matches a criteria entry that is on the Data Loss Prevention Allow List
- Allow the subset of the Service Flow that does not match a criteria entry on any of the Data Loss Prevention lists, per the second bullet of [R90]
- Block the Service Flow that matches a criteria entry on the Data Loss Prevention Block List
- Block the subset of the Service Flow that matches a criteria entry on the Data Loss Prevention Block List and Allow the remainder of the Service Flow
- Block the subset of the Service Flow that matches a criteria entry on the Data Loss Prevention Quarantine List and Allow the remainder of the Service Flow
- Block the subset of the Service Flow that does not match a criteria entry on any of the Data Loss Prevention lists, per the first bullet of [R90]
- Remove the PII and/or CPI from the subset of the Service Flow that matches a criteria entry on either the Data Loss Prevention Block List or the Data Loss Prevention Quarantine List and Allow the remainder of the Service Flow

Note that if removing PII or CPI from the Service Flow is attempted and fails, then one of the other actions indicated in [R91] applies.

The following requirements deal with the situation where a subset of a Service Flow (e.g., an Object such as a file attachment) cannot be scanned (e.g., it is encrypted or password-protected).

[R92] When a Data Loss Prevention Security Function Policy is included in a Service Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that cannot be scanned:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

The Subscriber and Service Provider need to agree on the behavior for such cases.

[R93] When a Data Loss Prevention Security Function Policy is included in a Service Policy for a given Service Flow, the Data Loss Prevention Security Function

MUST perform one of the following actions for a subset of the Service Flow that cannot be scanned:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

If a given Service Flow is encrypted, the Data Loss Prevention Security Function requires the Middlebox Security Function to first decrypt the Service Flow so the Data Loss Prevention Security Function can be performed. If a subset of the Service Flow cannot be decrypted by the Middlebox Security Function, then Data Loss Prevention cannot be performed for that subset of the Service Flow. The Service Provider and the Subscriber need to agree on the action to be taken when these kinds of events occur. Such action could involve Blocking or Allowing the subset of the Service Flow that cannot be decrypted.

9.6 DNS Security Functions

DNS Security Functions apply to Service Flows carrying DNS messages. There are two DNS Security Functions specified in this Standard.

- DNS Protocol Filtering - see Section 9.6.1.
- Protective DNS - see Section 9.6.2.

There are two scenarios to consider.

Scenario A: The Service Provider does not host a DNS server. In this case, when DNS Protocol Filtering, Section 9.6.1, is *Enabled*, only unencrypted DNS messages (Port 53) can be checked. Encrypted DNS messages are handled as any other packets in the Service Flow.

Scenario B: The Service Provider hosts a DNS server, and the Subscriber uses this DNS server for queries. Protective DNS cover this scenario, and can check encrypted DNS messages using DNS over HTTPS (DoH), as specified in RFC 8484 [49]; and/or DNS over TLS (DoT), as specified in RFC 7858 [44]. DNS messages involving DNSSEC can also be checked. Unencrypted DNS queries can also be checked. See Section 9.6.2.

TOR, as specified in RFC 7686 [43], is a special case that does not use the DNS infrastructure for resolving domains associated with the '.onion' top level domain.

9.6.1 DNS Protocol Filtering

DNS Protocol Filtering is defined as the Security Function that determines whether a Service Flow, or subset of a Service Flow, contains Domain Name System (DNS) messages that are to be Allowed or Blocked. DNS Protocol Filtering applies to the DNS protocol operating over port 53/TCP and port 53/UDP. DNS [38] messages are specified in RFC 1035 [6], RFC 1996 [8] and RFC 6895 [38].

An example of DNS Protocol Filtering is shown in Figure 7**Error! Reference source not found.**.

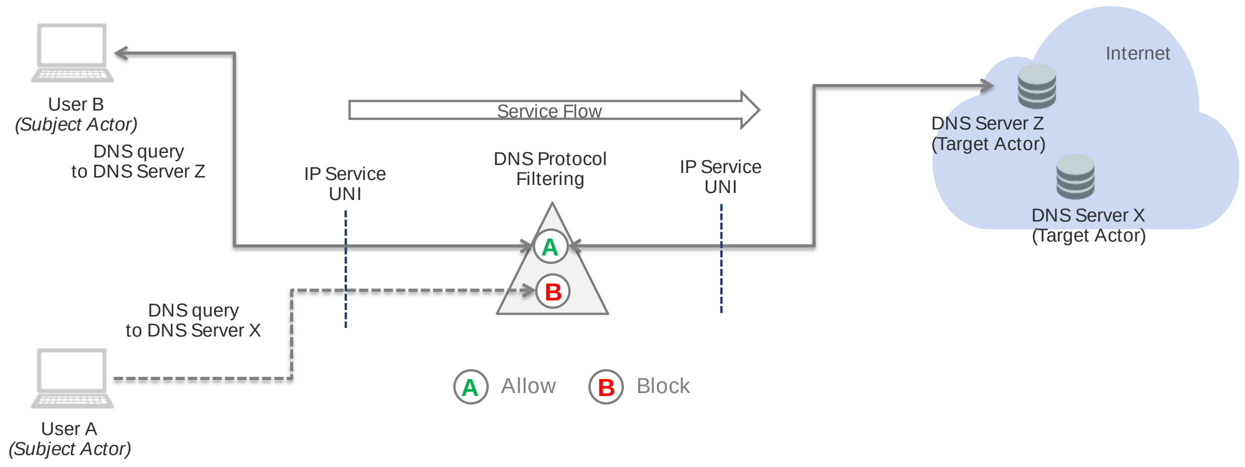


Figure 7 - Example of DNS Protocol Filtering

In Figure 7**Error! Reference source not found.**, DNS Server Z (Target Actor) is on the DNS Protocol Filtering Allow List and DNS Server X (Target Actor) is on the DNS Protocol Filtering Block List. When a DNS Query is sent to DNS Server Y (Target Actor) from User B (Subject Actor), it is Allowed by the DNS Protocol Filtering Security Function. When a DNS Query is sent to DNS Server X from User A (Subject Actor), it is Blocked by the DNS Protocol Filtering Security Function. Depending on prior agreement of the Service Provider and Subscriber, the Service Provider may choose to provide more details about the reasons for the Block, using an appropriate mechanism, e.g., a DNS response code.

A DNS Protocol Filtering Block List is a list of criteria entries used by the DNS Protocol Filtering Security Function to Block the subset of the Service Flow that contains a match to one of the entries, e.g., a DNS Query to a particular DNS server is Blocked. A DNS Protocol Filtering Allow List is a list of criteria entries used by the DNS Protocol Filtering Security Function to Allow the subset of the Service Flow that contains a match to one of the entries, e.g., a DNS Query to a particular DNS server is Allowed. The DNS Protocol Filtering Quarantine List is a list of criteria entries that are not on the DNS Protocol Filtering Block List but are deemed suspicious. The subset of the Service Flow that contains a match to one of the entries on the DNS Protocol Filtering Quarantine List is Blocked by the DNS Protocol Filtering Security Function. The three lists are maintained by the Service Provider.

[R94] Each criteria entry on a DNS Protocol Filtering List **MUST** include the fields listed in Table 6.

Additional fields can be agreed between the Subscriber and Service Provider.

Field	Description
SAV4	List of IPv4 source addresses. Any could be used to indicate that any source IPv4 address is on the list. The list can be empty.
DAV4	List of IPv4 destination addresses. Any could be used to indicate that any destination IPv4 address is on the list. The list can be empty.
SAV6	List of IPv6 source addresses. Any could be used to indicate that any source IPv6 address is on the list. The list can be empty.
DAV6	List of IPv6 destination addresses. Any could be used to indicate that any destination IPv6 address is on the list. The list can be empty.
DNS Message Type	One of the DNS message types, as specified in RFC 6895 [38]. Examples of a valid entry are DNS Query, DNS Response, DNS Update.

Table 6 - Criteria entry fields for DNS Protocol Filtering

Requirements related to the DNS Protocol Filtering Block List, the DNS Protocol Filtering Allow List and the DNS Protocol Filtering Quarantine List are found in Section 7.1 of this document.

[R95] When a DNS Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the DNS Protocol Filtering Block List, the DNS Protocol Filtering Allow List and the DNS Protocol Filtering Quarantine List.

[R96] When a DNS Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the DNS Protocol Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R97] When a DNS Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the DNS Protocol Filtering Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the DNS Protocol Filtering lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R98] When a DNS Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, the DNS Protocol Filtering Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the subset of the Service Flow that matches a criteria entry on the DNS Protocol Filtering Allow List
- Allow the subset of the Service Flow that does not match a criteria entry on any of the DNS Protocol Filtering lists, per the second bullet of [R97]
- Block the subset of the Service Flow that matches a criteria entry on the DNS Protocol Filtering Block List
- Block the subset of the Service Flow that matches a criteria entry on the DNS Protocol Filtering Quarantine List
- Block the subset of the Service Flow that does not match a criteria entry on any of the DNS Protocol Filtering lists, per the first bullet of [R97]

[R99] The DNS Protocol Filtering Security Function **MUST** be capable of informing the DNS client in the Subscriber's network immediately of any DNS message failure.

[D10] When a DNS Protocol Filtering Security Function Policy is included in a Security Policy for a given Service Flow, and when a DNS message in that Service Flow is Blocked, the DNS Protocol Filtering Security Function **SHOULD** send an appropriate DNS response code, per Section 2.3 of RFC 6895 [38], to the DNS client in the Subscriber's network.

This document does not mandate the method for informing the DNS client. One example could be that a DNS query is re-directed by the Service Provider to a web page that gives the reason(s) why the DNS query is Blocked. The method used is based on agreement of the Service Provider and Subscriber.

9.6.2 Protective DNS

With respect to secure DNS, MEF 88 considers only DNS over UDP/TCP ports 53, which is not necessarily reflective of modern DNS service design. Furthermore, it predominantly considers DNS as a mechanism to determine what to do with Application Flows, rather than as a service in and of itself. Protective DNS (PDNS) [67] gives customers and users the option to consider "is this DNS request allowed?", rather than simply "is the user allowed to communicate with the endpoint to which this DNS points?".

[R100] Implementations **MUST** be capable of inspecting and influencing DNS either directly or indirectly even before any future Service Flow is established.

[R101] The contents of DNS requests from clients and the resultant responses from DNS servers **MUST** themselves be evaluated in the same manner as other Service Flows.

Protective DNS is the Security Function that examines DNS request and response records, and which can allow/block/alter them to protect the recipient.

With respect to how DNS is serviced within the context of this document, all or a subset of the options can be considered:

- Secure DNS Proxy function – allows clients to leverage DNS over HTTPS (DoH) [49] and DNS over TLS (DoT) [44] for upstream resolution irrespective of whether a full end to end DoH/DoT flow can be established

- DNS Resolution function (see Section 7.4) supports resolution of all DNS zones on behalf of client systems using traditional services offered over UDP/TCP ports 53

- Middlebox Security Function – inspects DNS requests and responses in-line by examining a Service Flow without the need for the Middlebox Security Function to be explicitly configured by the client as recursive/authoritative DNS service for the DNS zone

The Protective DNS Security Function can be applied in all three cases; however, the scope of interaction will necessarily vary depending on the implementation.

Figure 8 below illustrates a Protective DNS architecture.

Editor Note 5: *In one of our weekly calls leading up to the Q4 2023 meeting, we discussed updating Figure 8. Neil will provide a contribution on this, which needs to also address the third bullet above, which may need to change depending on changes to Figure 8.*

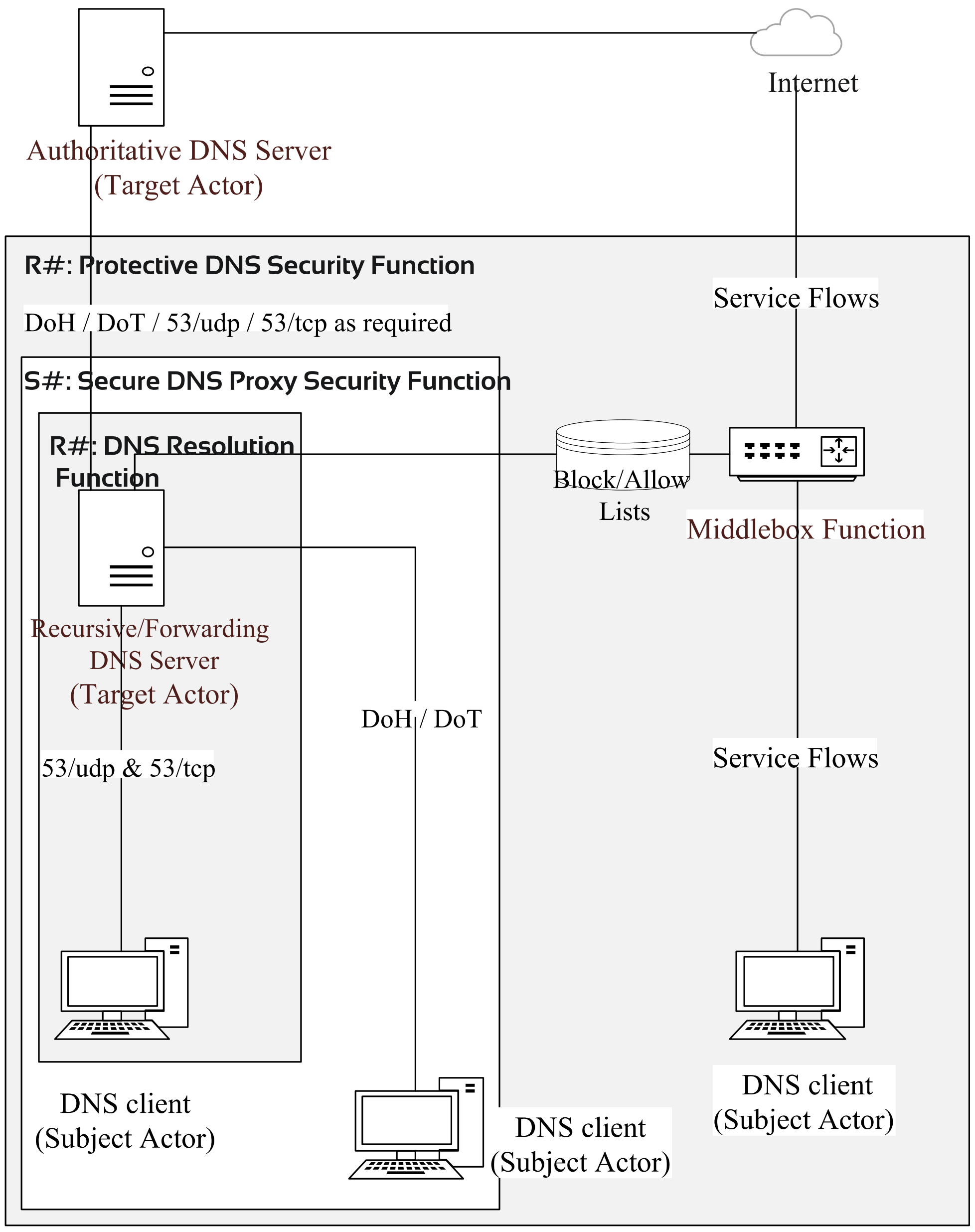


Figure 8 - Protective DNS Architecture Example

The following requirements relate to DNS resource records, as defined by IETF in the indicated RFCs.

[R102] When a Protective DNS Security Function Policy is included in a Service Policy for a given Service Flow, the Service Provider **MUST** support the ability to perform resolution for the following DNS resource record types.

- SOA (Start of Authority) – specifies authoritative information about a DNS zone, including the primary name server, the email of the domain administrator, the domain serial number, and several timers relating to refreshing the zone (RFC 1035 [6])
- NS (Name Server) – delegates a DNS zone to use the given authoritative name servers (RFC 1035 [6])
- A – Returns a 32-bit IPv4 address, most commonly used to map hostnames to an IP address of the host, but it is also used for DNSBLs, storing subnet masks in RFC 1101, etc. (RFC 1035 [6])
- AAAA – Returns a 128-bit IPv6 address, most commonly used to map hostnames to an IP address of the host (RFC 3596 [15])
- CNAME (Canonical Name) – alias of one name to another: the DNS lookup will continue by retrying the lookup with the new name (RFC 1035 [6])
- MX (Mail Exchange) – maps a domain name to a list of message transfer agents for that domain (RFC 1035 [6])
- SRV (Service) – delegates a DNS zone to use the given authoritative name servers for the purpose of service lookups, e.g., SIP, XMPP (RFC 2782 [10])
- PTR (Pointer) – pointer to a canonical name. Unlike a CNAME, DNS processing stops and just the name is returned. The most common use is for implementing reverse DNS lookups, but other uses include such things as DNS-SD (RFC 6763 [36])

[R102] means that Protective DNS is required to support the ability to limit resolution based on the each of the above DNS resource record types and any of the corresponding values for the given record.

[D11] When a Protective DNS Security Function Policy is included in a Service Policy for a given Service Flow, the Service Provider **SHOULD** support the ability to perform resolution for the following DNS resource record types.

- DLV (DNSSEC Lookaside Validation) – for publishing DNSSEC trust anchors outside of the DNS delegation chain. Uses the same format as the DS record. RFC 5074 [26] describes a way of using these records.
- DNSKEY – used to identify the DNSSEC signing key of a delegated zone (public signing key) (RFC 4034 [19])
- DS (Delegation Signer) – used to identify the DNSSEC signing key of a delegated zone (hash of DNSKEY record) (RFC 4034 [19])
- NSEC (Next Secure) – part of DNSSEC—used to prove a name does not exist. Uses the same format as the (obsolete) NXT record (RFC 4034 [19])
- NSEC3 (Next Secure 3) – an extension to DNSSEC that allows proof of nonexistence for a name without permitting zone walking (RFC 5155 [27])
- NSEC3PARAM (Next Secure 3 Parameter) – parameter record for use with NSEC3 (RFC 5155 [27])
- OPENPGPKEY – A DNS-based Authentication of Named Entities (DANE) method for publishing and locating OpenPGP public keys in DNS for a specific email address using an OPENPGPKEY DNS resource record (RFC 7929 [45])
- RRSIG (Resource Record Digital Signature) – signature for a DNSSEC-secured record set. Uses the same format as the SIG record. (RFC 4034 [19])
- RP (Responsible Person) – information about the responsible person(s) for the domain. Usually an email address with the @ replaced by a . (RFC 1183 [7])
- SMIMEA (S/MIME Certificate Association) - associates an S/MIME certificate with a domain name for sender authentication. (RFC 8162 [46])
- SSHFP (Secure Shell Fingerprint) – used for publishing SSH public host key fingerprints in the DNS, to aid in verifying the authenticity of the host. RFC 6594 [34] defines ECC SSH keys and SHA-256 hashes. See the IANA SSHFP RR parameters registry for details. (RFC 4255 [23])
- TA (DNSSEC Trust Authorities) – part of a deployment proposal for DNSSEC without a signed DNS root. See the IANA database and Weiler Spec for details. Uses the same format as the DS record.
- TKEY (Transaction Key) – a method of providing keying material to be used with TSIG that is encrypted under the public key in an accompanying KEY RR. (RFC 2930 [12])
- TLSA (TLSA certificate association) - A record for DANE. RFC 6698 [35] defines "The TLSA DNS resource record is used to associate a TLS server certificate or public key with the domain name where the record is found, thus forming a 'TLSA certificate association'".

- TSIG (Transaction Signature) – used to authenticate dynamic updates as coming from an approved client, or to authenticate responses as coming from an approved recursive name server similar to DNSSEC. (RFC 8945 [50])
- URI (Uniform Resource Identifier) – can be used for publishing mappings from hostnames to URIs. (RFC 7553 [42])
- ZONEMD (Message Digests for DNS zones) – provides a cryptographic message digest over DNS zone data at rest. (RFC 8976 [51])

[D11] means that it is recommended that the Protective DNS Security Function supports the ability to limit resolution based on the each of the above DNS resource record types and any of the corresponding values for the given record.

While not all these record types may be in common usage, it is likely that usage will increase, particularly for DNSSEC related records. These record types also make good candidates for determining malicious actors and flows based on the values encapsulated in the response. For example, SSHFP could be used to identify on-path attacks against SSH services.

Any obsolete record types as well as any other types not explicitly referenced in either [R102] or [D11] are out of scope of this document.

[R103] For records where resolution of the record type is intended to be limited, inspection of responses **MUST** at minimum consist of comparing the record value (by IP and/or hostname) using the same source of truth, and with the same policies that would otherwise be applied via the traffic inspection within the Middlebox Security Function.

Inspection could optionally consider other properties of the record value at the implementors discretion.

The following requirements apply.

[R104] Each criteria entry on a Protective DNS List **MUST** include the fields listed in Table 7 for the appropriate DNS resource record type identified in [R102].

Additional fields can be agreed between the Subscriber and Service Provider.

[D12] A Protective DNS Security Function **SHOULD** support the ability to inspect DNS messages encrypted with DNS over HTTPS (DoH), per RFC 8484 [49].

[D13] A Protective DNS Security Function **SHOULD** support the ability to inspect DNS messages encrypted with DNS over TLS (DoT), per RFC 7858 [44].

Editor Note 6: *At the Q4 2023 editing session, it was agreed to add in a recommendation for DNSSEC, (see [D14] below) since many of the recommended DNS resource record types are part of DNSSEC. Comments are welcome.*

[D14] A Protective DNS Security Function **SHOULD** support the ability to inspect DNS messages encrypted with DNSSEC, per RFCs 4033 [18], 4034 [19], and 4035 [20].

DNS resource record type	**Criteria entries**
SOA	parameters listed in section 3.3.13 of RFC 1035 [6]
NS	parameters listed in section 3.3.11 of RFC 1035 [6]
A	parameters listed in section 3.4.1 of RFC 1035 [6]
AAAA	parameters listed in section 2 of RFC 3596 [15]
CNAME	parameters listed in section 3.3.1 of RFC 1035 [6]
MX	parameters listed in section 3.3.9 of RFC 1035 [6]
SRV	parameters listed in RFC 2782 [10]
PTR	parameters listed in RFC 6763 [36]

Table 7 - Criteria entries for the mandatory DNS resource records

[D11] lists many DNS Resource Record types that are recommended, with references to RFCs, as appropriate. The parameters in those RFCs need agreement as criteria entries.

Editor Note 7: Do we need to put all 17 of the recommended DNS resource record types into a separate table, as done in Table 7 for the mandatory DNS resource record types, or is the above paragraph sufficient?

A Protective DNS Block List consists of a list of criteria entries used by the Protective DNS Security Function to Block the subset of the Service Flow that contains a match to one of the entries. A Protective DNS Allow List consists of a list of criteria entries used by the Protective DNS Security Function to Allow the subset of the Service Flow that contains a match to one of the entries. A Protective DNS Quarantine List is a list of criteria entries that are not on the Protective DNS Block List but are deemed suspicious. The subset of the Service Flow that contains a match to one of the entries on the Protective DNS Quarantine List is Blocked by the Protective DNS Security Function. A Protective DNS Security Function Supported List consists of two lists that are agreed to be processed by the Protective DNS Security Function: a) a list of the DNS encryption methods (unencrypted, DNS over HTTPS, DNS over TLS, DNSSEC, Other), and b) a list of the criteria entries (parameters) for each of the supported DNS resource record types. A Protective DNS Security Function Unsupported List consists of two lists that are agreed to be not processed by the Protective DNS Security Function: a) a list of the DNS encryption methods (unencrypted, DNS over HTTPS, DNS over TLS, DNSSEC, Other), and b) a list of the criteria entries (parameters) for each of the unsupported DNS resource record types. The five lists are maintained by the Service Provider.

Editor Note 8: There's a question whether a Quarantine List should be part of Protective DNS. All good comments are welcome.

Editor Note 9: The above text re: Supported and Unsupported Lists for Protective DNS was agreed to be changed during the Q4 editing session in Dallas, October 2023. We may want to instead tie these lists together, encryption method and DNS resource record parameters. For example, should we have: a) two separate PDNS Supported Lists (one list for encryption methods supported, and one list *for DNS resource records supported), or b) one PDNS Supported List, each containing two sub-lists, (one for encryption methods supported, and one PDNS Supported List for DNS resource records supported), or c) one PDNS supported List with the encryption method and DNS resource record tied together? Please review and provide careful comments.*

Requirements related to the Protective DNS Block List, the Protective DNS Allow List, the Protective DNS Quarantine List, the Protective DNS Supported List, and the Protective DNS Unsupported List are found in Section 7.1 of this document.

[R105] When a Protective DNS Security Function is included in a Service Policy, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the Protective DNS Block List, the Protective DNS Allow List, the Protective DNS Quarantine List, the Protective DNS Supported List, and the Protective DNS Unsupported List.

Editor Note 10: Consider whether to add a flow diagram showing the different supported/unsupported encryption methods and DNS resource records to show what is allowed, blocked. Contributions are welcome.

When a Protective DNS Security Function Policy is included in a Policy for a given Service Flow, there are four cases to consider.

- Case 1: A subset of the Service Flow matches a criteria entry on the Protective DNS Security Function Block List. The Protective DNS Security Function Blocks this subset.
- Case 2: A subset of the Service Flow matches a criteria entry on the Protective DNS Security Function Allow List and on the Protective DNS Security Function Supported List. The Protective DNS Security Function processes this subset.
- Case 3: A subset of the Service Flow matches a criteria entry on the Protective DNS Security Function Allow List and on the Protective DNS Security Function Unsupported List. The Protective DNS Security Function passes this subset through, unchanged.
- Case 4: A subset of the Service Flow does not match an entry on any of the lists. This is the 'no-match' case, and this is covered by the agreement of the Subscriber and Service Provider to either pass this through the Protective DNS Security Function unchanged or to Block it.

The following requirements specify the Protective DNS Security Function behavior for these cases.

[R106] When a Protective DNS Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Protective DNS Security Function Block List **MUST** be Blocked.

[R107] When a Protective DNS Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Protective DNS Security Function Allow List and on the Protective DNS Security Function Supported List **MUST** be processed.

[R108] When a Protective DNS Security Function Policy is included in a Security Policy for a given Service Flow, the subset of the Service Flow that matches a criteria entry on the Protective DNS Security Function Allow List and on the Protective DNS Security Function Unsupported List **MUST** be passed through the Protective DNS Security Function without change.

It is possible that a subset of the Service Flow does not match a criteria entry on any of these lists. Requirements [R109] and [R110] cover this gap.

[R109] When a Protective DNS Security Function Policy is included in a Service Policy for a given Service Flow, the Service Provider **MUST** support both of the following actions for a subset of the Service Flow that does not match a criteria entry on any of the Protective DNS lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R110] When a Protective DNS Security Function Policy is included in a Service Policy for a given Service Flow, the Protective DNS Security Function **MUST** perform one of the following actions for each subset of the Service Flow that does not match a criteria entry on any of the Protective DNS lists:

- Block the subset of the Service Flow
- Allow the subset of the Service Flow

[R111] When a Protective DNS Security Function Policy is included in a Service Policy for a given Service Flow, and when a subset of the Service Flow is transporting DNS messages, the Protective DNS Security Function **MUST** perform one of the following actions, based on agreement between the Service Provider and the Subscriber:

- Allow the DNS message that matches a criteria entry that is on the Protective DNS Allow List
- Allow the DNS message that does not match a criteria entry on any of the Protective DNS lists, per the second bullet of [R110]
- Block the DNS message that matches a criteria entry on the Protective DNS Block List
- Block the DNS message that matches a criteria entry on the Protective DNS Quarantine List
- Block the DNS message that does not match a criteria entry on any of the Protective DNS lists, per the first bullet of [R110]
- Alter the DNS message response to protect the recipient

[R112] The Protective DNS Security Function **MUST** be capable of informing the Subscriber and/or the client in the Subscriber's network immediately when a DNS message is either Blocked or altered.

10 Security Function Policy

A Security Function Policy is defined in this Standard as a set of parameters that control the externally visible behavior for a given Security Function. This set of parameters needs to be agreed between the Subscriber and Service Provider. A Security Function Policy is an Atomic Policy, as defined in SASE Service Attributes and Service Framework Standard (MEF 117 [4]), that fits within a Policy for a given Service. It is up to the service-specific Standard to define the policy hierarchy. Unless otherwise specified in this document, how these Policies are assigned and utilized in a given Service is up to that Service and thus beyond the scope of this document.

A Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, Quarantined, or passed through by a given Security Function. Each Security Function Policy is agreed between the Subscriber and Service Provider.

How such an agreement is reached is outside the scope of this document. Some examples of how agreement could be reached are given below, but this is not an exhaustive list.

- The Service Provider mandates a particular value and purchase of the Service implies implicit agreement.
- The Subscriber selects from a set of options specified by the Service Provider.
- The Subscriber requests a particular value, and the Service Provider indicates whether they accept it.
- The Subscriber and the Service Provider negotiate to reach a mutually acceptable value.

How the agreement is reached, and the specific values agreed, might have an impact on the price of the service or on other business or commercial aspects of the relationship between the Service Provider and the Subscriber; this is outside the scope of this document.

Security Function Policy parameters describe the externally visible behavior of a specific Security Function; they do not constrain how the Security Function is implemented by a Service Provider, or how the Subscriber implements their network.

The Subscriber and the Service Provider agree upon the initial value for each parameter in advance of the service deployment. The Subscriber and the Service Provider may subsequently agree on changes to the values of certain parameters. This document does not constrain how such agreement is reached; for example, if the Service Provider allows the Subscriber to select an initial value from a pre-determined set of values, they might further allow them to change their selection at any time during the lifetime of the service.

[R113] When a Security Function Policy is in force for Service Flows, the Service Provider **MUST** inform the Subscriber of any expected impact to the service performance metrics.

A Subscriber could have more than one Security Function Policy for a given Security Function. Within the scope of the Subscriber's service, each Security Function Policy has a unique identifier.

10.1 Security Policy Identifier Parameter

This document does not require a Service to use a Security Policy. When a Service does use a Security Policy, then the requirements in this section apply.

Security Policy Identifier is a parameter of a Security Policy that provides a unique identifier for a given Security Policy.

[R114] When a Service uses a Security Policy, the *Security Policy Identifier* **MUST** be an Identifier String.

[R115] When a Service uses a Security Policy, each *Security Policy Identifier* **MUST** be unique among all Security Policies for a given Service instance.

10.2 Security Function Policy Identifier Parameter

Security Function Policy Identifier is a parameter of a Security Function Policy that provides a unique identifier for a given Security Function Policy.

[R116] The Security Function Policy Identifier **MUST** be an Identifier String.

[R117] Each *Security Function Policy Identifier* **MUST** be unique among all Security Function Policies for a given Service instance.

10.3 Middlebox Security Function Policy

The Middlebox Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is decrypted/re-encrypted, Blocked, or passed through unchanged by the Middlebox Security Function, which is described in more detail in Section 8.

[R118] Each Middlebox Security Function Policy **MUST** contain the parameters listed in Table 8.

Additional parameters can be agreed between the Subscriber and Service Provider.

Middlebox Security Function Policy Parameter	Description
Policy ID	The identifier for a given Middlebox Security Function Policy
Supported List	List of criteria entries in the Supported List for that Middlebox Security Function Policy
Unsupported List	List of criteria entries in the Unsupported List for that Middlebox Security Function Policy
Block List	List of criteria entries in the Block List for that Middlebox Security Function Policy
Allow List	List of criteria entries in the Allow List for that Middlebox Security Function Policy
No-match	The action that the Middlebox Security Function takes when there is no match. The possible values are Allow or Block.
List of Trusted CA's	List of the Subscriber's trusted CAs for that Middlebox Security Function Policy
Invalid server certificate behavior	Behavior of the Middlebox Security Function for that Middlebox Security Function Policy when the target server certificate is invalid. The possible values are Allow or Block.
Unencrypted packets	The action that the Middlebox Security Function takes when there are unencrypted packets in the Service Flow. The possible values are Allow or Block.

Table 8 - Middlebox Security Function Policy Parameters

10.4 IP, Port and Protocol Filtering (IPPF) Security Function Policy

The IP, Port and Protocol Filtering (IPPF) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the IPPF Security Function, which is described in more detail in Section 9.1.

[R119] Each IP, Port and Protocol Filtering Security Function Policy **MUST** contain the parameters listed in Table 9.

Additional parameters can be agreed between the Subscriber and Service Provider.

IPPF Security Function Policy Parameter	Description
Policy ID	The identifier for a given IPPF Security Function Policy
Block List	List of criteria entries in the Block List for the IPPF Security Function Policy
Allow List	List of criteria entries in the Allow List for the IPPF Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the IPPF Security Function Policy
No-match	The action that the IPPF Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the IPPF security threat database

Table 9 - IP, Port and Protocol Filtering Security Function Policy Parameters

10.5 Domain Name Filtering (DNF) Security Function Policy

The Domain Name Filtering (DNF) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the DNF Security Function, which is described in more detail in Section 9.2.

[R120] Each Domain Name Filtering Security Function Policy **MUST** contain the parameters listed in Table 10.

Additional parameters can be agreed between the Subscriber and Service Provider.

DNF Security Function Policy Parameter	**Description**
Policy ID	The identifier for a given DNF Security Function Policy
Block List	List of criteria entries in the Block List for the DNF Security Function Policy
Allow List	List of criteria entries in the Allow List for the DNF Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the DNF Security Function Policy
No-match	The action that the DNF Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the DNF security threat database

Table 10 - Domain Name Filtering Security Function Policy Parameters

10.6 URL Filtering (URLF) Security Function Policy

The URLF Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the URLF Security Function, which is described in more detail in Section 9.3.

[R121] Each URL Filtering Security Function Policy **MUST** contain the parameters listed in Table 11.

Additional parameters can be agreed between the Subscriber and Service Provider.

URLF Security Function Policy Parameter	**Description**
Policy ID	The identifier for a given URLF Security Function Policy
Block List	List of criteria entries in the Block List for the URLF Security Function Policy
Allow List	List of criteria entries in the Allow List for the URLF Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the URLF Security Function Policy
No-match	The action that the URLF Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the URLF security threat database

Table 11 - URL Filtering Security Function Policy Parameters

10.7 Malware Detection and Removal (MD+R) Security Function Policy

The Malware Detection and Removal (MD+R) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the MD+R Security Function, which is described in more detail in Section 9.4.

[R122] Each Malware Detection and Removal Security Function Policy **MUST** contain the parameters listed in Table 12.

Additional parameters can be agreed between the Subscriber and Service Provider.

MD+R Security Function Policy Parameter	Description
Policy ID	The identifier for a given MD+R Security Function Policy
Block List	List of criteria entries in the Block List for the MD+R Security Function Policy
Allow List	List of criteria entries in the Allow List for the MD+R Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the MD+R Security Function Policy
No-match	The action that the MD+R Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the MD+R security threat database
Detection Type	Signature scan, behavior analysis, or both
Malware removal behavior	As specified in [R86]. Allowed values are: Block the Service Flow, Block the subset of the Service Flow, Quarantine the Service Flow, Quarantine the subset of the Service Flow, or Remove the Malware and Allow the rest of the Service Flow.

Table 12 - Malware Detection and Removal Security Function Policy Parameters

10.8 Data Loss Prevention (DLP) Security Function Policy

The Data Loss Prevention (DLP) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the DLP Security Function, which is described in more detail in Section 9.5.

[R123] Each Data Loss Prevention Security Function Policy **MUST** contain the parameters listed in Table 13.

Additional parameters can be agreed between the Subscriber and Service Provider.

DLP Security Function Policy Parameter	Description
Policy ID	The identifier for a given DLP Security Function Policy
Block List	List of criteria entries in the Block List for the DLP Security Function Policy
Allow List	List of criteria entries in the Allow List for the DLP Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the DLP Security Function Policy
No-match	The action that the DLP Security Function takes when there is no match. The possible values are Allow or Block.
No-scan	The action that the DLP Security Function takes when a subset of the Service Flow cannot be scanned. The possible values are Allow or Block.
PII, CPI removal behavior	As specified in [R91]. Allowed values are: Block the Service Flow, Block the subset of the Service Flow, or Remove the PII/CPI and Allow the rest of the Service Flow.

Table 13 - Data Loss Prevention Security Function Policy Parameters

10.9 DNS Protocol Filtering (DPF) Security Function Policy

The DNS Protocol Filtering (DPF) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the DPF Security Function, which is described in more detail in Section **Error! Reference source not found.**.

[R124] Each DNS Protocol Filtering Security Function Policy **MUST** contain the parameters listed in **Error! Reference source not found.**.

Additional parameters can be agreed between the Subscriber and Service Provider.

DPF Security Function Policy Parameter	Description
Policy ID	The identifier for a given DPF Security Function Policy
Block List	List of criteria entries in the Block List for the DPF Security Function Policy
Allow List	List of criteria entries in the Allow List for the DPF Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the DPF Security Function Policy
No-match	The action that the DPF Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the DPF security threat database

Table 14 - DNS Protocol Filtering Security Function Policy Parameters

10.10 Protective DNS Security Function Policy

The Protective DNS (PDNS) Security Function Policy is an Atomic Policy that defines the rules that determine whether a subset of the Service Flow is Allowed, Blocked, or Quarantined by the PDNS Security Function, which is described in more detail in Section 9.6.

[R125] Each Protective DNS Security Function Policy **MUST** contain the parameters listed in Table 15.

Additional parameters can be agreed between the Subscriber and Service Provider.

PDNS Security Function Policy Parameter	**Description**
Policy ID	The identifier for a given PDNS Security Function Policy
Supported List - DNS encryption	List of encryption methods supported by the PDNS Security Function Policy. Possible values: unencrypted, DNS over HTTPS, DNS over TLS, DNSSEC, Other
Unsupported List - DNS encryption	List of encryption methods not supported by the PDNS Security Function Policy. Possible values: unencrypted, DNS over HTTPS, DNS over TLS, DNSSEC, Other
Supported List - DNS RR	List of criteria entries for each of the DNS resource records that are supported by the PDNS Security Function Policy
Unsupported List - DNS RR	List of criteria entries for each of the DNS resource records that are not supported by the PDNS Security Function Policy
Block List	List of criteria entries in the Block List for the PDNS Security Function Policy.
Allow List	List of criteria entries in the Allow List for the PDNS Security Function Policy
Quarantine List	List of criteria entries in the Quarantine List for the PDNS Security Function Policy
No-match	The action that the PDNS Security Function takes when there is no match. The possible values are Allow or Block.
Duration	Duration of time between updates of the PDNS security threat database

Table 15 - Protective DNS Security Function Policy Parameters

Editor Note 11: The information in the four rows (Supported and Unsupported Lists) in Table 15 were agreed to be added during the Q4 editing session in Dallas, October 2023. This needs a good review and comments. What we agree on here needs to be aligned with the eventual text referred to in Editor Note 9:. Not sure if we need only two rows of lists tied together, or four rows. Thoughtful comments are welcome.

11 References

[1] MEF 61.1, IP Service Attributes, January 2019

[2] MEF 70.1, SD-WAN Service Attributes and Services, October 2021

[3] MEF 88, Application Flow Security for SD-WAN Services, November 2021

[4] MEF 117, SASE Service Attributes and Service Framework, October 2022

[5] MEF 118, Zero Trust Framework for MEF Services, October 2022

[6] IETF RFC 1035, DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION, by P. Mockapetris, November 1987

[7] IETF RFC 1183, New DNS RR Definitions, by C. Everhart, L. Mamakos, R. Ullmann, and P. Mockapetris, October 1990.

[8] IETF RFC 1996, A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY), by Paul Vixie, August 1996

[9] IETF RFC 2119, Key words for use in RFCs to Indicate Requirement Levels, by Scott Bradner, March 1997

[10] IETF RFC 2782, A DNS RR for specifying the location of services (DNS SRV), by A. Gulbrandsen, P. Vixie, and L. Esibov, February 2000. Copyright © The Internet Society (2000). All Rights Reserved.

[11] IETF RFC 2818, HTTP Over TLS, by Eric Rescorla, May 2000. Copyright © The Internet Society (2000). All Rights Reserved.

[12] IETF RFC 2930, Secret Key Establishment for DNS (TKEY RR), by D. Eastlake, 3rd, September 2000. Copyright © The Internet Society (2000). All Rights Reserved.

[13] IETF RFC 3339, Date and Time on the Internet: Timestamps, by Chris Newman and Graham Klyne, July 2002. Copyright © The Internet Society (2002). All Rights Reserved.

[14] IETF RFC 3507, Internet Content Adaptation Protocol (ICAP), by Jeremy Elson and Alberto Cerpa, April 2003. Copyright © The Internet Society (2003). All Rights Reserved.

[15] IETF RFC 3596, DNS Extensions to Support IP Version 6, by S. Thomson, C. Huitema, V. Knisant, M. Souissi, October 2003. Copyright © The Internet Society (2003). All Rights Reserved.

[16] IETF RFC 3629, UTF-8, a transformation format of ISO 10646, by Francois Yergeau, November 2003. Copyright © The Internet Society (2003). All Rights Reserved.

[17] IETF RFC 3986, Uniform Resource Identifier (URI): Generic Syntax, by Tim Berners-Lee, Roy T. Fielding, and Larry Masinter, January 2005. Copyright © The Internet Society (2005).

[18] IETF RFC 4033, DNS Security Introduction and Requirements, by R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, March 2005. Copyright © The Internet Society (2005).

[19] IETF RFC 4034, Resource Records for the DNS Security Extensions, by R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, March 2005. Copyright © The Internet Society (2005).

[20] IETF RFC 4035, Protocol Modifications for the DNS Security Extensions, by R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, March 2005. Copyright © The Internet Society (2005).

[21] IETF RFC 4106, The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP), by J. Viega, D. McGrew, June 2005. Copyright © The Internet Society (2005).

[22] IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace, by Paul Leach, Michael Mealling, and Rich Salz, July 2005. Copyright © The Internet Society (2005).

[23] IETF RFC 4255, Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints, by J. Schlyter, W. Griffin, January 2006. Copyright © The Internet Society (2006).

[24] IETF RFC 4754, IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA), by D. Fu, J. Solinas, January 2007. Copyright © The IETF Trust (2007).

[25] IETF RFC 4868, Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec, by S. Kelly, S. Frankel, May 2007. Copyright © The IETF Trust (2007).

[26] IETF RFC 5074, DNSSEC Lookaside Validation (DLV), by S. Weiler, November 2007. Copyright © The IETF Trust (2007).

[27] IETF RFC 5155, DNS Security (DNSSEC) Hashed Authenticated Denial of Existence, by B. Laurie, G. Sisson, R. Arends, and D. Blacka, March 2008. Copyright © The IETF Trust (2008).

[28] IETF RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, by Tim Dierks and Eric Rescorla, August 2008. Copyright © The IETF Trust (2008).

[29] IETF RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, by David Cooper, et al., May 2008. Copyright © The IETF Trust (2008).

[30] IETF RFC 5282, Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol, by D. Black, D. McGrew, August 2008. Copyright © The IETF Trust (2008).

[31] IETF RFC 5903, Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2, by D. Fu, J. Solinas, June 2010. Copyright © 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

[32] IETF RFC 6066, Transport Layer Security (TLS) Extensions: Extension Definitions, by Donald Eastlake, January 2011. Copyright © 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

[33] IETF RFC 6071, IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap, by S. Frankel, S. Krishnan, February 2011. Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

[34] IETF RFC 6594, Use of the SHA-256 Algorithm with RSA, Digital Signature Algorithm (DSA), and Elliptic Curve DSA (ECDSA) in SSHFP Resource Records, by O. Sury, April 2012. Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

[35] IETF RFC 6698, The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA, by P. Hoffman, J. Schlyter, and Kirei AB, August 2012. Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

[36] IETF RFC 6763, DNS-Based Service Discovery, by S. Cheshire, M. Krochmal, February 2013. Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

[37] IETF RFC 6797, HTTP Strict Transport Security (HSTS), by Jeff Hodges, Collin Jackson, and Adam Barth, November 2012. Copyright © 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

[38] IETF RFC 6895, Domain Name System (DNS) IANA Considerations, by Donald Eastlake, April 2013. Copyright © 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

[39] IETF RFC 6962, Certificate Transparency, by Ben Laurie, Adam Langley, and Emilia Kasper, June 2013. Copyright © 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

[40] IETF RFC 7230, Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing, by Roy Fielding and Julian Reschke, June 2014. Copyright © 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

[41] IETF RFC 7296, Internet Key Exchange Protocol Version 2 (IKEv2), by C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, and T. Kivinen, October 2014. Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

[42] IETF RFC 7553, The Uniform Resource Identifier (URI) DNS Resource Record, by P. Faltstrom, O. Kolkman, June 2015. Copyright © 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

[43] IETF RFC 7686, The ".onion" Special-Use Domain Name, by Jacob Appelbaum and Alec Muffett, October 2015. Copyright © 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

[44] IETF RFC 7858, Specification for DNS over Transport Layer Security (TLS), by Zi Hu et al., May 2016. Copyright © 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

[45] IETF RFC 7929, DNS-Based Authentication of Named Entities (DANE) Bindings for OpenPGP, by P. Wouters, August 2016. Copyright © 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

[46] IETF RFC 8162, Using Secure DNS to Associate Certificates with Domain Names for S/MIME, by P. Hoffman, J. Schlyter, Kirei AB, May 2017. Copyright © 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

[47] IETF RFC 8174, Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words, by Barry Leiba, May 2017. Copyright © 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

[48] IETF RFC 8446, The Transport Layer Security (TLS) Protocol Version 1.3, by Eric Rescorla, August 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

[49] IETF RFC 8484, DNS Queries over HTTPS (DoH), by Paul Hoffman, October 2018. Copyright © 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

[50] IETF RFC 8945, Secret Key Transaction Authentication for DNS (TSIG), by F. Dupont, S. Morris, P. Vixie, D. Eastlake 3rd, O. Gudmundsson, B. Wellington, November 2020. Copyright © 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

[51] IETF RFC 8976, Message Digest for DNS Zones, by D. Wessels, P. Barber, M. Weinberg, W. Kumari, and W. Hardaker, February 2021. Copyright © 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

[52] IETF RFC 9000, QUIC: A UDP-Based Multiplexed and Secure Transport, by Jana Iyengar and Martin Thomson, May 2021. Copyright © 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

[53] IANA TLS Cipher Suites Registry, https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml

[54] IANA, Number Resources, https://www.iana.org/numbers

[55] IANA, Protocol Numbers, https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

[56] IANA, Service Name and Transport Protocol Port Number Registry, https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[57] FIPS PUB 140-2, SECURITY REQUIREMENTS FOR CRYPTOGRAPHIC MODULES, May 2001, https://csrc.nist.gov/publications/detail/fips/140/2/final

[58] NIST SP 800-52, Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations, Rev. 2, August 2019

[59] NIST SP 800-53 (Rev. 5), Security and Privacy Controls for Federal Information Systems and Organizations, Information System Monitoring, September 2020. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf

[60] NIST SP 800-56B (Rev. 2), Recommendation for pair-wise key establishment using integer factorization cryptography, March 2019, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf

[61] NIST SP 800-57 (Part 1, Rev. 5), Recommendation for Key Management: Part 1 – General, May 2020, https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-5/final

[62] NIST SP 800-77 Rev. 1, Guide to IPsec VPNs, June 2020.

[63] NIST SP 800-122, GUIDE TO PROTECTING THE CONFIDENTIALITY OF PERSONALLY IDENTIFIABLE INFORMATION (PII), April 2010, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf

[64] CA/Browser Forum, Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates, version 1.7.1, August 2020. https://cabforum.org/baseline-requirements/

[65] MITRE, https://cve.mitre.org

[66] MITRE, ATT&CK, https://attack.mitre.org

[67] NSA and CISA, Selecting a Protective DNS Service, U/OO/117652-21 | PP-21-0251 | Ver. 1.2, May 2021.

Appendix A Service Flows and Security Functions (Informative)

This Appendix describes several use cases of Service Flows, with one or more Security Functions enabled. The use cases involve unencrypted Service Flows, encrypted Service Flows, and a Service Flow with a mix of traffic.

It describes the Security Function behavior expected at the IP Service Edge by describing a hypothetical architecture. It does not imply that any implementation be architected or implemented based on this model.

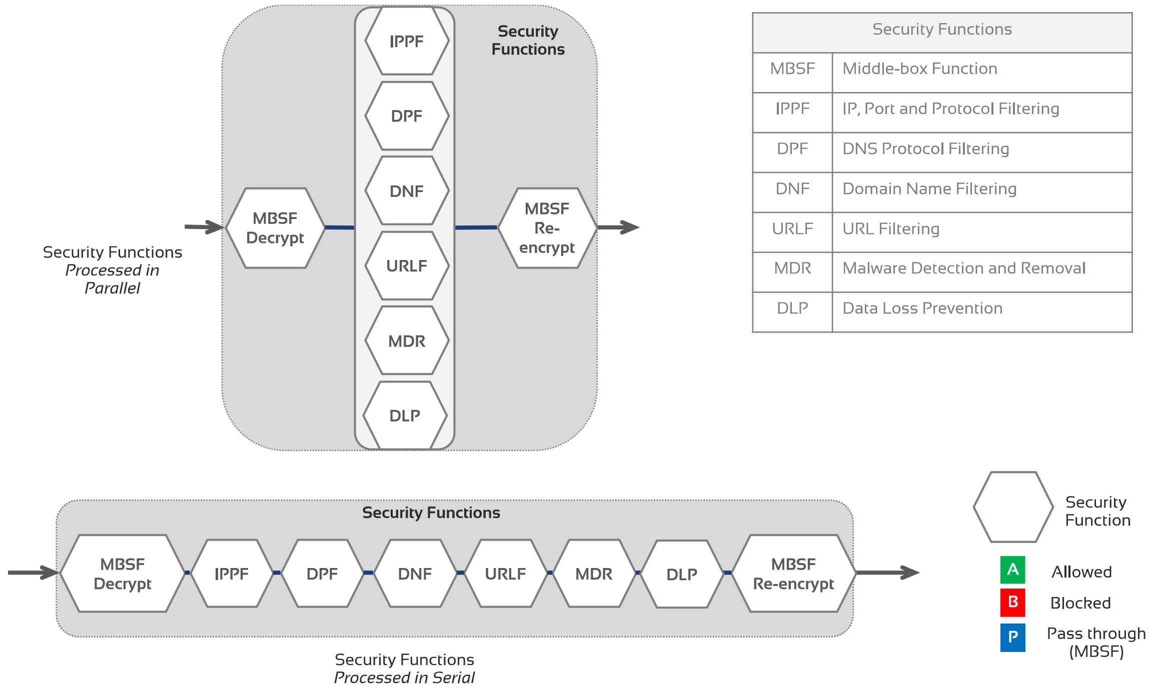
Editor Note 12: A contribution is solicited to add a use case for DNS, including Protective DNS. Note that the preferred DNS Server is not part of PDNS – IPPF could be used to control this. The contribution needs to also deal with encrypted / unencrypted flows.

The following acronyms are used throughout this Appendix and are briefly described here.

- MBSF – Middlebox Security Function
- IPPF – IP, Port and Protocol Filtering
- DPF – DNS Protocol Filtering
- DNF – Domain Name Filtering
- URLF – URL Filtering
- MD+R – Malware Detection and Removal
- DLP – Data Loss Prevention
- A – Allowed
- B – Blocked
- P – Pass through (applies to MBSF)

Editor Note 13: 1) we need to add Protective DNS to the above bullet list, and 2) we need to update Figure 9, tying the DNS security functions together. Editor has a new figure, but it first needs review before we agree to put it in.

The Security Functions are depicted with hexagons and labelled appropriately. These functions can be thought to be processed in parallel or serially, as shown in Figure 9 .



Security Functions	
MBSF	Middle-box Function
IPPF	IP, Port and Protocol Filtering
DPF	DNS Protocol Filtering
DNF	Domain Name Filtering
URLF	URL Filtering
MDR	Malware Detection and Removal
DLP	Data Loss Prevention

Figure 9 - Legend of Security Functions

This document does not constrain how these Security Functions are processed. This appendix describes the Security Function behavior expected in the Service by describing a hypothetical architecture. It does not imply that any implementation be architected or implemented based on this model. A brief explanation of both is given below.

Parallel processing of Security Functions can provide the following benefits: all Security Functions processed in one location; single packet lookup across multiple functions; and reduced latency. It requires a single Service Provider providing all the functions. A challenge of this approach is a possibly higher compute power needed at the location.

Serial processing of Security Functions can provide the following benefits: lower compute power at a given location; allows for multiple security provider solution: and might provide for more complex security models. Serial processing requires multiple security applications. A challenge of this approach is the integration of the multiple security providers. Note that the ordering of security functions is implementation dependent.

This document recognizes that both serial and parallel processing are valid implementations. Each has its benefits and challenges. This document does not take a position as to which should be utilized when applying Security Functions to a Service.

For the sake of this Appendix, serial processing has been utilized in all the use cases, as it is easier to graphically represent the content of the use cases. This should not be construed as an endorsement of one option over another and is provided as exemplary only.

When the term Service Flow is used in the following use cases, it could apply to either an Ingress Service Flow or an Egress Service Flow.

Editor Note 14: Neil has agreed to provide a contribution featuring a use case(s) for DNS service flows, and possibly some changes to other use cases. Note that the preferred DNS Server is not included as part of Protective DNS or DNS Protocol Filtering, so it is expected that IP, Port and Protocol Filtering would be used to constrain service flows with DNS messages to a list of preferred DNS servers.

A.1 Use Case 1: Encrypted Service Flow is Allowed

The Subscriber's Service Flow is using TLS 1.2. The Service Policy includes the following Security Functions for Service Flow A: MBSF, IPPF, DPF, DNF, URLF, and MDR.

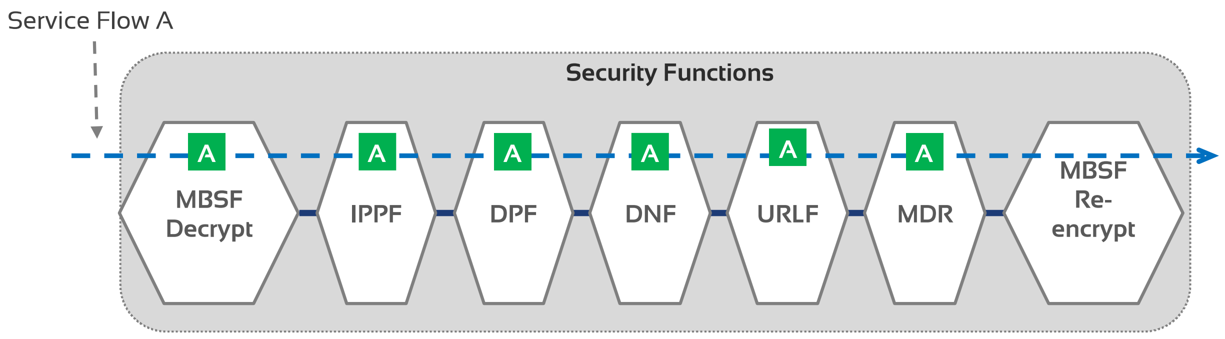


Figure 10 - Use Case 1: Encrypted Service Flow is Allowed

In Use Case 1, MBSF decrypts and re-encrypts Service Flow A, and the individual Security Functions are applied. All checks are good, and Service Flow A is Allowed.

A.2 Use Case 2: Encrypted Service Flow, Malware is Detected

The Subscriber's Service Flow is using TLS 1.2. The Service Policy includes the following Security Functions for Service Flow B: MBSF, IPPF, DPF, DNF, URLF, and MDR. E-mail Messages are subject to Malware Detection. Messages with Malware are Blocked; and messages without Malware are Allowed.

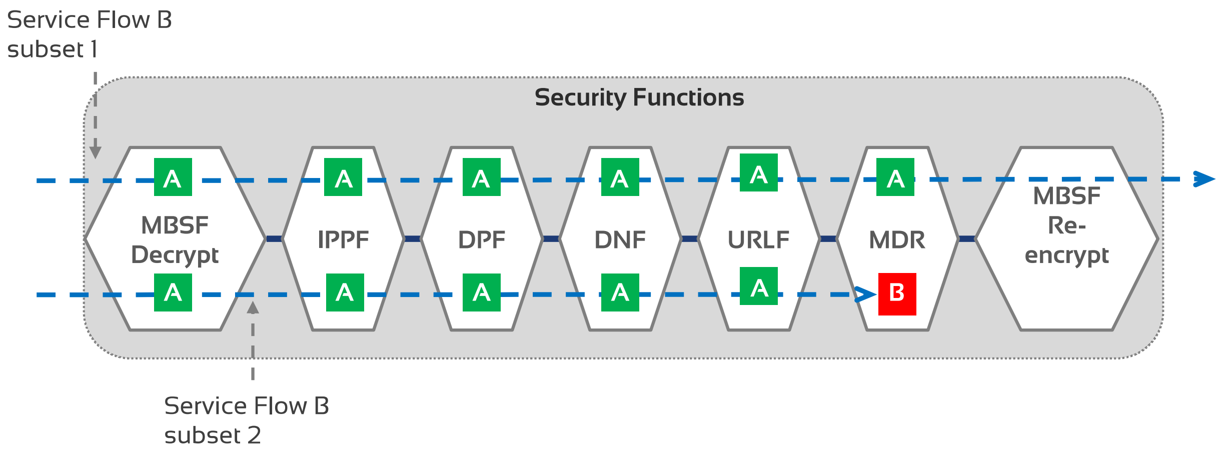


Figure 11 - Use Case 2: Encrypted Service Flow, Malware is Detected

MBSF decrypts and re-encrypts Service Flow B and the individual Security Functions are applied. Subset 1 of Service Flow B is clean (no Malware), and it is Allowed. Malware is detected in subset 2 of Service Flow B and subset 2 is Blocked.

A.3 Use Case 3: Encrypted Cloud Storage application using QUIC

The Subscriber is running a cloud storage application using QUIC (IETF RFC 9000 [52]). The Service Policy includes the following Security Functions for Service Flow Q: MBSF, IPPF, and DNS Protocol Filtering (DPF). QUIC is included in both the MBSF Unsupported List and the MBSF Allow List and so the MBSF passes QUIC through unchanged (in Figure 12, see the blue box labelled P).

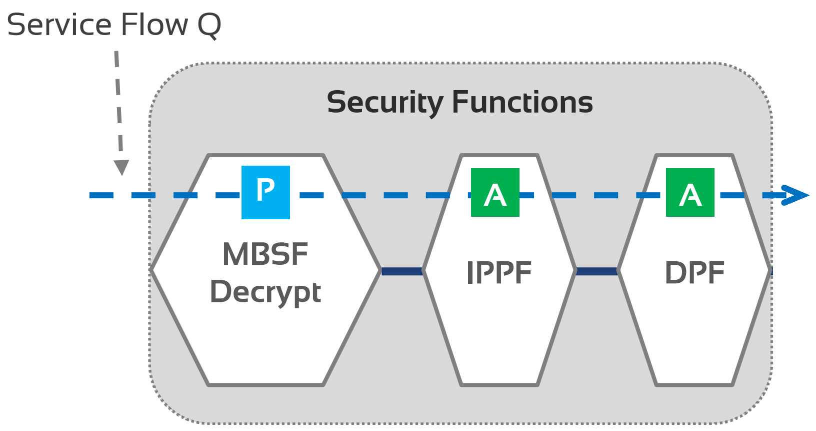


Figure 12 - Use Case 3: Encrypted Cloud Storage Application using QUIC

Service Flow Q is defined as QUIC, using UDP port 443 and an allowed list of IP addresses of QUIC servers. MBSF, IPPF, and DPF checks are good. Service Flow Q is Allowed.

A.4 Use Case 4: Encrypted Web Traffic to Public Resource, DNF Blocks subset 2

The application consists of web traffic to a public resource. The Service Policy includes the following Security Functions for Service Flow G: MBSF, IPPF, DPF, and DNF. In this use case, HTTP clients are trying to access two different domains. Subset 1 of Service Flow G is trying to connect to site.qaz.com web server, while Subset 2 of Service Flow G is trying to connect to badresource.badcompany.org web server.

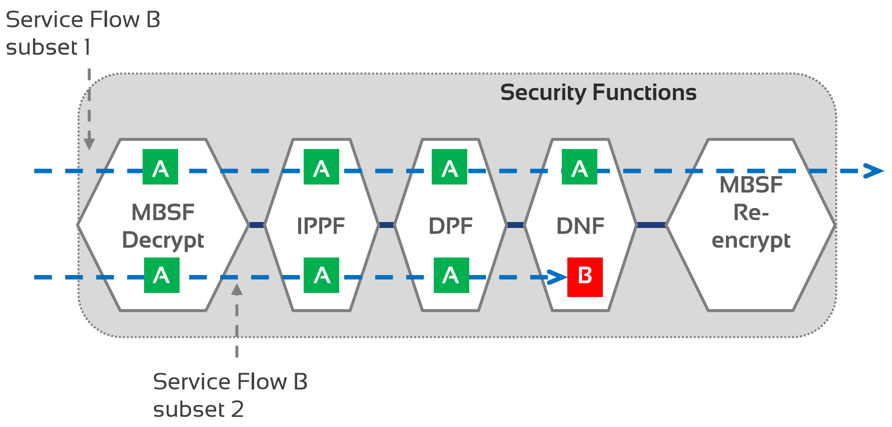


Figure 13 - Use Case 4: Encrypted Web Traffic to a Public Resource, DNF Blocks subset 2

Subset 1 is Allowed since all Security Function checks are good. For subset 2, the IPPF and DPF checks are good, but since the badresource.badcompany.org domain name is on the Domain Name Filtering Block List, subset 2 is Blocked.

A.5 Use Case 5: File Transfer Application with FTP and SSH

The application is file transfer. The Service Policy includes the following Security Functions for Service Flow F: MBSF, IPPF, and DPF. File uploads consist of Secure Shell (SSH) and File Transfer Protocol (FTP). By policy, FTP is not Allowed (FTP is on the IPPF Block List) while SSH is Allowed (SSH is on the IPPF Allow List).

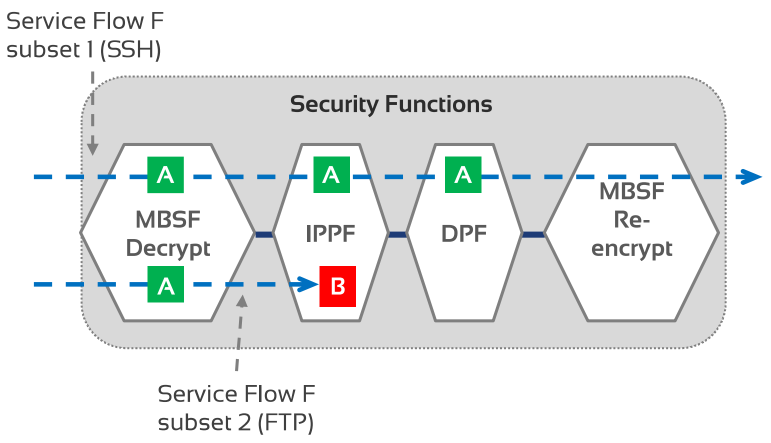


Figure 14 - Use Case 5: File Transfer Application Flow with FTP and SSH

Subset 1 of Service Flow F is Allowed since SSH is on the IPPF Allow List. In the case of subset 2, DPF checks are good, but IPPF checks identify a file upload using FTP, which is on the IPPF Block List. Therefore, subset 2 of Service Flow F is Blocked.

A.6 Use Case 6: Encrypted e-mail application, PII is Detected

The Subscriber's encrypted e-mail application is using TLS 1.2. The Service Policy includes the following Security Functions for Service Flow B: MBSF, IPPF, DPF, DNF, URLF, and DLP. E-mail Messages are subject to scanning for Personally Identifiable Information (PII). Messages with PII are Blocked; and messages without PII are Allowed.

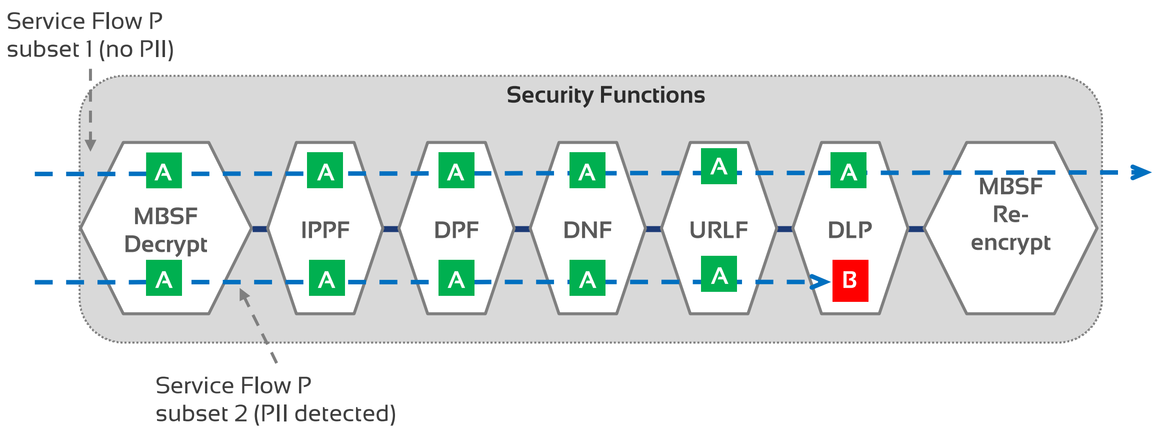


Figure 15 - Use Case 6: Encrypted e-mail application, PII is Detected

MBSF decrypts and re-encrypts Service Flow P and the individual Security Functions are applied. Subset 1 of Service Flow P is clean (no PII), and it is Allowed. Subset 2 of Service Flow P matches a criteria entry on the DLP Block List and, therefore, the Data Loss Prevention Security Function Blocks subset 2 of Service Flow P.

A.7 Use Case 7: Unencrypted Web Traffic to a Weather Site

The application is unencrypted web traffic to a weather site. The Service Policy includes the following Security Functions for Service Flow W: MBSF, IPPF, and DPF.

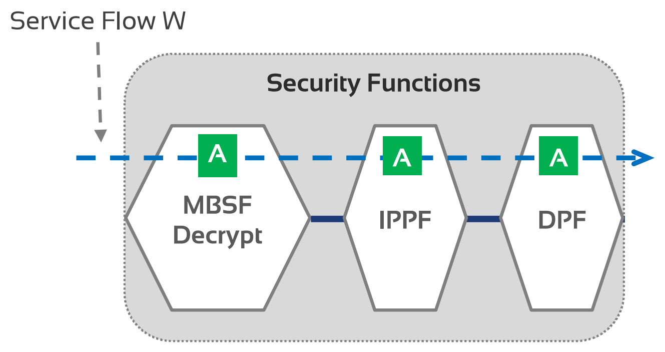


Figure 16 - Use Case 7: Unencrypted Web Traffic to a Weather Site

Service Flow W does not match any entries on any of the MBSF, IPPF or DPF Lists. By policy, the default action for each of these Security Functions is to Allow. Service Flow W is Allowed.

A.8 Use Case 8: Unencrypted Web Traffic, DPF Blocks subset 2

The application is unencrypted web traffic to a public resource. The Service Policy includes the following Security Functions for Service Flow D: MBSF, IPPF, and DPF. Service Flow D consists of DNS messages to two different DNS servers. Subset 1 of Service Flow D is sending DNS messages to DNS Server 1, which is on the Allow List while subset 2 of Service Flow D is sending DNS messages to DNS Server 2, which is on the Block List.

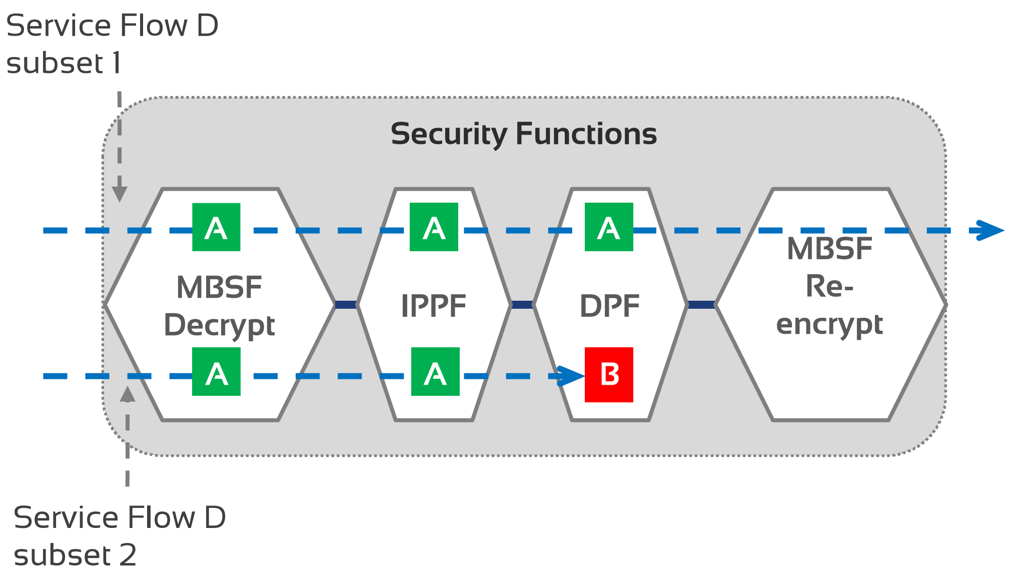


Figure 17 - Use Case 8: Unencrypted Web Traffic, DPF Blocks subset 2

The DNS messages to DNS Server 1 matches a criteria entry on the DPF Allow List, and therefore, the DPF Security Function Allows subset 1. As for subset 2, the DPF Security Function detects DNS messages to DNS Server 2 that match a criteria entry on the DPF Block List and Blocks those messages.

A.9 Use Case 9: Encrypted and Unencrypted Traffic to a Commerce Website

The application is web traffic to a commerce web site. The Subscriber wants only encrypted traffic to get through. The Service Policy includes the following Security Functions for Service Flow C: MBSF, IPPF, DPF, DNF, URLF, MDR, and DLP. Subset 1 of Service Flow C is using TLS 1.2 with an appropriate cipher suite, which matches a criteria entry on the MBSF Allow List. Subset 2 of Service Flow C is sending unencrypted traffic, which is not allowed.

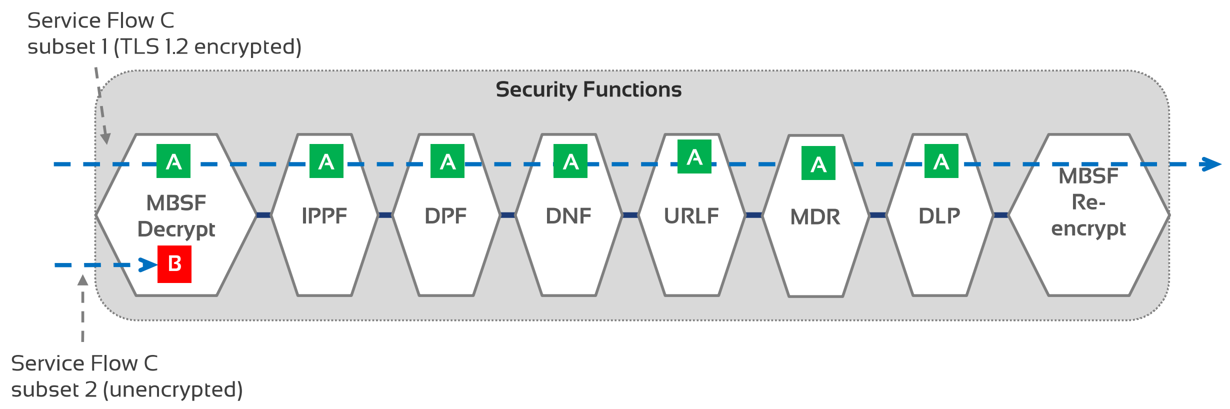


Figure 18 - Use Case 9: Encrypted and Unencrypted Traffic to a Commerce Website

In this case, all Security Functions are enabled, and all Security Function checks are good. Subset 1 is Allowed. MBSF is configured to Block unencrypted traffic, i.e., the default behavior for MBSF is to Block any subset not matching a criteria entry on the MBSF Allow List, so MBSF Blocks subset 2.

A.10 Use Case 10: Unencrypted Traffic to an Internet Website

The application is unencrypted web traffic to a commerce web site. The Service Policy includes the following Security Functions for Service Flow X: MBSF, IPPF, DPF, DNF, URLF, MDR, and DLP. By policy, unencrypted traffic is Allowed on this Service Flow.

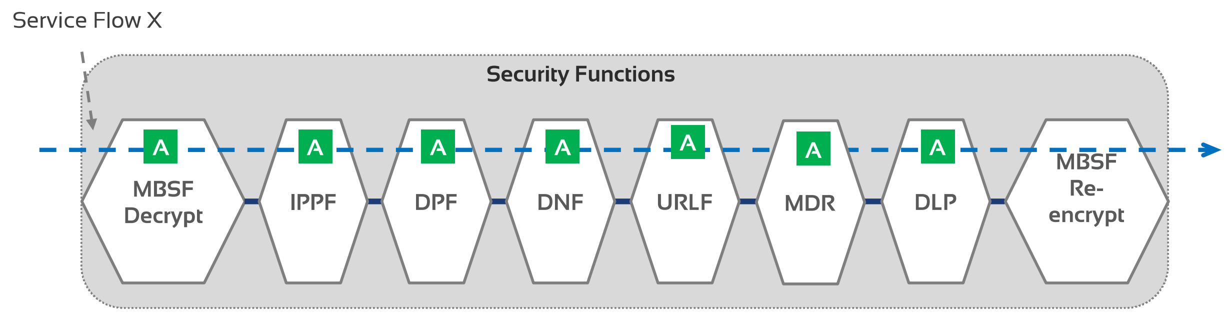


Figure 19 - Use Case 10: Internet Website, unencrypted traffic

A Middlebox Security Function is configured to Allow the unencrypted traffic, and all other Security Function checks are good. The Service Flow is Allowed.

A.11 Use Case 11: Social Media Website, URL Filtering Blocks subset 2

The application is access to a social media web site, allowing a mix of unencrypted and encrypted traffic. The Subscriber wants to Block access to adult content. The Service Policy includes the following Security Functions for Service Flow Z: MBSF, IPPF, DPF, DNF, URLF, and MDR. Subset 1 of Service Flow Z consists of requests to site.qaz.com/user-site, while subset 2 of Service Flow Z consists of requests to site.qaz.com/advertisement/adultcontent. By policy, URLs containing adult content need to be Blocked.

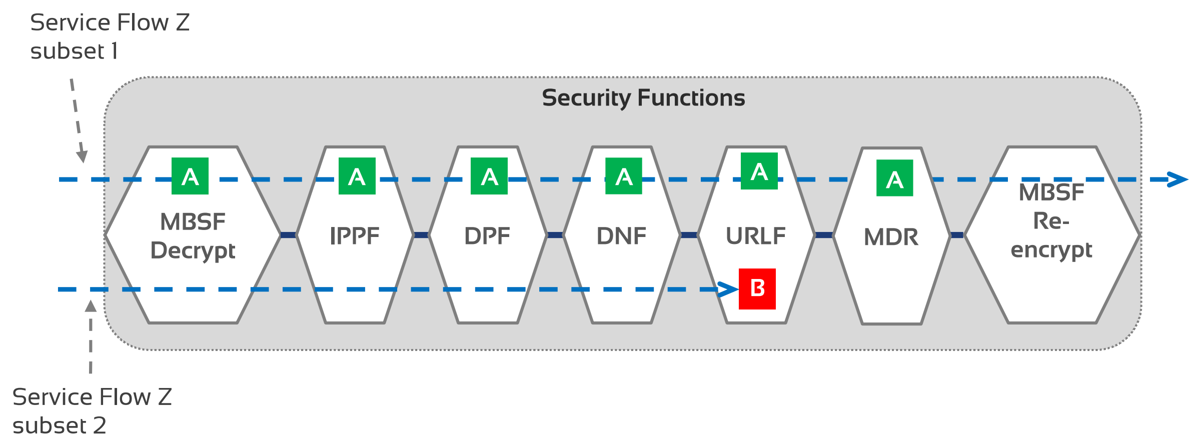


Figure 20 - Use Case 11: Internet Website, unencrypted traffic

For subset 1 of Service Flow Z, all Security Function checks are good, and subset 1 is Allowed. For subset 2 of Service Flow Z, the URLF Security Function detects a URL that is on the URL Filtering Block List, and therefore Blocks subset 2.

Appendix B Examples of Security Event Notifications (Informative)

The SEN example depicted in Table 16 is related to a Malware Detection and Removal Security Function that detected a malware signature, per a CVE item.

Item	Example Value	Comments
Issuer	Acme Internet Services	
Timestamp of SEN	2022-12-03T14:35:07.978	
SEN ID	14990089-f356-4f01-8090-8236e546efb8	
Security Function	Malware Detection and Removal	
Security Function Policy ID	*Malware-2*	
Type of SEN	IOC	
Type of Security Event	CVE-2021-22893: https[://]nvd[.]nist[.]gov/vuln/detail/CVE-2021-22893	CVE-2021-22893 – Pulse Secure Pulse Connect Secure[5]: Remote Arbitrary code execution exploit
Security Event Source IP address	192.168.50.2	
Security Event UNI ID	HQ VPN Pool 3	Recommended
Security Event details	MD+R Violation: Rule 42 - Remote Command Execution Violation: Details: User Neil (192.168.50.2) attempted to send arbitrary code for execution.	
Action Taken	Blocked transmission attempt.	

Table 16 - Example of Type of SEN = IOC

The SEN example depicted in Table 17 is related to a Data Loss Prevention Security Function that detected an attempt to upload credit card numbers to a cloud account, per an ATT&CK item.

[5] *Pulse Connect Secure* is a product offered by the company *Pulse Secure*.

Item	Example Value	Comments
Issuer	Acme Internet Services	
Timestamp of SEN	2022-12-01T04:05:02.345	
SEN ID	169e1af9-1557-4019-b175-07fbe89089b1	
Security Function	Data Loss Prevention	
Security Function Policy ID	*DLP-1*	
Type of SEN	ATT&CK	
Type of Security Event	TA0010	TA0010 - Exfiltration
Security Event Source IP address	10.10.42.42	
Security Event UNI ID	San Jose HQ VLAN 101	Recommended
Security Event details	T1537: DLP Violation: PCI-DSS Rule 500: Details: User Neil (10.10.42.42) attempted to upload file creditcardnos.xls to dropbox[.]com.	T1537 - Transfer data to cloud account – DLP Security Function would identify Credit Card Numbers per PCI-DSS Rule 500. Also, showing normalization of URL.
Action Taken	Blocked file upload	

Table 17 - Example of Type of SEN = ATT&CK

Appendix C Examples of Malware Detection (Informative)

The following examples describe different possible outcomes when Malware Detection and Removal is *Enabled* for a given Service Flow. In these examples, it is assumed that the Service Flow is using TLS 1.2 and that a Middlebox Security Function has been *Enabled* to see unencrypted traffic. Other Security Functions would have already been applied to the Service Flow before checking for Malware.

Figure 21 shows an example of a clean file going through the Malware Detection and Removal process.

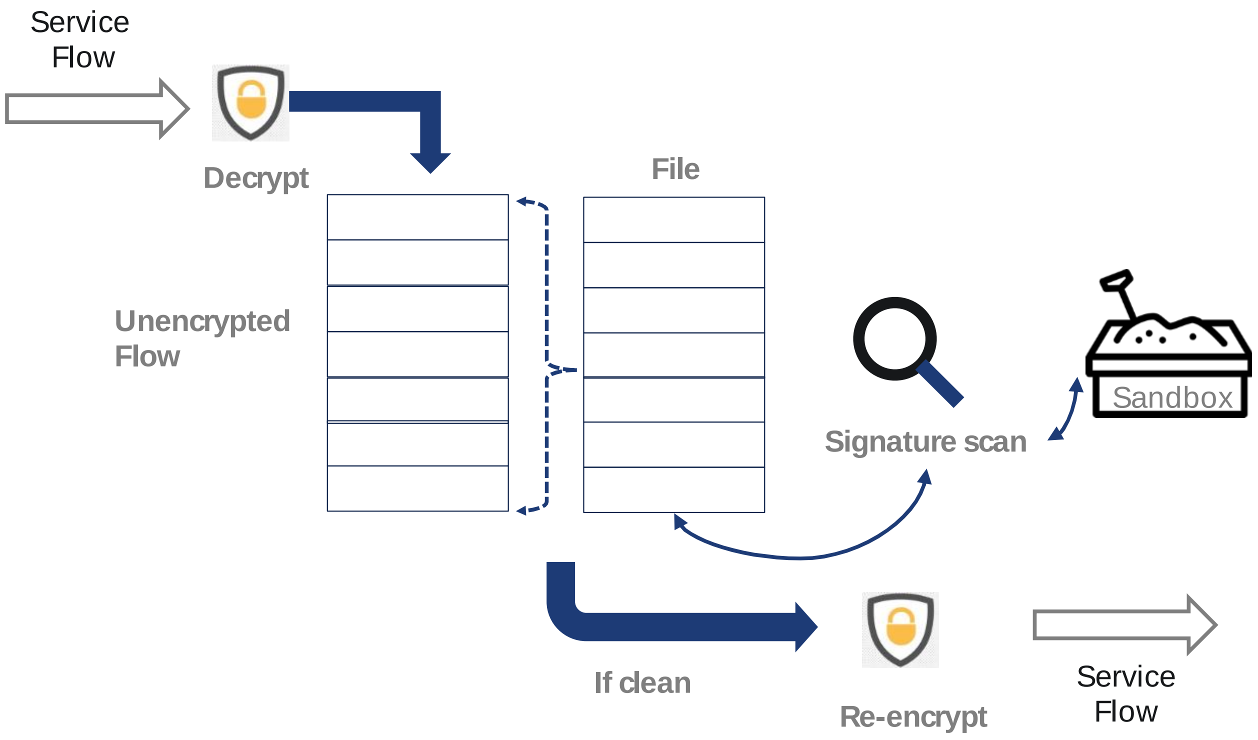


Figure 21 - Example of a Clean File

In the example shown in Figure 21, the file is identified and scanned to detect a possible Malware. The scan result is normal, and there is no need for further checking in the sandbox and so the Service Flow is re-encrypted and forwarded.

Figure 22 shows an example of a file containing Malware that is detected with a signature scan. In this figure, the red rectangle represents a part of a file that contains Malware.

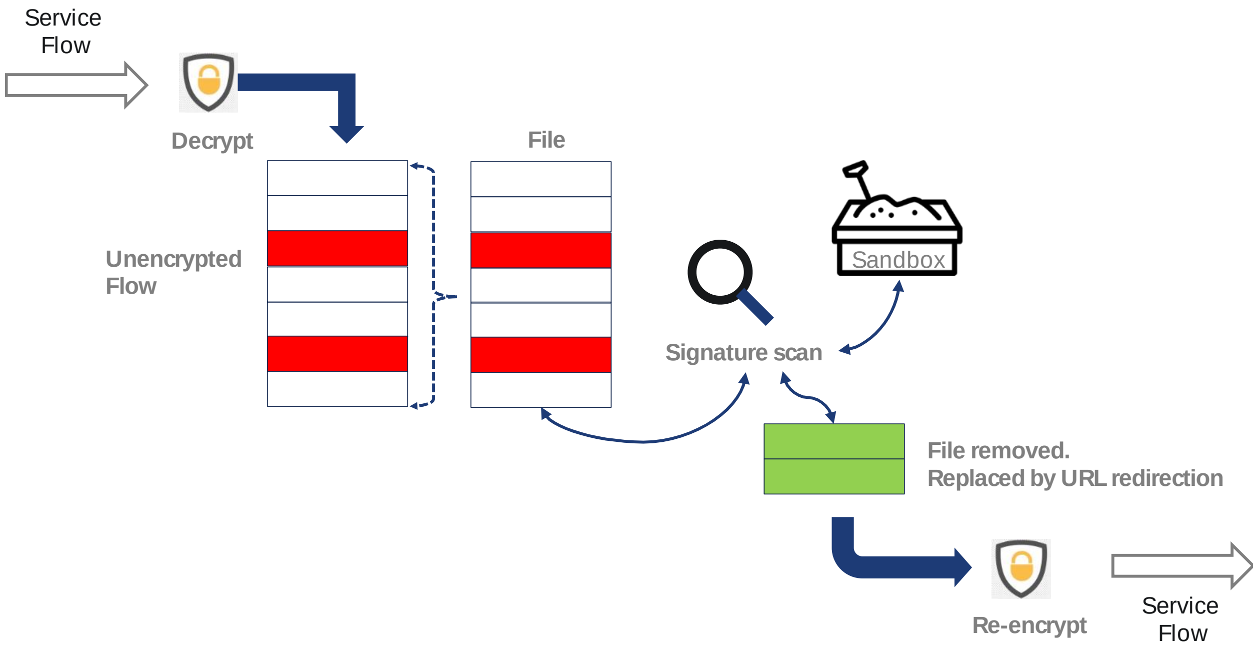


Figure 22 - Example of Malware Detected and File Removed using a Signature Scan

In the example shown in Figure 22, the file is identified and scanned to detect a possible Malware and Malware is detected. A decision is made to remove the file and replace it with a URL re-direction that explains that a file containing Malware has been detected and removed.

Figure 23 shows an example of a file containing Malware that is detected in the sandbox. As previously noted, the red rectangle represents a part of a file that contains Malware. In this example, the Internet Content Adaptation Protocol (ICAP), as specified in RFC 3507 [14], is used for transferring data into and back from the sandbox.

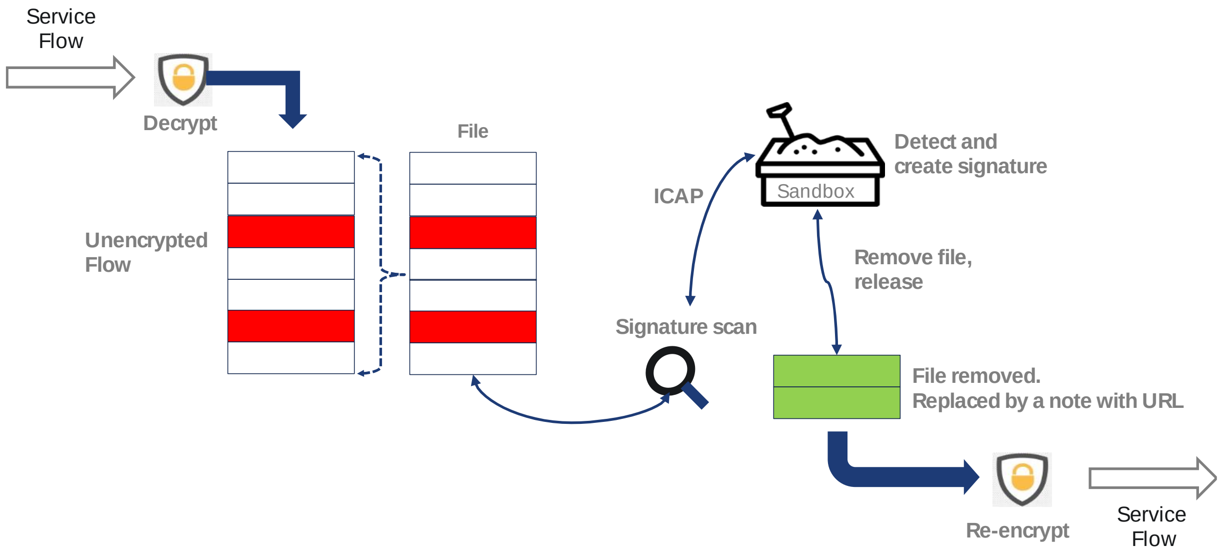


Figure 23 - Example of Malware Detected and File Removed using a Sandbox

In the example shown in Figure 23, the file is identified, and the signature scan is normal, but the file looks suspicious, and a decision is made to sandbox the file for further analysis. Malware is detected in the sandbox and a new signature is created for the unknown Malware. The file is removed and replaced with a URL re-direction that explains that a file containing Malware has been detected and removed.

Figure 24 shows an example of a file containing Malware that is detected by a signature scan. As previously noted, the red rectangle represents a part of a file that contains Malware.

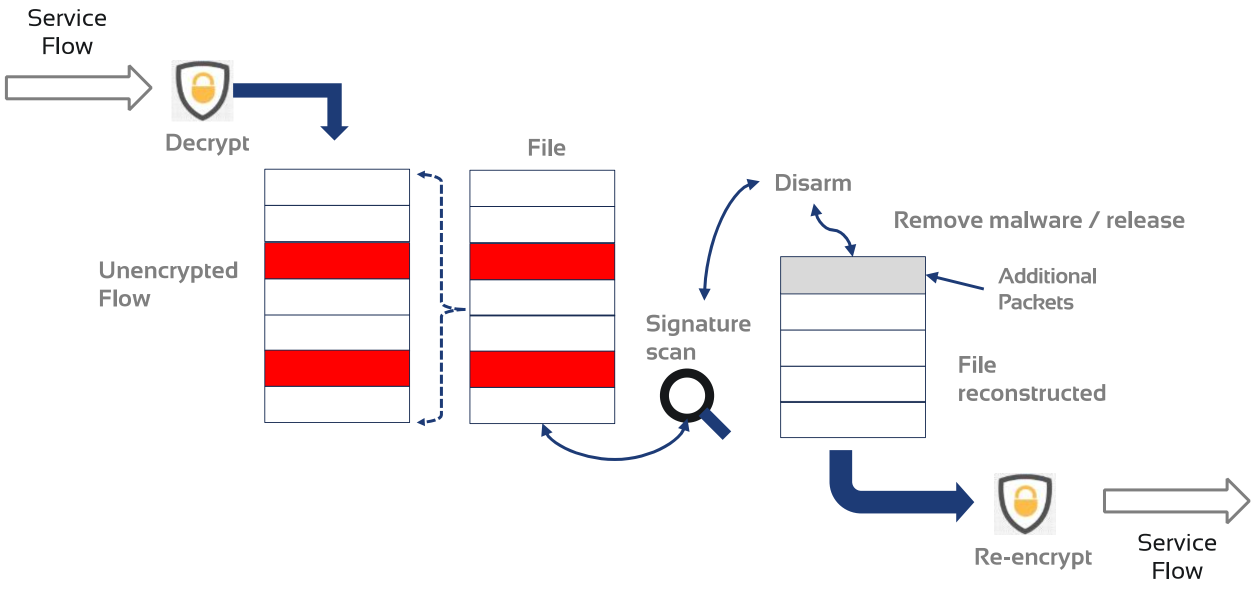


Figure 24 - Example of Malware Detected and File Reconstructed

In example shown in Figure 24, the file is identified and scanned to detect a possible Malware and Malware is detected. A decision is made to remove the part of the file that has the Malware and reconstruct the clean part of the file for forwarding. A message could be added explaining that a file containing Malware has been detected and the Malware removed from the file.

Appendix D Threat Modeling (Informative)

Editor Note 15: This appendix will be updated after CfC 3. Comments are welcome.

Threat modelling is the process by which threats, whether vulnerabilities or the absence of appropriate controls, can be described and mitigations or remediations planned. The purpose of the process is to provide those responsible for designing, developing, implementing, operating, or using the system with a systematic understanding of what controls have been included and any gaps that may exist, given the nature of the system. In practice, this should consider the likely attacker's profile, the tools, tactics, and procedures they use to compromise such system and the assets which are most at risk. For this document, threat modelling was used to answer questions such as "Where would an SD-WAN solution be most vulnerable to attack?" and "What security controls do we need to design into the standard to protect against these attacks?".

For the purposes of this document, the working group took the following approach:

- Created an application diagram using Microsoft's Threat Modelling Tool
- Annotated the diagram with the properties as understood for each asset and functional flow
- Reviewed the working drafts to identify coverage of the threats identified by the previous two steps
- Annotated the threat with related references from this document
- Discussed and made changes to the overall document based on any gaps identified

The threat model identifies threats and groups them into six categories, based on the STRIDE model.

- Spoofing is defined as pretending to be someone or something other than yourself. Spoofing violates the property of authentication.
- Tampering is defined as modifying something on disk, network, memory or somewhere else. Tampering violates the property of integrity.
- Repudiation is defined as claiming you didn't do something or were not responsible. Repudiation violates the property of non-repudiation.
- Information disclosure is defined as providing information to someone not authorized to access it. Information disclosure violates the property of confidentiality.
- Denial of Service is defined as exhausting resources needed to provide service. Denial of Service violates the property of availability.
- Elevation of privilege is defined as allowing someone to do something they are not authorized to do. Elevation of privilege violates the property of authorization.

For threats that the document proposes to mitigate or remediate, the threat model provided attempts to reference the appropriate sections of this document in justifying their state as

"Mitigation Implemented". While a number of items within the threat model were also deemed to be "Out of scope", their inclusion in the references in the supplied threat model should be instructive as to some of the responsibilities of individual implementations which will need consideration by software engineering, implementation and/or operational teams.

In a network that leverages a Middlebox Security Function to enable security policy enforcement, the likelihood of the above categories of threat above can be critically affected by how the Middlebox Security Function intercepts and mediates access. The impact of design decisions in this regard can impact the security and privacy of both end-users as well as operators of the service, Middlebox Security Function, and indeed remote application endpoints. Since much of today's network traffic is encrypted (typically with TLS), it is critical that any interception and mediation performed by the Middlebox Security Function does not impact the efficacy of the integrity and confidentiality protections that encryption provides. Ensure that all properties of encrypted traffic (whether they relate to TLS versions, cipher suite selections, PKI certificate management or other) are maintained and that they are implemented and configured in-line with good security practices. Mitigation: [R47], [R48], [R50], [R51], [R52], [R53], [R55], [R56], [R57], [R58], [R59], [D6], [D7], [R60], [R61], [R63], and [R64] deals with transport security as it relates to the Middlebox Security Function.

An indicative selection of threats from the threat model are listed here, where mitigations are covered by requirements in this document. These are for illustrative purposes.

Threat ID 20

- Category: Elevation of Privilege.
- Description: Common SSO implementations such as OAUTH2 and OAUTH Wrap can be vulnerable to on-path attacks if cryptographic controls are weakened. Privilege manipulation attacks apply to supporting functions by which identity is asserted as much as to the original Service Flow itself, particularly if the application places additional trust on the flow because of the presence of the MIDDLEBOX SECURITY FUNCTION.
- Mitigation: [R52] deals with the need to secure network flows when communicating with supporting functions.

Threat ID 22

- Category: Information disclosure.
- Description: Improper data protection of Policy Enforcement can allow an attacker to read information not intended for disclosure, for example authentication and authorization flows. Information disclosure threats apply to supporting functions as much as the original Service Flow itself, particularly if the application places additional trust on the flow because of the presence of the MIDDLEBOX SECURITY FUNCTION. Review authorization settings.
- Mitigation: [R52] deals with the need to secure network flows when communicating with supporting functions.

Threat ID 23

- Category: Spoofing.
- Description: Policy Enforcement may be spoofed by an attacker, and this may lead to incorrect data delivered to the Authorization Provider. Spoofing attacks apply to supporting functions as much as the original Service Flow itself. Consider using a standard authentication mechanism to identify the source data store.
- Mitigation: [R52] deals with the need to secure network flows when communicating with supporting functions.

Threat ID 47

- Category: Repudiation.
- Description: Does the log capture enough data to understand what happened in the past? Do your logs capture enough data to understand an incident after the fact? Is such capture lightweight enough to be left on all the time? Do you have enough data to deal with repudiation claims? Make sure the log has sufficient and appropriate data to handle a repudiation claim. You might want to talk to an audit expert as well as a privacy expert about your choice of data.
- Mitigation: Requirements around logging and auditing are covered by [R23], [R25] and [R26].

Threat ID 51

- Category: Tampering.
- Description: Log readers can come under attack via log files. Remember that any user supplied data could be malicious and consider ways to canonicalize data in all logs. Implement a single reader for the logs, if possible, to reduce attack surface area. Be sure to understand and document log file elements which come from untrusted sources.
- Mitigation: Requirements around the trustworthiness of user originated data extracted from Service Flows are covered by [D3].

The threat model is available at this link:

https://github.com/MEF-GIT/MEF-SDWAN-Application-Flow-Security-Threat-Model/tree/24990298f45f28c7c0f0dca485566aaba28580a6

The commit ID for this version of the document referenced in the above link: 24990298f45f28c7c0f0dca485566aaba28580a6.

Appendix E Examples of Security Function Policies (Informative)

This Appendix describes examples of Security Function Policies.

Table 18 provides an example of a Middlebox Security Function Policy.

Security Function Policy	**Security Function Policy Parameter**	**Parameter Value**
MBSF-1	MBSF Policy Identifier	*MBSF-1*
	S (MBSF Supported List)	*[<1.2, {0xC0,0x2C, 0xC0,0x30, 0x00,0xA8}[6]>}*
	U (MBSF Unsupported List)	*[<1.3, {Any}>]*
	B (MBSF Block List)	*[<1.0, {Any}>, <1.1, {Any}>]*
	A (MBSF Allow List)	*[<1.2, {0xC0,0x2C, 0xC0,0x30, 0x00,0xA}>, <1.3, {Any}>]*
	Nm (No-match)	*Block*
	CA (list of trusted CAs)	*[TRUST-A, TRUST-B]*
	I (Invalid target certificate)	*Block*
	UP (Unencrypted packets)	*Block*

Table 18 - Example of a Middlebox Security Function Policy

Table 19 provides an example of an IP Port and Protocol Filtering Security Function Policy.

Security Function Policy	**Security Function Policy Parameter**	**Parameter Value**
IPPF-3	IPPF Policy Identifier	*IPPF-3*
	B (IPPF Block List)	*[<Any, Any, UDP, Any, Any, UDP, Any, 443>]*
	A (IPPF Allow List)	*[<Any, Any, TCP, Any, Any, TCP, Any, 443>]*
	Q (IPPF Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	D (security threat database update duration)	*4 hours*

Table 19 - Example of an IP Port and Protocol Filtering Security Function Policy

Table 20 provides an example of a DNS Protocol Filtering Security Function Policy.

Security Function Policy	**Security Function Policy Parameter**	**Parameter Value**
DPF-4	DPF Policy Identifier	*DPF-4*
	B (DPF Block List)	*[< Any, Any, Any, Any, DNS Query>]*
	A (DPF Allow List)	*[<All, All, All, All, DNS Response>]*
	Q (DPF Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	D (Security threat database update duration)	*4 hours*

Table 20 - Example of a DNS Protocol Filtering Security Function Policy

[6] Note that {0xC0,0x2C} is the cipher suite value for TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384.

Table 21 provides an example of a Domain Name Filtering Security Function Policy

Security Function Policy	Security Function Policy Parameter	Parameter Value
DNF-7	DNF Policy Identifier	*DNF-7*
	B (DNF Block List)	*[*.domain.tld]*
	A (DNF Allow List)	*[] (empty list)*
	Q (DNF Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	D (Security threat database update duration)	*4 hours*

Table 21 - Example of a Domain Name Filtering Security Function Policy

Table 22 provides an example of a URL Filtering Security Function Policy

Security Function Policy	Security Function Policy Parameter	Parameter Value
URLF-9	URLF Policy Identifier	*URF-9*
	B (URLF Block List)	*[host.domain.tld/section/*]*
	A (URLF Allow List)	*[] (empty list)*
	Q (URLF Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	D (Security threat database update duration)	*4 hours*

Table 22 - Example of a URL Filtering Security Function Policy

Table 23 provides an example of a Malware Detection and Removal Security Function Policy

Security Function Policy	Security Function Policy Parameter	Parameter Value
MD+R-11	MD+R Policy Identifier	*MD+R-11*
	B (MDR Block List)	['ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c'[7]]
	A (MDR Allow List)	*[] (empty list)*
	Q (MDR Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	D (security threat database update duration)	*4 hours*
	Dt (Detection type)	*Signature*
	Bh (Behavior)	*Remove Malware from the Object and Allow the associated subset of the Service Flow*

Table 23 - Example of a Malware Detection and Removal Security Function Policy

Table 24 provides an example of a Data Loss Prevention Security Function Policy.

[7] The SHA-256 file hash value for Poison Ivy, a known malicious Malware.

Security Function Policy	**Security Function Policy Parameter**	**Parameter Value**
DLP-13	*DLP Policy Identifier*	*DLP-13*
	B (DLP Block List)	*[“CPI”, “SSN”]*
	A (DLP Allow List)	*[] (empty list)*
	Q (DLP Quarantine List)	*[] (empty list)*
	Nm (No-match)	*Allow*
	Ns (No-scan)	*Allow.*
	PII, CPI removal behavior	*Remove the PII/CPI and Allow the rest of the Service Flow.*

Table 24 - Example of a Data Loss Prevention Security Function Policy

Editor Note 16: We need a good contribution to structure the PDNS policy example. Tim offered to provide a contribution, per Table 25. Four of the rows (supported and unsupported lists) in the table may need to change, depending how we agree to change Table 15.

Table 25 provides an example of a Protective DNS Security Function Policy.

Security Function Policy	**Security Function Policy Parameter**	**Parameter Type (host name, IP address)**	**Parameter Value**
PDNS-11	*PDNS Policy ID*		*PDNS-1*
	Sem (PDNS Supported List, DNS encryption methods)		
	Uem (PDNS Unsupported List, DNS encryption methods)		
	Srr (PDNS Supported List, DNSRR parameters)		
	Urr (PDNS Unsupported List, DNSRR parameters)		
	B (PDNS Block List)		
	A (PDNS Allow List)		
	Q (PDNS Quarantine List)		
	Nm (No-match)		
	Duration		

Table 25 - Example of a Protective DNS Security Function Policy

Appendix F Major Changes from MEF 88 (Informative)

The following lists the major changes from MEF 88 [3]:

- The term *Service Flow* is used throughout this document instead of the term *Application Flow*, since this document specifies Security Functions that can be used with any IP-based service, not only SD-WAN services.
- The terms Subject Actor and Target Actor are now appearing more in the document, with reference to MEF 118 [5], in the context of examples, e.g., informative text and some figures.
- The term *match criteria entry* is changed to *criteria entry*.
- Added requirements for Supported and Unsupported Lists into the Security Action Lists (sections 7.1.4 and 7.1.5).
- A Notification Action Modifier (Section 7.1.7) is added for Block List, Allow List and Quarantine List entries to allow the Subscriber to indicate which entries, when matched in a subset of a Service Flow, require notification to the Subscriber. Multiple levels of notification are also supported.
- The Security Event Notification (Section 7.2) is elaborated to:
  - change the notification requirement with respect to the Notification Action Modifier and the multiple levels of notification.
  - support multiple types of SEN, including IOC, ATT&CK, Other.
- A Security Admin Notification (Section 7.3) is specified to notify Subscribers of changes to Security Function Policies.
- Updated the notification requirement for SEN and SAN to include contact method(s) for each recipient in the list of recipients.
- Made several changes re: immediate communication to the client (basically no longer mandating immediate communication when a Security Function blocks a subset of the Service Flow).
- The term *Subscriber's Security Administrator* has been added to the terminology table.
- Informative text on DNS resolution (Section 7.4) is added to provide the capability to securely respond to customer DNS requests. This text then points to the DNS Protocol Filtering and the Protective DNS Security Functions for the specific requirements.
- Changed the format of the criteria entry for the following Security Functions from tuple to table.
  - IP, Port and Protocol Filtering Security Function (Table 5).
  - DNS Protocol Filtering Security Function (Table 6).
- Changed the format of the policy parameters for the following Security Functions from tuple to table.
  - Middlebox Security Function (Table 8)
  - IP, Port and Protocol Filtering Security Function (Table 9)
  - DNS Protocol Filtering Security Function (Table 14)
  - Domain Name Filtering Security Function (Table 10)

    - URL Filtering Security Function (Table 11)
    - Malware Detection and Removal Security Function (Table 12)
- The Middlebox Security Function (Section 8):
    - is elaborated to include the ability to decrypt and re-encrypt IPsec connections, in addition to TLS, which was specified in MEF 88 [3].
    - added a requirement relating to IPsec about when an invalid/unknown initiator and responder endpoint (IPsec) is detected.
    - Added informative text re: the behavior of unencrypted packets when the Middlebox Security function is enabled for a Service Flow containing unencrypted packets.
- Two new Security Functions are now specified:
    - Data Loss Prevention (Section 9.5), and
    - Protective DNS (Section 9.6.2)
- Consolidated the DNS-related security functions into one section (Section 9.6), applicable to Service Flows containing DNS messages. This new section includes subsections for the DNS Protocol Filtering Security Function (Section 9.6.1) and the Protective DNS Security Function (Section 9.6.2).
- The Security Function Policies specified in this document (Section 10) are now clearly atomic policies, each of which includes parameters that need to be agreed between the Subscriber and Service Provider. An IP Service specification that references this Standard determines how to incorporate these Security Function Policies into its service policy.
- Added editor's notes in Appendix A (Use Cases) mainly related to the Protective DNS Security Function: list of security functions needs updating, (Figure 9) showing the security functions needs updating, one or more use cases will get added featuring Service Flows with DNS messages.
- Added editor's note in Appendix E (Examples of Security Function Policies) related to Protective DNS Security Function - this needs a good example.

Appendix G Acknowledgements (Informative)

The following people participated in the development of this Standard and have requested to be included in this list.

Editor Note 17: This list will be put together just before going to Letter Ballot