MEF Standard

MEF 88

Application Flow Security for SD-WAN Services

November 2021

Disclaimer

The information in this publication is freely available for reproduction and use by any recipient and is believed to be accurate as of its publication date. Such information is subject to change without notice and MEF Forum (MEF) is not responsible for any errors. MEF does not assume responsibility to update or correct any information in this publication. No representation or warranty, expressed or implied, is made by MEF concerning the completeness, accuracy, or applicability of any information contained herein and no liability of any kind shall be assumed by MEF as a result of reliance upon such information.

The information contained herein is intended to be used without modification by the recipient or user of this document. MEF is not responsible or liable for any modifications to this document made by any other party.

The receipt or any use of this document or its contents does not in any way create, by implication or otherwise:

a) any express or implied license or right to or under any patent, copyright, trademark or trade secret rights held or claimed by any MEF member which are or may be associated with the ideas, techniques, concepts or expressions contained herein; nor

b) any warranty or representation that any MEF members will announce any product(s) and/or service(s) related thereto, or if such announcements are made, that such announced product(s) and/or service(s) embody any or all of the ideas, technologies, or concepts contained herein; nor

c) any form of relationship between any MEF member and the recipient or user of this document.

Implementation or use of specific MEF standards, specifications or recommendations will be voluntary, and no Member shall be obliged to implement them by virtue of participation in MEF Forum. MEF is a non-profit international organization to enable the development and worldwide adoption of agile, assured and orchestrated network services. MEF does not, expressly or otherwise, endorse or promote any specific products or services

Table of Contents

List of Figures

List of Tables

1 List of Contributing Members

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

- Bell Canada
- Ciena
- Cisco
- CMC Networks
- Fortinet
- Fujitsu
- Futurewei
- Nokia
- Oracle
- Orange
- PCCW Global
- Spirent
- Versa Networks

2 Abstract

This Standard specifies the requirements needed to add Application Flow Security to SD-WAN Services. As such, it is based on the SD-WAN Service Attributes and Service Framework, as specified in MEF 70.1 [2], where Application Flows are comprehensively defined. This Standard defines Security Policy as a set of parameters, the values of which are agreed between the Subscriber and Service Provider (as part of the SWVC List of Policies Service Attribute) and that specify which Security Functions are to be applied to an Application Flow. It also defines Security Functions that, when enabled, enforce Security Policies on a per-Application 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, or decryption and re-encryption by a Middle-Box Function of a TLS-encrypted Application Flow. The capabilities required to support these Security Functions are also defined: Block List, Allow List, Quarantine List and Security Event Notification (SEN).

3 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] and MEF 70.1 [2] 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 an Application Flow or a subset of an Application Flow to pass.	This document
Allow List	A list of match criteria entries (IP addresses, domain names, URLs, hashes, etc.) that when applied to an Application Flow permits the subset of the Application Flow that contains a match to one of the entries.	This document
Block	An action taken by a Security Function that does not permit an Application Flow or a subset of an Application Flow to pass.	This document
Block List	A list of match criteria entries (IP addresses, domain names, URLs, hashes, etc.) that when applied to an Application Flow denies the subset of the Application 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 [36]
Certificate Authority (CA)	An organization that is responsible for the creation, issuance, revocation, and management of Certificates.	CA Browser Forum [36]
Common Vulnerabilities and Exposures (CVE®)	A list of entries - each containing an identification number, a description, and at least one public reference - for publicly known cybersecurity vulnerabilities.	MITRE [37]
DNS Protocol Filtering (DPF)	The Security Function that determines whether an Application Flow, or subset of an Application Flow, contains Domain Name System (DNS) messages that are to be Allowed or Blocked.	This document
Domain Name Filtering (DNF)	The Security Function that determines whether an Application Flow, or subset of an Application Flow, contains domain names that are to be Allowed or Blocked.	This document
Hypertext Transfer Protocol (HTTP)	A stateless application-level protocol for distributed, collaborative, hypertext information systems.	RFC 7230 [19]

Term	Definition	Reference
Hypertext Transfer Protocol Secure (HTTPS)	HTTP over TLS.	RFC 2818 [7]
Indicators of compromise (IOC)	Forensic artifacts from intrusions that are identified on organizational information systems (at the host or network level).	NIST 800-53 [33]
IP, Port and Protocol Filtering (IPPF)	The Security Function that determines whether an Application Flow's source or destination IP addresses, source or destination port numbers, or IP protocols are to be Allowed or Blocked.	This document
Malware	Software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system.	This document
Malware Detection and Removal (MD+R)	The Security Function that determines whether an Application Flow, or subset of an Application Flow, contains Malware, and removes the Malware or Blocks the subset of the Application Flow containing the Malware.	This document
Middle-Box Function (MBF)	A function used to decrypt and re-encrypt secured sessions, e.g., TLS, in an Application Flow that allows Security Functions to apply to the unencrypted Application Flow.	This document
Public Key Infrastructure (PKI)	The architecture, organization, techniques, practices, and procedures that collectively support the implementation and operation of a certificate-based public key cryptographic system. Framework established to issue, maintain, and revoke public key certificates.	NIST 800-53 [33]
Quarantine List	A list of match criteria entries that are not on the Block List but are deemed suspicious. The subset of the Application Flow that contains a match to one of the entries on the Quarantine List is Blocked.	This document
Security Function	The component that, when enabled per the Security Policy, makes a decision to Allow or Block a subset of an Application Flow.	This document
Security Policy	A set of parameters that are agreed between the Subscriber and Service Provider (as part of the SWVC List of Policies Service Attribute) and that specify which Security Functions are to be applied to an Application Flow.	This document
Security Event Notification (SEN)	A communication of a security event, i.e., a SEN is issued when a subset of an Application Flow is Blocked or modified.	This document

Term	Definition	Reference
Service Provider	An organization that provides services to Subscribers. In this document, Service Provider means SD-WAN Service Provider.	MEF 70.1 [2]
Social Engineering	In an information security context, the psychological manipulation of people into performing actions or divulging confidential information.	This document
Transport Layer Security version 1.2 (TLS 1.2)	A cryptographic protocol designed to provide communications security over a computer network.	RFC 5246 [13]
Uniform Resource Identifier (URI)	An identifier consisting of a sequence of characters matching the syntax rule named in Section 3 of RFC 3986. It enables uniform identification of resources via a separately defined extensible set of naming schemes. A URI can be classified as a locator, a name or both.	RFC 3986 [11]
Uniform Resource Locator (URL)	The subset of URIs that, in addition to identifying a resource, provides a means of locating the resource by describing its primary access mechanism (e.g., its network "location").	RFC 3986 [11]
Universally Unique Identifier (UUID)	An identifier that is unique across both space and time with respect to the space of all UUIDs.	RFC 4122 [12]
URL Filtering (URLF)	The Security Function that determines whether an Application Flow, or subset of an Application Flow, contains a URL that is to be Allowed or Blocked.	This document
UTC	Coordinated Universal Time	RFC 3339 [8]

Table 1 – Terminology and Abbreviations

4 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 [6], RFC 8174 [23]) 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**.**

5 Introduction

Security is a top requirement for Subscribers of SD-WAN Services. Subscribers are investing more in cybersecurity due to the increasing quantity and types of threats, the data protection requirements imposed by regulations, the costs of fixing damaged systems caused by cybercrime, and related costs (e.g., loss of revenue, damaged reputation, and the cost to regain confidence) after a serious data breach. A Service Provider that provides Security Functions to mitigate such threats can help a Subscriber maintain their security posture over time.

In an SD-WAN Service that is enhanced with the Security Functions specified in this document, the following cyberthreats could be mitigated:

- Malware: Software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system. Examples include computer virus, computer worm, trojan horse, wiper attack, ransomware, etc.
- Social Engineering: In an information security context, the psychological manipulation of people into performing actions or divulging confidential information. Examples include phishing, scareware, etc.

This document focuses on Application Flow security - not infrastructure security (e.g., Denial of Service attacks or unauthorized access to the infrastructure). It is assumed that Application 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 SD-WAN service to which Application Flow Security is added.

The term Security Function is defined in this document as the component that, when enabled per the Security Policy, makes a decision to Allow or Block a subset of an Application Flow. Note that all the Security Functions specified in this document (see Sections 8 and 9) are in the context of an SD-WAN service, as specified in MEF 70.1 [2].

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

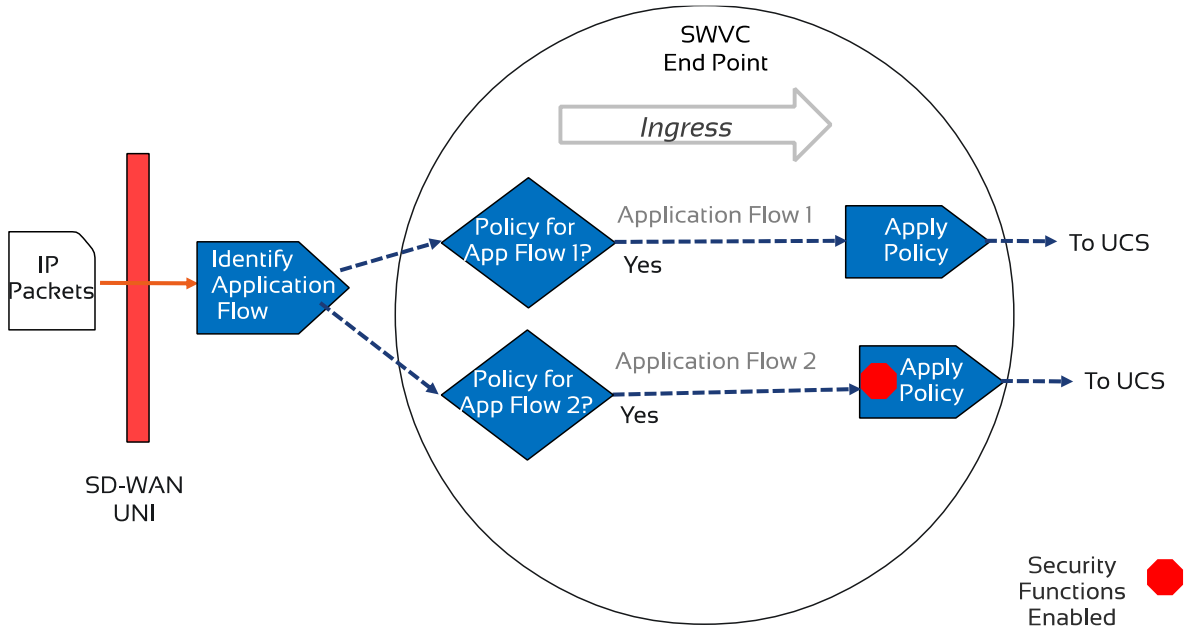


Figure 1 – Example of Application Flow Security for an Ingress Application Flow

In this example, Application Flow 1 carries traffic that does not require additional Security Functions defined in this document. Application 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 in the context of traditional SD-WAN Application Flow classification and forwarding. Security Functions can include any combination of Middle-Box Function, IP, Port and Protocol Filtering, DNS Protocol Filtering, Domain Name Filtering, URL Filtering, and Malware Detection and Removal, as appropriate to the Application Flows that are being sent and received. Security Functions can be enabled for Ingress Application Flows and/or Egress Application Flows.

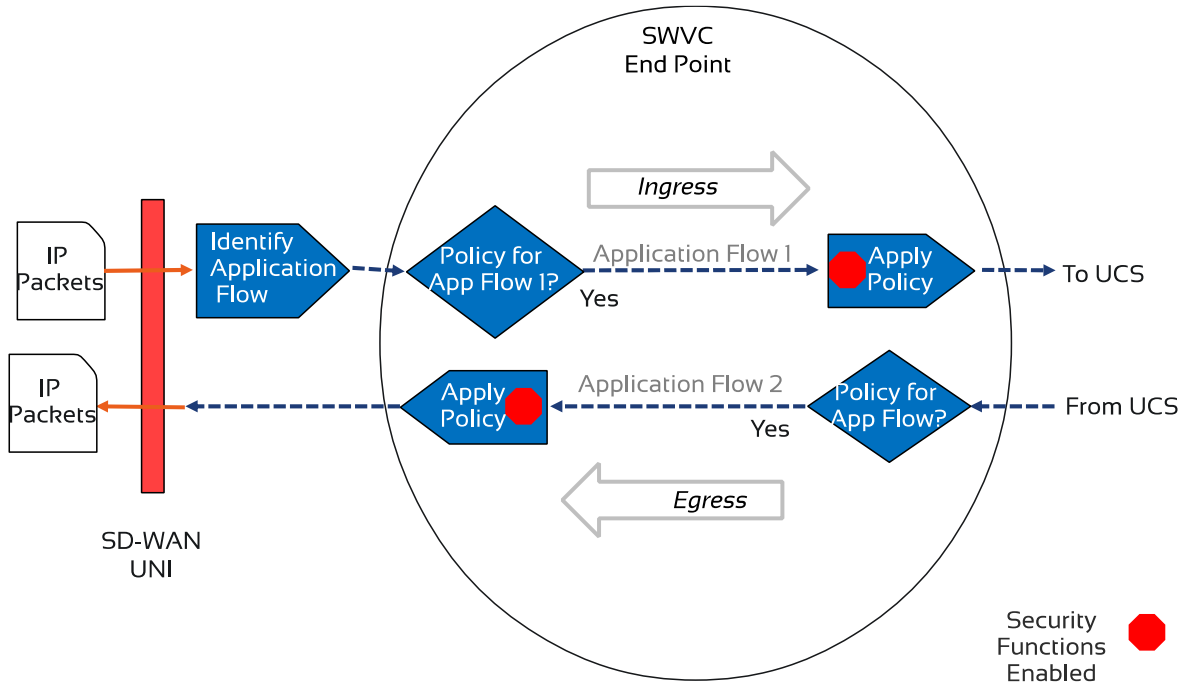


Figure 2 – Example of the Relationship of Security Functions and Application Flows

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

Section 6 defines Security Policy as a set of parameters that are agreed between the Subscriber and Service Provider (as part of the SWVC List of Policies Service Attribute), and that specify which Security Functions are to be applied to a Subscriber's Application Flows.

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 in this section.
- Security Event Notification (SEN): notifies the Subscriber personnel when security actions are taken.

The Middle-Box Function (MBF), when enabled for a given Application Flow, decrypts the Transport Layer Security (TLS) encrypted subset of the Application Flow to allow for scanning and then re-encrypts it. In this document, use of the term *TLS* refers to TLS 1.2, as specified in IETF 5246 [13], unless another version is specified. MBF is a Security Function in a special class, 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 MBF 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 Application Flow, can Allow or Block access based on source or destination IP address, source, or destination port number and/or IP protocol.
- DNS Protocol Filtering, when enabled for a given Application Flow, can Allow or Block access based on DNS Message Type and source/destination IP addresses.
- Domain Name Filtering, when enabled for a given Application Flow, can Allow or Block access to a list of domains. Wildcard entries are supported.
- URL Filtering, when enabled for a given Application Flow, can Allow or Block access to a list of URLs. Wildcard entries are supported.
- Malware Detection and Removal, when enabled for a given Application Flow, scans Objects for Malware and takes appropriate action when Malware is detected.

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 an Application Flow or a subset of an Application Flow to pass. Block is defined as an action taken by a Security Function that does not permit an Application Flow or a subset of an Application Flow to pass. Note that these actions apply to each Security Function independently, i.e., for a given Application Flow, one Security Function might Allow, while another Security Function might Block. In this case, the overall effect is to deny. See Appendix A for use case examples of how these Security Functions interact with different Application Flows.

The remainder of this document is organized as follows:

- References are listed in Section 10.
- Use cases relating Application Flows and Security Functions are described in Appendix A.
- Examples of Malware Detection and Removal are described in Appendix B.
- A summary description of a detailed threat modeling report is provided in Appendix C, together with a link to the threat modeling tool output.
- An example of a Security Policy is provided in Appendix D.

Readers should review the related threat model document, referenced in Appendix C, to better familiarize themselves with the overriding threat model of SD-WAN 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 MBF context, can be found in section 8. In particular, readers

should familiarize themselves with [R39], which covers constraints on how data should be processed if it leaves the MBF.

This document does not impose any explicit constraints on where in the SD-WAN 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 an Application Flow, all possible paths have the same security scanning. A Service Provider may distribute Security Functions, e.g., in a cloud, edge computing node, Customer Premises Equipment (CPE) or virtual CPE (vCPE), depending on the functionality and where it can be optimally placed. This document also does not address requirements to secure the SD-WAN network itself, or the encrypted Tunnel Virtual Connections (TVCs). 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.

6 Security Policy

MEF 70.1 [2] specifies the List of Security Policies as a Service Attribute that is agreed between the Subscriber and the Service Provider. This section defines the content of each Security Policy that is included in the List of Security Policies Service Attribute (and hence is agreed between the Subscriber and the Service Provider). Each Security Policy may be referenced in one or more SD-WAN Policies, using the AF-SECURITY-INGRESS (for Ingress SD-WAN Policies) or AF-SECURITY-EGRESS (for Egress SD-WAN Policies) Policy Criteria defined in MEF 70.1 [2]. When an SD-WAN Policy is applied to an Application Flow, the Security Policy referenced in the SD-WAN Policy (if any) is also applied to that Application Flow.

This document defines a Security Policy as a set of parameters that are agreed between the Subscriber and Service Provider (as part of the SWVC List of Policies Service Attribute) and that specify which Security Functions are to be applied to an Application Flow. A Security Policy could be referenced in zero or more Ingress (SD-WAN) Policies and zero or more Egress (SD-WAN) Policies. A Security Policy is expressed as an 8-tuple <*ID, SEN, MBF, IPPF, DPF, DNF, URLF, MD+R*>, with a description of each in the following subsections.

[R1] When a Security Policy is in force for a given Application Flow, the Service Provider **MUST** inform the Subscriber of any expected impact to the SD-WAN service performance.

An example of a Security Policy is provided in Appendix D.

A Subscriber could have more than one Security Policy. Within the scope of the Subscriber's service, each Security Policy has a unique identifier. The requirements related to the Security Policy Identifier are specified in Section 6.1.

6.1 Security Policy Identifier Parameter

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

[R2] The Security Policy Identifier **MUST** be an Identifier String.

[R3] Each Security Policy Identifier **MUST** be unique among all Security Policies for a given SWVC.

[R4] The value of the Security Policy Identifier **MUST NOT** be none.

6.2 Security Event Notification Parameter

A Security Event Notification (SEN) is a notification to the Subscriber of a security event. See Section 7.2 for a detailed specification of the SEN.

SEN is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the SEN. The value of *SEN* is either *None* or a two-tuple of the

form (*R, T*), where *R* is a list of recipients authorized to receive the SEN, and *T* is the format of the timestamp of the SEN, e.g., date and time in UTC or local time.

Note that the value *None* indicates that the SEN is not needed for this Security Policy.

6.3 MBF Parameter

The MBF Security Function provides for the decryption and re-encryption of Application Flows that use Transport Layer Security (TLS). See Section 8 for a detailed specification of the MBF Security Function.

MBF is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the MBF Security Function. The value of *MBF* is either *None* or a seven-tuple of the form (*S*, *U*, *B*, *A*, *Nm*, *CA*, *I*), where

- *S* is the list of match criteria entries in the MBF Supported List, as specified in Section 8.
- *U* is the list of match criteria entries in the MBF Unsupported List, as specified in Section 8.
- *B* is the list of match criteria entries in the MBF Block List, as specified in Section 8.
- *A* is the list of match criteria entries in the MBF Allow List, as specified in Section 8.
- *Nm* is the parameter that determines the behavior when a subset of the Application Flow does not match a match criteria entry on either the MBF Block List or the MBF Allow List. The possible values are Allow or Block.
- *CA* is the list of the Subscriber's trusted CAs.
- *I* is the parameter that determines the behavior of the MBF when the target server certificate is invalid. The possible values are Allow or Block.

Note that the value *None* indicates that the MBF Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

6.4 IP, Port and Protocol Filtering Parameter

The IP, Port and Protocol Filtering (IPPF) Security Function provides for the filtering of Application Flows with respect to IP addresses, port numbers and IP protocols. See Section 9.1 for a detailed specification of the IPPF Security Function.

IPPF is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the IPPF Security Function. The value of *IPPF* is either *None* or a five-tuple of the form (*B*, *A*, Q, *Nm*, *D*), where

- *B* is the list of match criteria entries in the IPPF Block List, as specified in Section 9.1.

- *A* is the list of match criteria entries in the IPPF Allow List, as specified in Section 9.1.
- *Q* is the list of match criteria entries in the IPPF Quarantine List, as specified in Section 9.1.
- *Nm* is the parameter that determines the behavior when a subset of the Application Flow does not match a match criteria entry on any of the IPPF Lists. The possible values are Allow or Block.
- *D* is the duration of time between updates of the IPPF security threat database.

Note that the value *None* indicates that the IPPF Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

6.5 DNS Protocol Filtering Parameter

The DNS Protocol Filtering (DPF) Security Function provides for the filtering of Application Flows with respect to IP addresses of DNS servers and DNS message types. See Section 9.2 for a detailed specification of the DPF Security Function.

DPF is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the DPF Security Function. The value of *DPF* is either *None* or a five-tuple of the form (*B*, *A*, Q, *Nm*, *D*), where

- *B* is the list of match criteria entries in the DPF Block List, as specified in Section 9.2.
- *A* is the list of match criteria entries in the DPF Allow List, as specified in Section 9.2.
- *Q* is the list of match criteria entries in the DPF Quarantine List, as specified in Section 9.2.
- *Nm* is the parameter that determines the behavior when a subset of the Application Flow does not match a match criteria entry on any of the DPF Lists. The possible values are Allow or Block.
- *D* is the duration of time between updates of the DPF security threat database.

Note that the value *None* indicates that the DPF Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

6.6 Domain Name Filtering Parameter

The Domain Name Filtering (DNF) Security Function provides for the filtering of Application Flows with respect to domain names. See Section 9.3 for a detailed specification of the DNF Security Function.

DNF is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the DNF Security Function. The value of *DNF* is either *None* or a five-tuple of the form (*B*, *A*, Q, *Nm*, *D*), where

- *B* is the list of match criteria entries in the DNF Block List, as specified in Section 9.3.
- *A* is the list of match criteria entries in the DNF Allow List, as specified in Section 9.3.
- *Q* is the list of match criteria entries in the DNF Quarantine List, as specified in Section 9.3.
- *Nm* is the parameter that determines the behavior when a subset of the Application Flow does not match a match criteria entry on any of the DNF Lists. The possible values are Allow or Block.
- *D* is the duration of time between updates of the DNF security threat database.

Note that the value *None* indicates that the DNF Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

6.7 URL Filtering Parameter

The URL Filtering (URLF) Security Function provides for the filtering of Application Flows with respect to URLs. See Section 9.4 for a detailed specification of the URLF Security Function.

URLF is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the URLF Security Function. The value of *URLF* is either *None* or a five-tuple of the form (*B*, *A*, Q, *Nm*, *D*), where

- *B* is the list of match criteria entries in the URLF Block List, as specified in Section 9.4.
- *A* is the list of match criteria entries in the URLF Allow List, as specified in Section 9.4.
- *Q* is the list of match criteria entries in the URLF Quarantine List, as specified in Section 9.4.
- *Nm* is the parameter that determines the behavior when a subset of the Application Flow does not match a match criteria entry on any of the URLF Lists. The possible values are Allow or Block.
- *D* is the duration of time between updates of the URLF security threat database.

Note that the value *None* indicates that the URLF Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

6.8 Malware Detection and Removal Parameter

The Malware Detection and Removal (MD+R) Security Function provides for the identification of Malware in an Application Flow and removal of the Malware or blocking of the subset of the Application Flow containing the Malware. See Section 9.5 for a detailed specification of the MD+R Security Function.

MD+R is a parameter of a Security Policy that documents the agreement between the Subscriber and Service Provider relating to the MD+R Security Function. The value of *MD+R* is either *None* or a seven-tuple of the form (*B*, *A*, Q, *Nm*, *D, Dt, Bh*), where

- *B* is the list of match criteria entries in the MD+R Block List, as specified in Section 9.5.
- *A* is the list of match criteria entries in the MD+R Allow List, as specified in Section 9.5.
- *Q* is the list of match criteria entries in the MD+R Quarantine List, as specified in Section 9.5.
- *Nm* is the parameter that determines the behavior when there is no match to any of the MD+R Lists. The possible values are Allow or Block.
- *D* is the duration of time between updates of the MD+R security threat database.
- *Dt* is the detection type (e.g., signature scan or behavioral analysis)
- *Bh* is the Malware removal behavior, as specified in [R76]

Note that the value *None* indicates that the MD+R Security Function is *Disabled* for all Application Flows to which this Security Policy is applied.

7 Required Capabilities in Support of Security Functions

The Application Flow 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 and Quarantine List are specified in Section 7.1.
- Security Event Notification (SEN) requirements are specified in Section 7.2.

7.1 Security Action Lists

Security Action Lists can be used by each Security Function to take a filtering action. The three lists specified in this section are: Block List, Allow List and Quarantine List. The term "match criteria entry" is used to indicate an entry in a list. Depending on the Security Function to which it applies, a match criteria entry could be a specific port number, protocol identifier, URL, domain name, IP address, hash or some other identifier within the context of an Application Flow. A given Security Function can have three types of ordered lists in which different match criteria entries are placed, based on user preferences, the match criteria entry's reputation, or other factors.

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

7.1.1 Block List

A Block List is a list of match criteria entries that when applied to an Application Flow denies the subset of the Application Flow that contains a match to one of the entries. There is a different Block List for each of the Security Functions, and when used for a given Security Function, it always includes the name of the Security Function up front, e.g., the URL Filtering Block List. The Service Provider is responsible for enforcing the Block List.

The list of match criteria entries in the Block List include:

- Match criteria entries that are deemed a security threat by the Subscriber
- Match criteria entries that are deemed a security threat by the Service Provider
- Match 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 match criteria entries. It is not practical for the Subscriber to provide complete/exhaustive match 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 match 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 match 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 match criteria entries for each category. The database of detailed match criteria entries within each category is also dynamic.

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.

[R5] For each Security Function, the Service Provider **MUST** maintain a list of match criteria entries in the Block List.

Once the service is turned up, the match 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.

[R6] For each Security Function, the Service Provider **MUST** allow the Subscriber to add match criteria entries to the Block List.

[R7] For each Security Function, the Service Provider **MUST** allow the Subscriber to remove a match 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.

[R8] For each Security Function, the Service Provider **MUST NOT** allow the Subscriber to remove a match 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 time frame used to update it, e.g., hourly, daily, weekly, per change.

[R9] For each Security Function, the Service Provider **MUST** provide to the Subscriber the time frame used to update 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 match criteria entry in the list (e.g., a typo).

[R10] 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 match criteria entries that when applied to an Application Flow permits the subset of the Application Flow that contains a match to one of the entries. There is a different Allow List for each of the Security Functions, and when used for a given Security Function, it always includes the name of the Security Function up front, e.g., the URL Filtering Allow List. The Service Provider is responsible for enforcing the match criteria entries on the Allow List. The match criteria entries in the Allow List include:

- Match criteria entries that are explicitly identified by the Subscriber

The Subscriber can provide explicit match 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 match criteria entries to the Allow List.

If a match criteria entry on the Allow List becomes a possible security threat, e.g., one URL in the corporate domain has been compromised, that match 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.

[R11] For each Security Function, the Service Provider **MUST** maintain a list of match criteria entries in the Allow List.

[R12] For each Security Function, the Service Provider **MUST** allow the Subscriber to add or remove match criteria entries in the Allow List.

7.1.3 Quarantine List

A Quarantine List is a list of match criteria entries that are not on the Block List but are deemed suspicious. The subset of the Application Flow that contains a match to one of the match criteria entries on the Quarantine List is denied. There is a different Quarantine List for each of the Security Functions, and when used for a given Security Function, it always includes the name of the Security Function up front, e.g., the URL Filtering Quarantine List. The match criteria entries in the Quarantine List include:

- Match criteria entries that are deemed suspicious by the Subscriber
- Match 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 match 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 match criteria entry (in this case a specific URL) that was on the URL Filtering Allow List, the Service Provider removes that match 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 SWVC, the Service Provider might determine that a particular Application Flow is compromised. If the Service Provider can quantify the compromise into a set of match criteria entries, the Service Provider would add those match criteria entries to the Quarantine List, removing match criteria entries from the Allow List in cases of conflict to comply with [R15].

While monitoring the Subscriber Network, the Subscriber might determine that a particular Application Flow is compromised. If the Subscriber can quantify the compromise into a set of match criteria entries, the Subscriber can either add those match criteria entries to the Block List or to the Quarantine List, removing match criteria entries from the Allow List in cases of conflict to comply with [R15]. 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 match 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 feed (e.g., vulnerabilities and IOCs), then the Service Provider would remove the set of match criteria entries from the Quarantine list and add them to the Block List. However, if the set of match criteria entries is not deemed severe enough to be added to the threat intelligence feed, then the Service Provider would not remove the set of match 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 match criteria entries remain on the Quarantine List until the Subscriber removes those entries.

The effect of this is that suspicious entries are added to the Quarantine List dynamically, and the Subscriber has the final decision to remove them from the Quarantine List with only one

exception, elevation to the threat intelligence feed (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 match criteria entry from the Quarantine List to the Block List, or
- Explicitly move the match criteria entry to the Allow List, or
- Ignore the match criteria entry and leave it in the Quarantine List.

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

[R13] For each Security Function, the Service Provider **MUST** maintain a list of match criteria entries in the Quarantine List.

[R14] For each Security Function, the Service Provider **MUST** allow the Subscriber to do each of the following:

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

7.1.4 Requirements Pertaining to All of the Lists

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

[R15] For each Security Function, the Service Provider **MUST** ensure that each match criteria entry is on at most one list.

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

Example 1: A match criteria entry on one list could be generalized, e.g., all URLs in www.qaz.com/world are to be permitted, and so the match 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 match criteria entry overrules the more general match 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 might deem suspicious a match 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 match criteria entry and decide to place it back on the Allow 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.

7.2 Security Event Notification (SEN)

A Security Event Notification (SEN) is a communication of a security event, i.e., a SEN is issued when a subset of an Application Flow is Blocked or modified. The SEN is sent to the Subscriber.

The SEN includes an Indicator of Compromise (IOC). 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 800-53 [33], "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."

[R16] A SEN **MUST** be issued whenever a subset of the Application Flow is Blocked or modified.

An example of a subset of an Application Flow that has been modified is the removal of an infected file attachment in an e-mail, and typically the removed file attachment would be replaced with a message stating that the file was infected and removed.

[R17] 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.

[R18] A SEN **MUST** include the items listed in Table 2.

Item	Value	Comments
Issuer	UTF-8 [10] String[1]	Examples: Service Provider Name, Security Vendor Name, etc.
Timestamp of IOC	date-time	RFC 3339 [8] Example: UTC
SEN ID	UUID	RFC 4122 [12] Universally Unique Identifier
Zone ID	UTF-8 [10] String	MEF 70.1 [2]
Source IP address	Human readable IPv4 dotted decimal IPv6 hexadecimal strings	IANA Number Resources [28]
SD-WAN UNI ID	UTF-8 [10] String	MEF 70.1 [2]
IOC Type	UTF-8 [10] String	Examples: CVE [37], STIX [40], CWE [35], CAPEC [39], ATT&CK [38], RFC 7970 [22].
IOC Information ID	UTF-8 [10] String	Identifies the IOC based on type
IOC Source	URL for the IOC type	CVE [37]
Type of Compromise	UTF-8 [10] String	Examples: known vulnerability, breach, data leakage, abuse of resources, jacking, where to find more information on the breach.
Compromise details	UTF-8 [10] String	Examples: Username, Source/Destination IP address, Source Media Access Control (MAC) address, neutralized URL, neutralized domain, Malware, Source/Destination port number, anomalous behavior.
Action Taken	UTF-8 [10] String	Examples: informational, quarantined or Blocked, Malware removed

Table 2 – Items to be included in a SEN

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 [11], 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.

[R19] Any domain name or URL in a SEN **MUST** be neutralized.

[D1] The method for neutralizing the domain name or URL in a SEN **SHOULD** use square brackets around each period.

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

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

[R20] The Service Provider **MUST** support UTC for the timestamp format.

[R20] means that the Service Provider needs to be able to use UTC format for the timestamp if the Subscriber wants that, although for a particular service or a particular Subscriber, use of a different time format could be agreed upon.

A SEN is used to notify interested parties of a security event.

[R21] A SEN **MUST** have a recipient list, which is agreed between the Service Provider and the Subscriber.

For example, the recipient list could include a Security Operations Center, a Network Operations Center and/or an individual.

8 Middle-Box Function (MBF)

Many Security Functions can only work when the Application Flow is unencrypted. An encrypted Application 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, MBF is a Security Function used to decrypt and re-encrypt one or more secured sessions, e.g., TLS, in an Application Flow, and that allows other Security Functions to apply to the unencrypted Application 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 Application Flow is decrypted using an MBF, it means that both decryption and re-encryption are performed on the Application Flow.

In the context of the MBF, the term 'client' is used to reference "the application entity that initiates a TLS connection to a server", and the term 'server' is used to reference "the application entity that responds to requests for connections from clients", per RFC 5246 [13].

For a given Application Flow, MBF can be either *Enabled* or *Disabled*, based upon agreement between the Subscriber and Service Provider.

Four lists (MBF Supported List, MBF Unsupported List, MBF Allow List and MBF Block List) are maintained by the Service Provider for each Application Flow, with each match criteria entry on a list using a 2-tuple of the form <*P, CS*>, where *P* is a specific TLS protocol version and *CS* is the list of cipher suites for that TLS protocol version. *Any* can be used to indicate that any cipher suite for a given TLS protocol version is on the list.

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 [27] provides a full list of cipher suites.

The MBF Supported List is a list of match criteria entries that specifies that the MBF is capable of decrypting the subset of the Application Flow that contains a match to one of the match criteria entries.

[R22] Based on agreement with the Subscriber, the Service Provider **MUST** maintain a list of match criteria entries in the MBF Supported List.

The MBF Unsupported List is a list of match criteria entries that specifies that the MBF is not capable of decrypting the subset of the Application Flow that contains a match to one of the match criteria entries.

[R23] Based on agreement with the Subscriber, the Service Provider **MUST** maintain a list of match criteria entries in the MBF Unsupported List.

[R24] The Service Provider **MUST** ensure that a match criteria entry on the MBF Supported List cannot also appear on the MBF Unsupported List.

The MBF Block List is a list of match criteria entries, which specifies that the MBF Blocks the subset of the Application Flow that contains a match to one of the entries, e.g., a TLS protocol and specific cipher suite is Blocked.

The MBF Allow List is a list of match criteria entries, which specifies that the MBF Allows the subset of the Application Flow that contains a match to one of the entries, e.g., a TLS protocol and specific cipher suite is Allowed.

Requirements related to the MBF Block List and the MBF Allow List are found it Section 7.1 of this document.

[R25] When MBF is *Enabled* for a given Application Flow, the Service Provider **MUST** meet the mandatory requirements specified in Section 7.1 of this document relating to the MBF Block List and the MBF Allow List.

The match criteria entries on the MBF Supported List and the MBF Unsupported List need to be included in the MBF Allow List or the MBF Block List, and vice versa. The following requirements apply.

[R26] The Service Provider **MUST** ensure that each match criteria entry on the MBF Supported List is also on either the MBF Allow List or the MBF Block List.

[R27] The Service Provider **MUST** ensure that each match criteria entry on the MBF Unsupported List is also on either the MBF Allow List or the MBF Block List.

[R28] The Service Provider **MUST** ensure that each match criteria entry on the MBF Allow List is also on either the MBF Supported List or the MBF Unsupported List.

[R29] The Service Provider **MUST** ensure that each match criteria entry on the MBF Block List is also on either the MBF Supported List or the MBF Unsupported List.

When MBF is *Enabled* for a given Application Flow, there are four cases to consider.

- Case 1: A subset of the Application Flow matches a match criteria entry on the MBF Block List. The MBF Blocks this subset.
- Case 2: A subset of the Application Flow matches a match criteria entry on the MBF Allow List and on the MBF Supported List. The MBF decrypts this subset, allowing for Security Functions to be applied to the unencrypted subset, and then re-encrypts the subset for transport in the Application Flow.

- Case 3: A subset of the Application Flow matches a match criteria entry on the MBF Allow List and on the MBF Unsupported List. The MBF passes this subset through, unchanged.
- Case 4: A subset of the Application 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 MBF unchanged or to Block it.

The following requirements specify the MBF behavior for these cases.

[R30] When MBF is *Enabled* for a given Application Flow, the subset of the Application Flow that matches a match criteria entry on the MBF Block List **MUST** be Blocked.

[R31] When MBF is *Enabled* for a given Application Flow, the subset of the Application Flow that matches a match criteria entry on the MBF Allow List and on the MBF Supported List **MUST** be decrypted.

[R32] When MBF is *Enabled* for a given Application Flow, the subset of the Application Flow that matches a match criteria entry on the MBF Allow List and on the MBF Unsupported List **MUST** be passed through the MBF without change.

It is possible that a subset of the Application Flow does not match a match criteria entry on any of these lists. Requirements [R33] and [R34] cover this gap.

[R33] When MBF is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the MBF lists:

- Block the subset of the Application Flow
- Pass the subset of the Application Flow through the MBF without change

[R34] When MBF is *Enabled* for a given Application Flow, the MBF Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the MBF lists:

- Block the subset of the Application Flow
- Pass the subset of the Application Flow through the MBF without change

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

Some protocols present challenges for an MBF to support. For example, the HTTP Strict Transport Security (HSTS) protocol, as specified in IETF RFC 6797 [16], can run over TLS and provides a mechanism implemented at the server to expressly prevent an MBF. Another example is QUIC, as specified in IETF RFC 9000 [26], that uses TLS 1.3 for secure transport.

[R35] The MBF **MUST** meet the mandatory requirements of TLS 1.2, per RFC 5246 [13].

[R36] The MBF **MUST** meet the mandatory requirements of section 9.3 of RFC 8446 [24] (Protocol Invariants).

This document does not require the MBF to support TLS 1.3 (see RFC 8446 [24]). For the case where TLS 1.3 is not supported by the MBF, i.e., TLS 1.3 and the associated cipher suites are on the MBF Unsupported List, the Subscriber who wants to use TLS 1.3 needs to ensure that the match criteria entries related to TLS 1.3 are on the MBF Allow List.

[R37] When MBF is *Enabled* for a given Application Flow, the MBF **MUST NOT** change the TLS protocol version for a TLS session as compared to the client request.

[R38] When MBF is *Enabled* for a given Application Flow, the MBF **MUST NOT** choose a weaker cipher suite in the negotiation for a TLS session as compared to the TLS session without the MBF.

NIST 800-52 [32] is a useful reference that provides a comprehensive set of guidelines for the selection, configuration, and use of Transport Layer Security (TLS) Implementations for US federal systems.

Figure 3 depicts an MBF used between a Subscriber's client and a server - the server can be either a public Internet host or an internal private host in the Subscriber's network. For simplicity, the SD-WAN network is not shown here. As depicted in Figure 3, a certificate chain consists of all the certificates needed to certify the MBF's certificate. This includes the MBF's proposed certificate, the certificate(s) of any intermediate CAs, and the certificate of a root CA trusted by all parties in the chain.

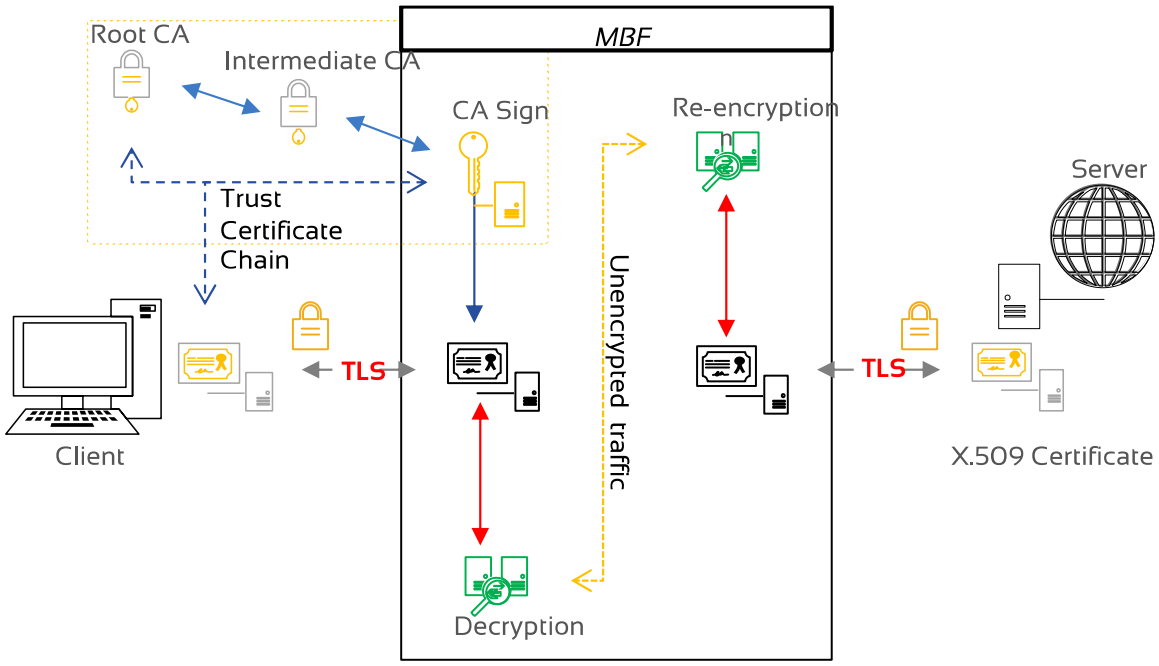


Figure 3 – Example of an MBF used between a Subscriber's client and a Server

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

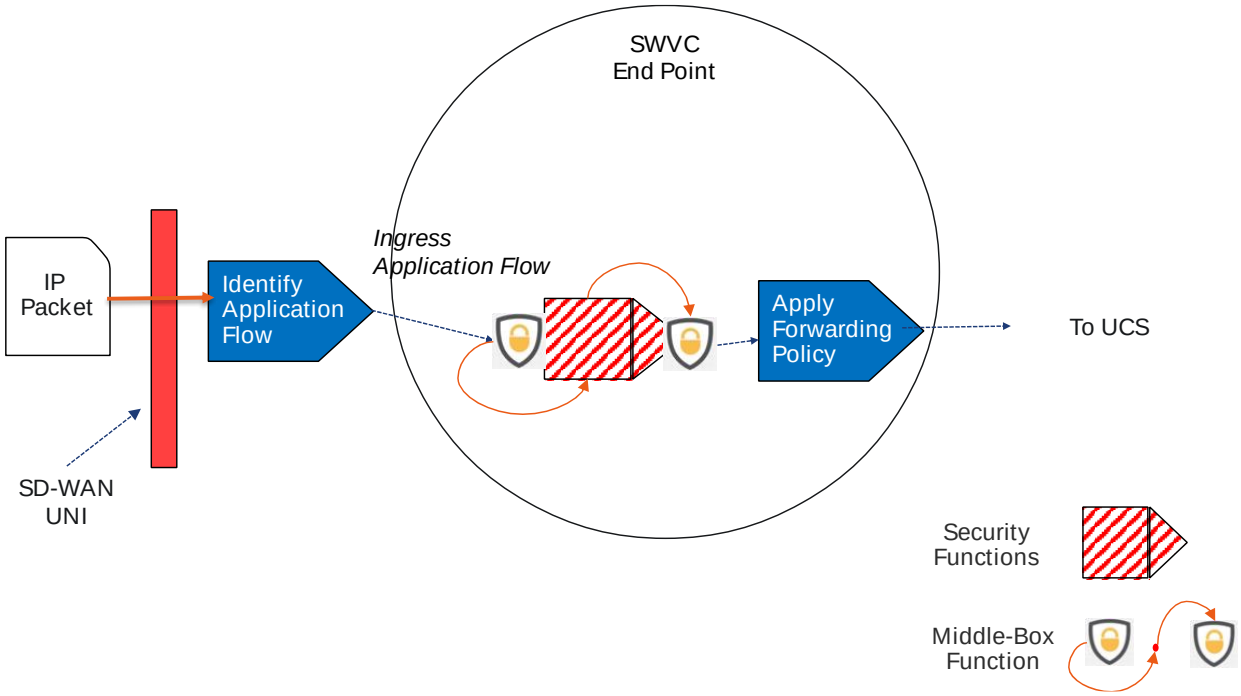


Figure 4 – Example of using MBF and Security Functions for an Application Flow

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

[R39] An Application Flow decrypted by the MBF **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 Application Flow.

The expectation is that the MBF 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 Application 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.

[R40] The MBF **MUST** be capable of issuing a valid, signed certificate for each TLS session with a CA that is trusted by the Subscriber.

[R41] When performing TLS inspection within the MBF, 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.

[R42] The Service Provider **MUST** ensure that the certificate chain allowing for this validation to occur, as described in [R41], 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.

[R43] Server certificates **MUST** be generated and regenerated with fresh, suitably random material per the requirements in FIPS-140-2 [31] for which the MBF processes Application Flows.

[R44] All replacement certificate properties, e.g., alternative server names, validity periods, and choices of cipher suites **MUST NOT** reduce the level of security functionality of the properties of the certificate being replaced.

[R45] Where a CA is operated in support of the TLS inspection within the MBF, it **MUST** clearly identify itself within the visible issuer properties (as defined in Section 4.1.2.4 of RFC 5280 [14]) of the certificates that it signs for reasons of

transparency, so that the Subscriber can identify when MBF is in the path versus when they are connecting directly to the originating server.

[D2] Any TLS based interception being performed by an MBF 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 [36] where certificates are securely created, used, revoked and destroyed.

[D3] Where possible, CAs **SHOULD** log any certificates that they issue using the standardized Certificate Transparency (CT) security standard, see RFC 6962 [18].

Per [D3], 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.

[R46] The MBF **MUST** be capable of accepting a valid, signed Subscriber certificate for each TLS session between the MBF and the Subscriber's server.

[R47] When an MBF is *Enabled* for a given Application Flow, the MBF **MUST** verify the validity of the targeted server certificate of the TLS session, as illustrated in Figure 3.

[R48] When an invalid server certificate is detected, an MBF **MUST** be capable of Blocking the TLS session and notifying the client in the Subscriber's network of the invalid certificate.

The method for notifying the client in the Subscriber's network 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 800-57 [34].

9 Security Functions

The Security Functions specified in this section can be applied to an Application Flow (Ingress or Egress) at a given SWVC End Point, and consist of:

- IP, Port and Protocol Filtering - detailed specification in Section 9.1
- DNS Protocol Filtering - detailed specification in Section 9.2
- Domain Name Filtering - detailed specification in Section 9.3
- URL Filtering - detailed specification in Section 9.4
- Malware Detection and Removal - detailed specification in Section 9.5

Each Security Function can be either *Enabled* or *Disabled* for a given Security Policy, based upon agreement between the Subscriber and Service Provider. A Security Function is considered to be *Enabled* or *Disabled* for an Application Flow, based on the Security Policy that is applied to the Application Flow.

9.1 IP, Port and Protocol Filtering

IP, Port and Protocol Filtering is defined as the Security Function that determines whether an Application Flow's source or destination IP addresses, source or destination port numbers, or IP protocols are to be Allowed or Blocked. Certain IP addresses, IP protocols and/or port numbers are commonly used by the Subscriber for use in their internal network, but these are generally not Allowed on an Application Flow that uses Internet Breakout. A Security Policy might also 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 match criteria entries used by the IP, Port and Protocol Filtering Security Function to Block the subset of the Application Flow that contains a match to one of the entries. An IP, Port and Protocol Filtering Allow List consists of a list of match criteria entries used by the IP, Port and Protocol Filtering Security Function to Allow the subset of the Application Flow that contains a match to one of the entries. An IP, Port and Protocol Filtering Quarantine List is a list of match criteria entries that are not on the IP, Port and Protocol Filtering Block List but are deemed suspicious. The subset of the Application 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, with each match criteria entry on a list using an 8-tuple of the form <*SAV4, DAV4, PROTV4, SAV6, DAV6, NEXTHEADV6, SPORT, DPORT*>.

- *SAV4* is the list of IPv4 source addresses. *Any* could be used to indicate that any source IPv4 address is on the list.
- *DAV4* is the list of IPv4 destination addresses. *Any* could be used to indicate that any destination IPv4 address is on the list.
- *PROTV4*, the IPv4 protocol list, is a list of integers in the range 0 to 255 or a list of keywords from IANA, Protocol Numbers [29], or a mix of integers and keywords.

- *SAV6* is the list of IPv6 source addresses. *Any* could be used to indicate that any source IPv6 address is on the list.
- *DAV6* is the list of IPv6 destination addresses. *Any* could be used to indicate that any destination IPv6 address is on the list.
- *NEXTHEADV6*, IPv6 protocol list, is a list of integers in the range 0 to 255 or a list of keywords from IANA, Protocol Numbers [29], or a mix of integers and keywords.
- *SPORT*, Transport Source Port, 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 [30] or a mix of integers and service names. *Any* could be used to indicate that any source port number is on the list.
- *DPORT*, Transport Destination Port, 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 [30] or a mix of integers and service names. *Any* could be used to indicate that any destination port number is on the list.

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

- A match 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 Application Flow that contains TCP and destination port 53 would be Blocked for both IPv4 and IPv6.
- A match 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 Application Flow that encapsulates GRE over IPv4 or IPv6 would be Blocked.
- A match 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 Application 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.

[R49] When IP, Port and Protocol Filtering is *Enabled* for a given Application 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.

[R50] When IP, Port and Protocol Filtering is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the IP, Port and Protocol Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R51] When IP, Port and Protocol Filtering is *Enabled* for a given Application Flow, the IP, Port and Protocol Filtering Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the IP, Port and Protocol Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R52] When IP, Port and Protocol Filtering is *Enabled* for a given Application 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 Application Flow that matches a match criteria entry on the IP, Port and Protocol Filtering Allow List
- Allow the subset of the Application Flow that does not match a match criteria entry on any of the IP, Port and Protocol Filtering lists, per the second bullet of [R51]
- Block the subset of the Application Flow that matches a match criteria entry on the IP, Port and Protocol Filtering Block List
- Block the subset of the Application Flow that matches a match criteria entry on the IP, Port and Protocol Filtering Quarantine List
- Block the subset of the Application Flow that does not match a match criteria entry on any of the IP, Port and Protocol Filtering lists, per the first bullet of [R51]

9.2 DNS Protocol Filtering

DNS Protocol Filtering is defined as the Security Function that determines whether an Application Flow, or subset of an Application 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 [17] messages are specified in RFC 1035 [4], RFC 1996 [5] and RFC 6895 [17].

An example of DNS Protocol Filtering is shown in Figure 5.

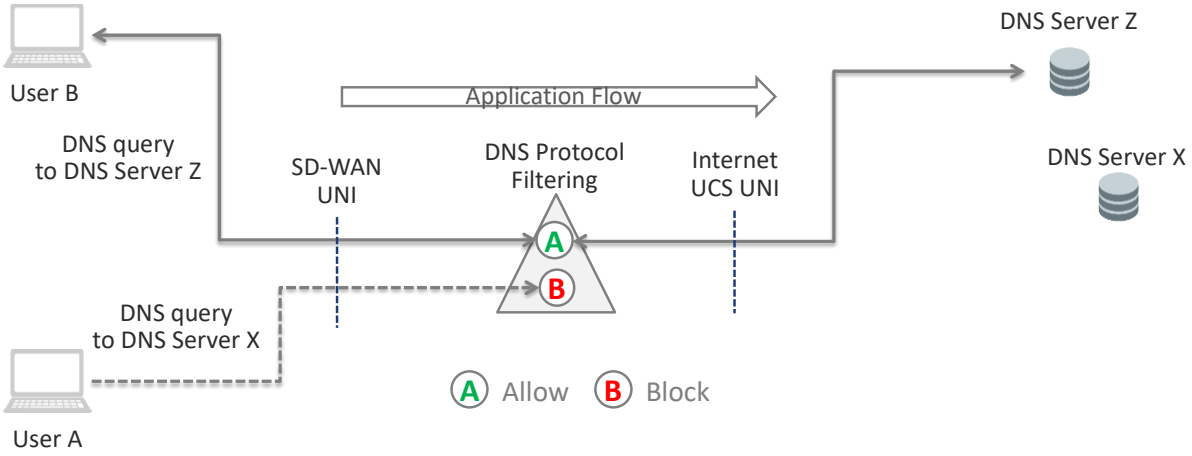


Figure 5 – Example of DNS Protocol Filtering

In Figure 5, DNS Server Z is on the DNS Protocol Filtering Allow List and DNS Server X is on the DNS Protocol Filtering Block List. When a DNS Query is sent to DNS Server X, it is Blocked by the DNS Protocol Filtering Security Function. Depending on prior agreement of the Service Provider and SD-WAN 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 match criteria entries used by the DNS Protocol Filtering Security Function to Block the subset of the Application 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 match criteria entries used by the DNS Protocol Filtering Security Function to Allow the subset of the Application 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 match criteria entries that are not on the DNS Protocol Filtering Block List but are deemed suspicious. The subset of the Application 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 lists are maintained by the Service Provider, with each match criteria entry on a list using a 5-tuple of the form <*SAV4, DAV4, SAV6, DAV6, DNS Message Type*>.

- *SAV4* is the list of IPv4 source addresses. *Any* can be used to indicate that any IPv4 source address is on the list. The list can be empty.
- *DAV4* is the list of IPv4 destination addresses. *Any* can be used to indicate that any IPv4 destination address is on the list. The list can be empty.
- *SAV6* is the list of IPv6 source addresses. *Any* can be used to indicate that any IPv6 source address is on the list. The list can be empty.
- *DAV6* is the list of IPv6 destination addresses. *Any* can be used to indicate that any IPv6 destination address is on the list. The list can be empty.
- *DNS Message Type* is one of the DNS messages, as specified in RFC 6895 [17]. Examples of a valid entry are DNS Query, DNS Response, DNS Update.

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.

[R53] When DNS Protocol Filtering is *Enabled* for a given Application 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.

[R54] When DNS Protocol Filtering is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the DNS Protocol Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R55] When DNS Protocol Filtering is *Enabled* for a given Application Flow, the DNS Protocol Filtering Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the DNS Protocol Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R56] When DNS Protocol Filtering is *Enabled* for a given Application 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 Application Flow that matches a match criteria entry on the DNS Protocol Filtering Allow List
- Allow the subset of the Application Flow that does not match a match criteria entry on any of the DNS Protocol Filtering lists, per the second bullet of [R55]
- Block the subset of the Application Flow that matches a match criteria entry on the DNS Protocol Filtering Block List
- Block the subset of the Application Flow that matches a match criteria entry on the DNS Protocol Filtering Quarantine List
- Block the subset of the Application Flow that does not match a match criteria entry on any of the DNS Protocol Filtering lists, per the first bullet of [R55]

[R57] 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.

[D4] When the DNS Protocol Filtering Security Function is *Enabled* for a given Application Flow, and when a DNS message in that Application Flow is Blocked, the DNS Protocol Filtering Security Function **SHOULD** send an appropriate DNS response code, per section 2.3 of RFC 6895 [17], 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.

There are two scenarios that can be used by the Service Provider to support DNS Protocol Filtering.

Scenario A: The Service Provider hosts a DNS server, and the Subscriber uses that DNS server for queries. This DNS server is included in the Allow List. In this case, when DNS Protocol Filtering is *Enabled*, encrypted (e.g., DNS over HTTPS and/or DNS over TLS) and unencrypted DNS queries are checked and either Allowed or Blocked, per the previous description.

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

There are special cases involving DNS that are not in scope for this document. Specifically,

- DNS over HTTPS (DoH), as specified in RFC 8484 [25]. DoH shares port 443 with all other HTTPS traffic. If a Subscriber wants to use DoH, it is better to have MBF *Disabled* for the Application Flow.
- DNS over TLS (DoT), as specified in RFC 7858 [21]. DoT uses dedicated port 853. It is possible to use the IP, Port and Protocol Filtering Security Function to either Allow/Block the subset of the Application Flow carrying DoT.
- DNSSEC, based on a suite of IETF RFCs, provides security extensions for DNS.
- TOR, as specified in RFC 7686 [20], is a special case that does not use the DNS infrastructure for resolving domains associated with the '.onion' top level domain.

9.3 Domain Name Filtering

Domain Name Filtering is defined as the Security Function that determines whether an Application Flow, or subset of an Application 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 Application 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 Application 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 match 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 lists are maintained by the Service Provider, with each match 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.

[R58] The Domain Name Filtering Security Function **MUST** be capable of using wildcard match criteria entries.

[R59] 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 four basic methods for identifying whether a subset of the Application Flow matches a match criteria entry on the Domain Name Filtering Block List, Domain Name Filtering Allow List,

or the Domain Name Filtering Quarantine List. All four methods may be used on a single Application Flow.

- Inspect the DNS messages, which are normally unencrypted, to identify the domain names.
- Inspect the subset of the Application Flow that is unencrypted to identify the domain names.
- For the subset of the Application 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 [15], inspect the Server Name Indication extension field to identify the domain names.
- If MBF is Enabled for the Application Flow and a subset of the Application Flow is encrypted, inspect the decrypted subset of the Application Flow to identify the domain names.

Note that the identification methods described in the first three bullets above can also be used when MBF 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 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 6 .

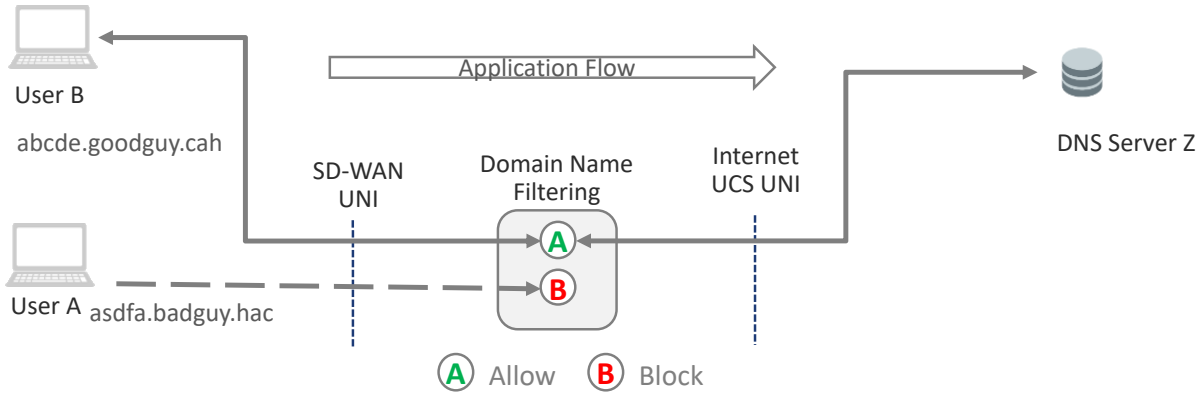


Figure 6 – Example of Domain Name Filtering using DNS Messages

As shown in Figure 6, users A and B send unencrypted request messages to DNS Server Z, 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 Application 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.

[R60] When Domain Name Filtering is *Enabled* for a given Application 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.

[R61] When Domain Name Filtering is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the Domain Name Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R62] When Domain Name Filtering is *Enabled* for a given Application Flow, the Domain Name Filtering Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the Domain Name Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R63] When Domain Name Filtering is *Enabled* for a given Application 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 Application Flow that matches a match criteria entry on the Domain Name Filtering Allow List
- Allow the subset of the Application Flow that does not match a match criteria entry on any of the Domain Name Filtering lists, per the second bullet of [R62]
- Block the subset of the Application Flow that matches a match criteria entry on the Domain Name Filtering Block List
- Block the subset of the Application Flow that matches a match criteria entry on the Domain Name Filtering Quarantine List
- Block the subset of the Application Flow that does not match a match criteria entry on any of the Domain Name Filtering lists, per the first bullet of [R62]

[R64] The Domain Name Filtering Security Function **MUST** be capable of informing the client in the Subscriber's network immediately when access to a domain is Blocked.

[D5] When access to a domain is Blocked, the client in the Subscriber's network **SHOULD** be immediately informed.

This document does not specify the method for informing the client, e.g., DNS client, HTTP client, etc. The method used is based on agreement of the Service Provider and Subscriber.

9.4 URL Filtering

URL Filtering is defined as the Security Function that determines whether an Application Flow, or subset of an Application Flow, contains a URL that is to be Allowed or Blocked. URL is specified in IETF RFC 3986 [11]. 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 Application 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 Application 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 lists are maintained by the Service Provider, with each match 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/*.

[R65] The URL Filtering Security Function **MUST** be capable of using wildcard match criteria entries.

[R66] 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 Application Flow matches a match criteria entry on one of the URL Filtering lists:

- If the Application Flow is unencrypted, inspect the Application Flow to identify the URL.
- If the Application Flow is encrypted, enable MBF for the Application Flow, and inspect the unencrypted packets to identify the URL.

An example of URL Filtering applied to a TLS session is shown in Figure 7 .

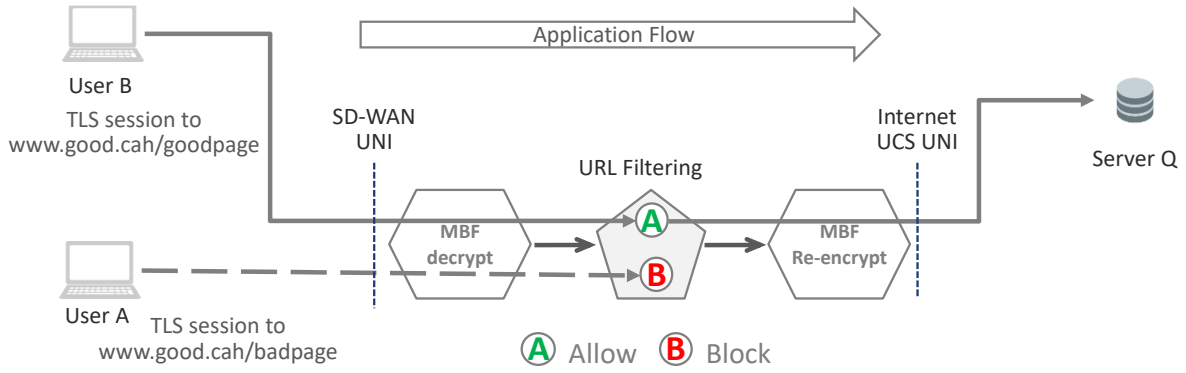


Figure 7 – Example of URL Filtering applied to TLS session

In the example shown in Figure 7, Server Q hosts several domains. URL Filtering is enabled for the Application Flow and since the Application Flow is encrypted, an MBF is also enabled. User A 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 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.

[R67] When URL Filtering is *Enabled* for a given Application 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.

[R68] When URL Filtering is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the URL Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R69] When URL Filtering is *Enabled* for a given Application Flow, the URL Filtering Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the URL Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R70] When URL Filtering is *Enabled* for a given Application 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 Application Flow that matches a match criteria entry that is on the URL Filtering Allow List
- Allow the subset of the Application Flow that does not match a match criteria entry on any of the URL Filtering lists, per the second bullet of [R69]
- Block the subset of the Application Flow that matches a match criteria entry on the URL Filtering Block List
- Block the subset of the Application Flow that matches a match criteria entry on the URL Filtering Quarantine List
- Block the subset of the Application Flow that does not match a match criteria entry on any of the URL Filtering lists, per the first bullet of [R69]

[R71] The URL Filtering Security Function **MUST** be capable of informing the client in the Subscriber's network immediately when access to a URL is Blocked.

[D6] When URL Filtering is *Enabled* for a given Application Flow, and when access to a URL is Blocked, the client in the Subscriber's network **SHOULD** be immediately informed.

This document does not specify the method for informing the client (e.g., the HTTP client). The method used is based on agreement of the Service Provider and Subscriber.

9.5 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 an Application Flow, or subset of an Application Flow, contains Malware, and removes the Malware or Blocks the subset of the Application 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 Application Flow is encrypted, Malware Detection and Removal can only be *Enabled* if an MBF is also *Enabled*.

A Malware Detection and Removal Block List is a list of match criteria entries used by the Malware Detection and Removal Security Function that Blocks the subset of the Application Flow that contains a match to one of the entries, i.e., Malware has been detected and the appropriate action taken (see [R76]). When this happens, the Subscriber is notified via the SEN. A Malware Detection and Removal Allow List is a list of match criteria entries used by the Malware Detection and Removal Security Function that Allows the subset of the Application Flow that contains a match to one of the entries. A Malware Detection and Removal Quarantine List is a list of match criteria entries that are deemed suspicious, but the match criteria entry has not been identified on the Malware Detection and Removal Block List. For a match criteria entry on the Domain Name Filtering Quarantine List, either the Malware is removed or the subset of the Application Flow containing the Malware is Blocked by the Malware Detection and Removal Security Function until further security checks can be done (see [R76]). When this happens, the Subscriber is notified via the SEN.

The lists are maintained by the Service Provider, with each match 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.

[R72] When Malware Detection and Removal is *Enabled* for a given Application 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.

In the rest of this section, the term Object is used to denote a part of an Application Flow that is scanned when Malware Detection and Removal is *Enabled*. Examples of an Object are a file

attached to an e-mail or downloaded from a web server; a link on a web page with embedded Malware; etc. An alternative term is artifact.

Malware Detection: All Objects are scanned, using known signatures, to detect Malware hidden in the Objects (e.g., antivirus scan on an e-mail). Objects without detected Malware are forwarded. Objects with detected Malware have an action applied. A decision is made to either Block the Object (Subscriber wants all such Objects Blocked) or to quarantine the Object (Subscriber wants the ability to inspect the quarantined Objects and decide whether to Allow them). In this latter case, the Subscriber takes the risk if they move an Object on the Quarantine List to the Allow List.

For a limited set of Objects determined by the Service Provider as potentially suspicious, behavioral analysis may be done. This consists of a simulated but monitored environment where the Objects are 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 on Objects for which no known signature exists. If a dangerous Object is detected, a signature is created.

If a Security Policy with Malware Detection and Removal *Enabled* is applied to an Application Flow in a given Zone using the MEF 70.1 [2] AF-SECURITY-INGRESS Policy Criterion, then all Application Flows in that Zone should have a Security Policy that has Malware Detection and Removal *Enabled* via the MEF 70.1 [2] AF-SECURITY-INGRESS Policy Criterion. Similarly, if a Security Policy with Malware Detection and Removal *Enabled* is applied to an Application Flow in a given Zone using the MEF 70.1 [2] AF-SECURITY-EGRESS Policy Criterion, then all Application Flows in that Zone should have a Security Policy that has Malware Detection and Removal *Enabled* via the MEF 70.1 [2] AF-SECURITY-EGRESS Policy Criterion.

[R73] When Malware Detection and Removal is *Enabled* for a given Application Flow, the Service Provider **MUST** describe which kind of detection (e.g., signature scan, behavioral analysis or both) is performed.

For a given Application 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 Malware Detection and Removal is *Enabled* for a given Application Flow, performance could be impacted.

If a subset of an Application Flow is encrypted and cannot be decrypted by the MBF, then Malware Detection and Removal cannot be performed for that subset of the Application 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 Application Flow that cannot be decrypted.

[R74] When Malware Detection and Removal is *Enabled* for a given Application Flow, the Service Provider **MUST** support both of the following actions for a subset of the Application Flow that does not match a match criteria entry on any of the Malware Detection and Removal Filtering lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R75] When Malware Detection and Removal is *Enabled* for a given Application Flow, the Malware Detection and Removal Security Function **MUST** perform one of the following actions for each subset of the Application Flow that does not match a match criteria entry on any of the Malware Detection and Removal lists:

- Block the subset of the Application Flow
- Allow the subset of the Application Flow

[R76] When Malware Detection and Removal is *Enabled* for a given Application Flow, and when an Object in the Application 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 Object and Block the associated subset of the Application Flow
- Block the Object and Allow the associated subset of the Application Flow
- Quarantine the Object and Block the associated subset of the Application Flow
- Quarantine the Object and Allow the associated subset of the Application Flow
- Remove Malware from the Object and Allow the associated subset of the Application Flow

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

Per [R16], a SEN is to be issued whenever Malware is removed.

[R77] The Service Provider **MUST** report which action is taken for detected Malware and make it available to the Subscriber via a SEN (see Section 7.2).

Appendix B describes four examples of Malware Detection and Removal.

10 References

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

[2] MEF 70.1, *SD-WAN Service Attributes and Service Framework*, November 2021

[3] ITU-T X.509, *Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks*, October 2016

[4] IETF RFC 1035, *Domain Names - Implementation and Specification*, by P. Mockapetris November 1987.

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

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

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

[8] 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.

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

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

[11] 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).

[12] 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).

[13] 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).

[14] 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).

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] IETF RFC 7970, *The Incident Object Description Exchange Format Version 2*, by Roman Danyliw, November 2016. Copyright © 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

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

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

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

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

[31] FIPS PUB 140-2, *Security Requirements for Cryptographic Modules*, May 2001, https://csrc.nist.gov/publications/detail/fips/140/2/final

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

[33] 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

[34] 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

[35] NIST, *National Vulnerability Database, Common Weakness Enumeration*, https://nvd.nist.gov/vuln/categories

[36] 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/

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

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

[39] MITRE, *CAPEC, Common Attack Pattern Enumeration and Classification*, https://capec.mitre.org

[40] STIX, *Structured Threat Information Expression*, https://oasis-open.github.io/cti-documentation/

Appendix A Application Flows and Security Functions (Informative)

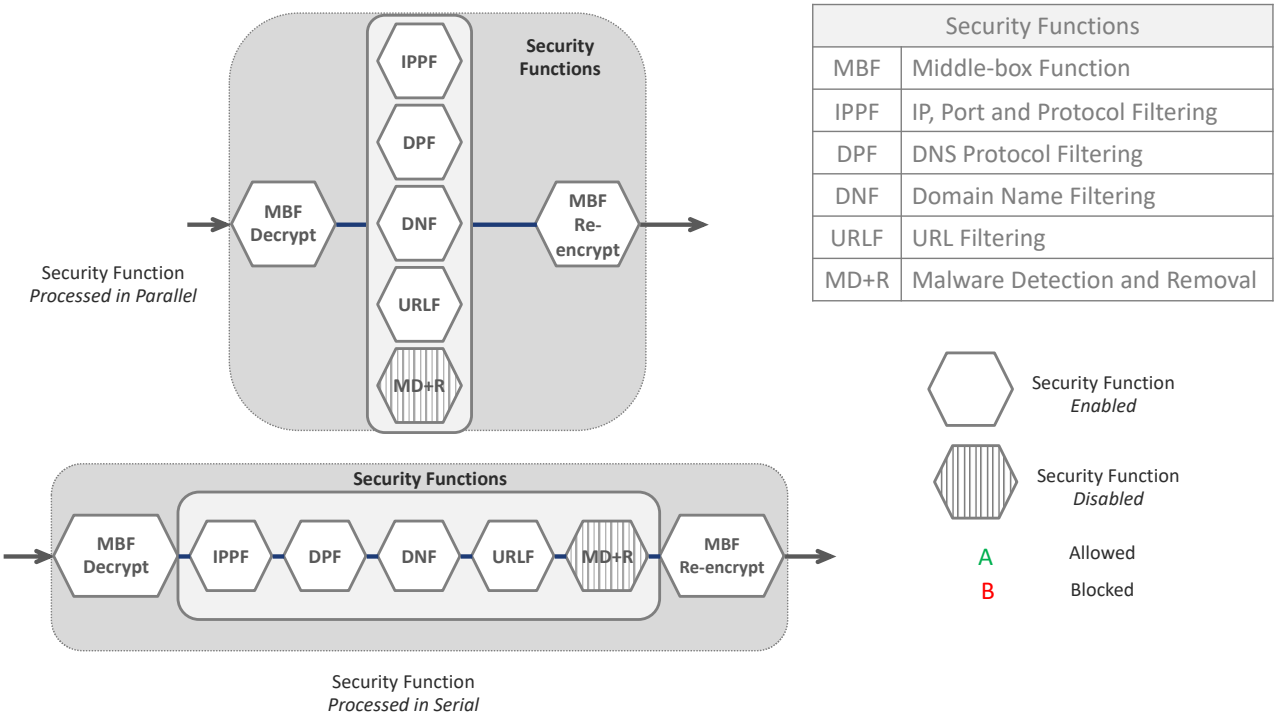
This Appendix describes several use cases of Application Flows, with one or more Security Functions enabled. The use cases involve unencrypted Application Flows, encrypted Application Flows and an Application Flow with a mix of traffic.

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

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

- MBF – Middle-Box Function
- IPPF – IP, Port and Protocol Filtering
- DPF – DNS Protocol Filtering
- DNF – Domain Name Filtering
- URLF – URL Filtering
- MD+R – Malware Detection and Removal
- A – Allowed
- B – Blocked

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 8 .



Security Functions	
MBF	Middle-box Function
IPPF	IP, Port and Protocol Filtering
DPF	DNS Protocol Filtering
DNF	Domain Name Filtering
URLF	URL Filtering
MD+R	Malware Detection and Removal

Figure 8 – Legend of Security Functions

This document does not constrain how these Security Functions are processed. 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 and allow for a multiple Service Provider solution. It requires multiple security applications. A challenge of this approach is the integration of the multiple Service Providers.

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 an SD-WAN 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 Application Flow is used in the following use cases, it could apply to either an Ingress Application Flow or an Egress Application Flow.

A.1 Use Case 1: TLS 1.2 Application Flow

The Subscriber is running a Business Office application over a TLS 1.2 encrypted session. All Security Functions are enabled.

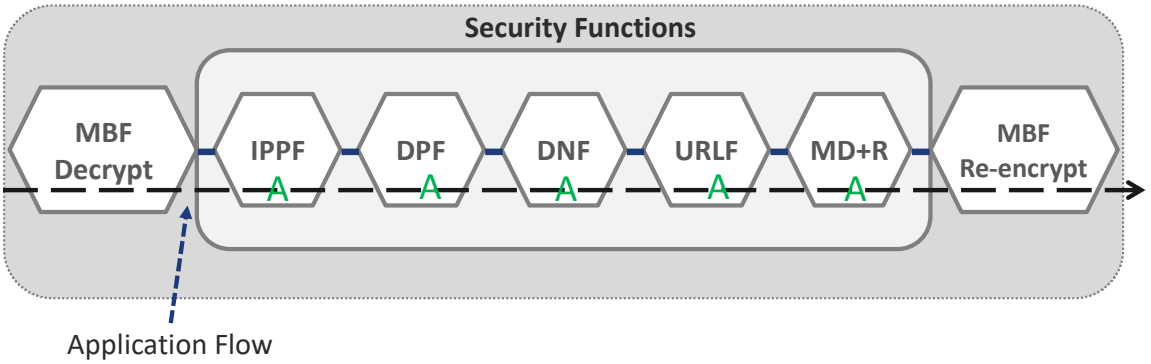


Figure 9 – Use Case 1: Business Office, TLS 1.2 Application Flow

In Use Case 1, MBF decrypts and re-encrypts the Application Flow and the individual Security Functions check the Application Flow. All checks are good, and the Application Flow is Allowed.

A.2 Use Case 2: Business Office, TLS 1.2 Application Flow, Malware is Detected

The Subscriber is running a Business Office application over a TLS 1.2 encrypted session. All Security Functions are enabled. E-mail Messages are subject to Malware Detection. Malware is detected and removed from a subset of the Application Flow.

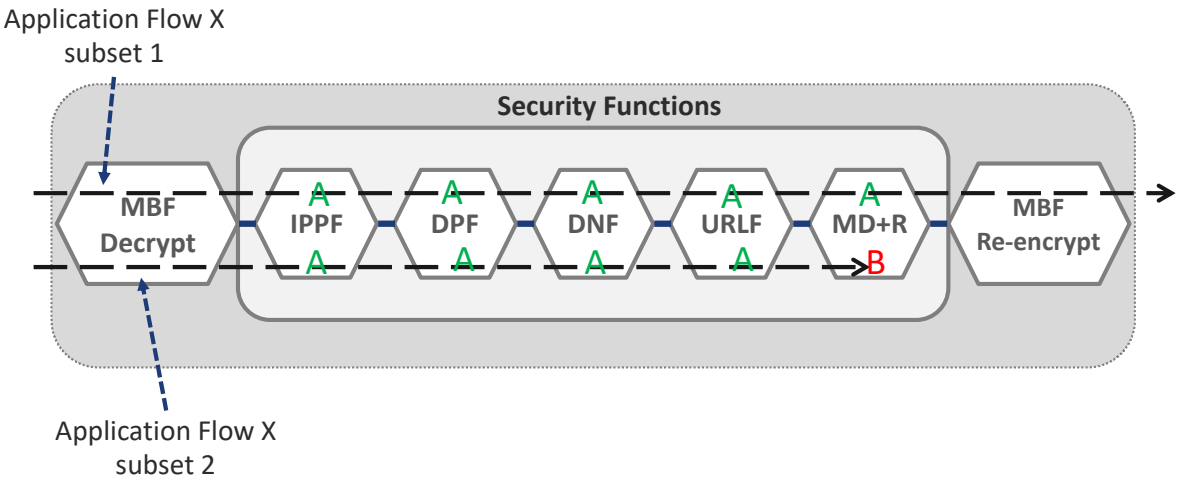


Figure 10 – Use Case 2: TLS 1.2 Application Flow, Malware is Detected

In Use Case 2, MBF decrypts and re-encrypts the Application Flow and the individual Security Functions check the Application Flow. The Malware Detection and Removal Security Function detects Malware in a message. The Malware is removed, and the rest of the Application Flow is Allowed.

A.3 Use Case 3: Cloud Drive, QUIC Application Flow

In Use Case 3, the Subscriber is running a Cloud Drive application over a QUIC encrypted session. The IP, Port and Protocol Filtering (IPPF) and DNS Protocol Filtering (DPF) Security Functions are enabled; the other Security Functions are disabled.

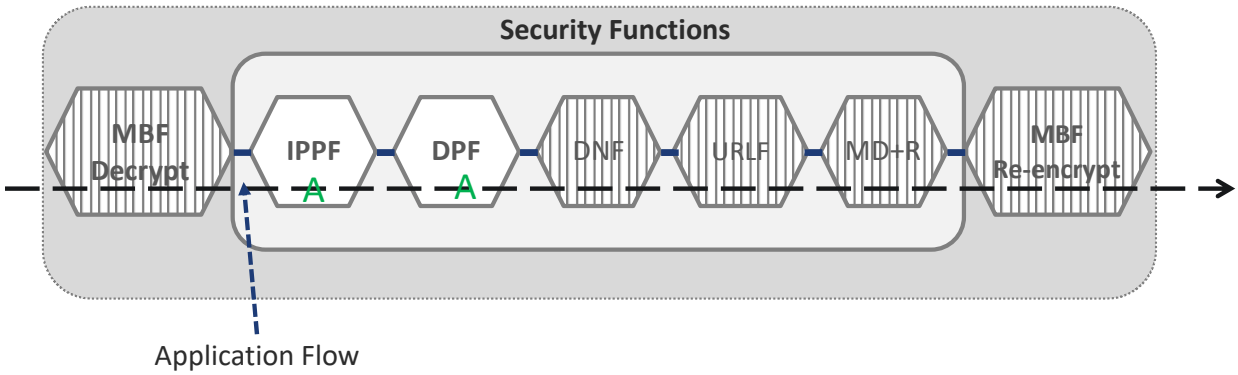


Figure 11 – Use Case 3: Cloud Drive, QUIC Application Flow

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

A.4 Use Case 4: Web Traffic to Public Resource

In Use Case 4, the Subscriber has an Application Flow supporting the Guest Wi-Fi zone providing access to the Internet via Internet Breakout. The IP, Port and Protocol Filtering (IPPF), DNS Protocol Filtering (DPF) and Domain Name Filtering (DNF) Security Functions are enabled; the other Security Functions are disabled. On this Application Flow, HTTP clients are trying to access two different domains. Subset 1 of the Application Flow is trying to connect to site.qaz.com web server. Subset 2 of the Application Flow is trying to connect to badresource.badcompany.org web server.

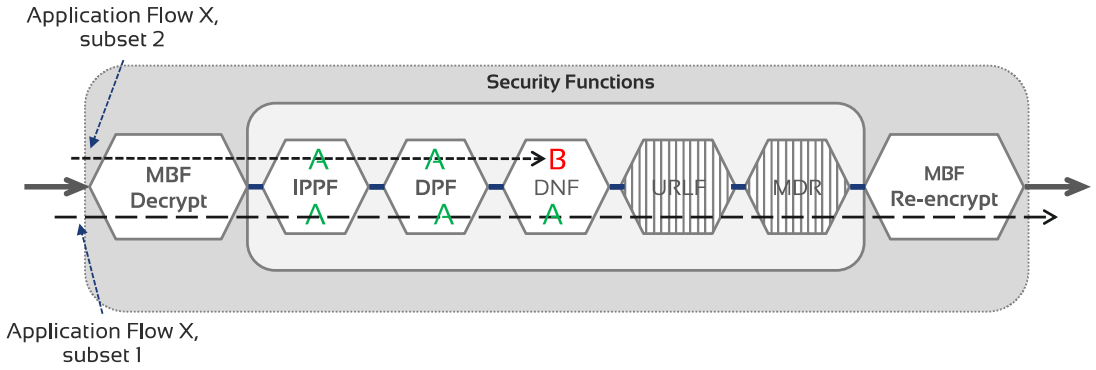


Figure 12 – Use Case 4: Web Traffic to a Public Resource

In this case, for subset 1, all Security Function checks are good, and subset 1 is Allowed. For subset 2, the IPPF and DPF checks are good, but since the badresource.badcompany.org domain name is on the Block List, subset 2 is Blocked.

A.5 Use Case 5: File Transfer Application

In Use Case 5, the Subscriber is using the Application Flow for file transfer. File uploads consist of Secure Shell (SSH) and File Transfer Protocol (FTP). By policy, FTP is not Allowed (on the Block List). The IP, Port and Protocol Filtering (IPPF) and DNS Protocol Filtering (DPF) Security Functions are enabled; the other Security Functions are disabled. Subset 1 of the Application Flow uses SSH, while subset 2 of the Application Flow uses FTP.

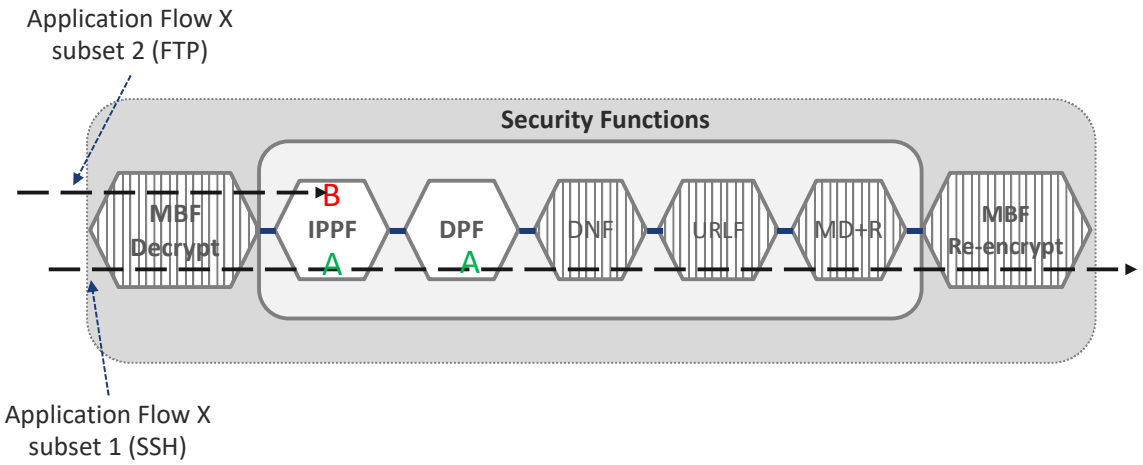


Figure 13 – Use Case 5: File Transfer, Application Flow is a mix of Encrypted and Unencrypted

In this case, DPF checks are good, but IPPF checks identify a file upload using FTP, which is on the Block List, and is therefore Blocked. The SSH file uploads are Allowed.

A.6 Use Case 6: Web Traffic to a Weather Site, Unencrypted Application Flow

In Use Case 6, the Subscriber is accessing a website to check weather. The IP, Port and Protocol Filtering (IPPF) and DNS Protocol Filtering (DPF) Security Functions are enabled; the other Security Functions are disabled.

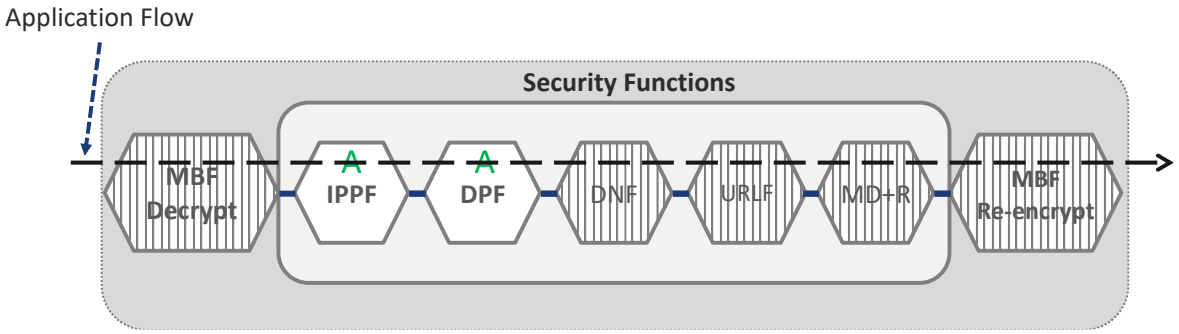


Figure 14 – Use Case 6: Web Traffic to a Weather Site, Unencrypted Application Flow

In this case, IPPF and DPF checks are good. The Application Flow is Allowed.

A.7 Use Case 7: Web Traffic to Public Resource, Unencrypted Application Flow

In Use Case 7, the Subscriber has an Application Flow supporting the Guest Wi-Fi zone providing access to the Internet via Internet Breakout. The IP, Port and Protocol Filtering (IPPF) and DNS Protocol Filtering (DPF) Security Functions are enabled; the other Security Functions are disabled. The Application Flow consists of DNS messages to two different DNS servers. Subset 1 of the Application Flow is sending DNS messages to DNS Server 1, which is on the Allow List. Subset 2 of the Application Flow is sending DNS messages to DNS Server 2, which is on the Block List.

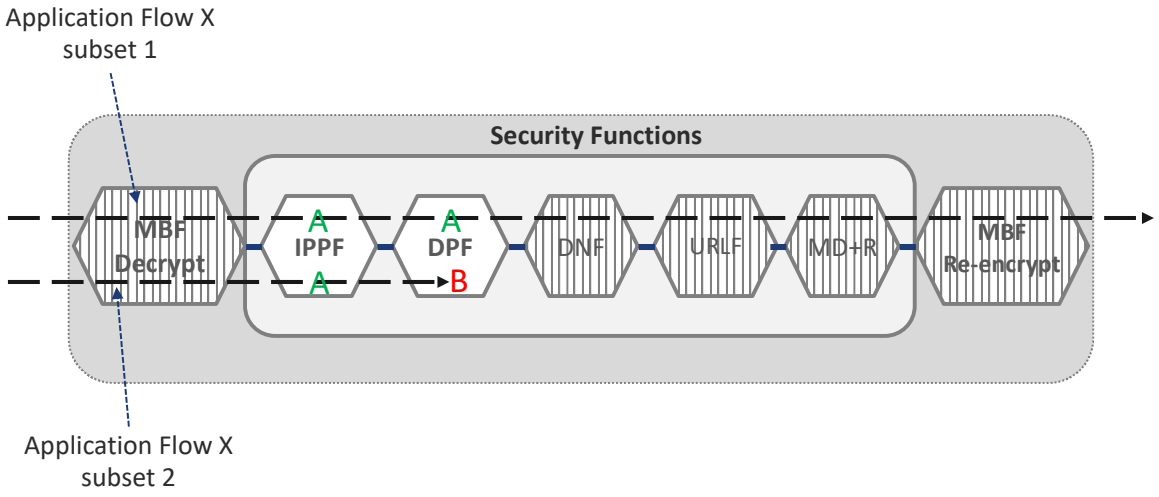


Figure 15 – Use Case 7: Web Traffic to Public Resource, Unencrypted Application Flow

In this case, DPF detects DNS messages to DNS Server 2 on the Block List and Blocks those messages. The DNS messages to the other DNS Server 1 are Allowed.

A.8 Use Case 8: Commerce Website

In Use Case 8, the Subscriber is running applications to a Commerce website. The Application Flow consists of a mix of unencrypted and encrypted traffic. All Security Functions are enabled. By policy, unencrypted traffic is not Allowed to this commerce website.

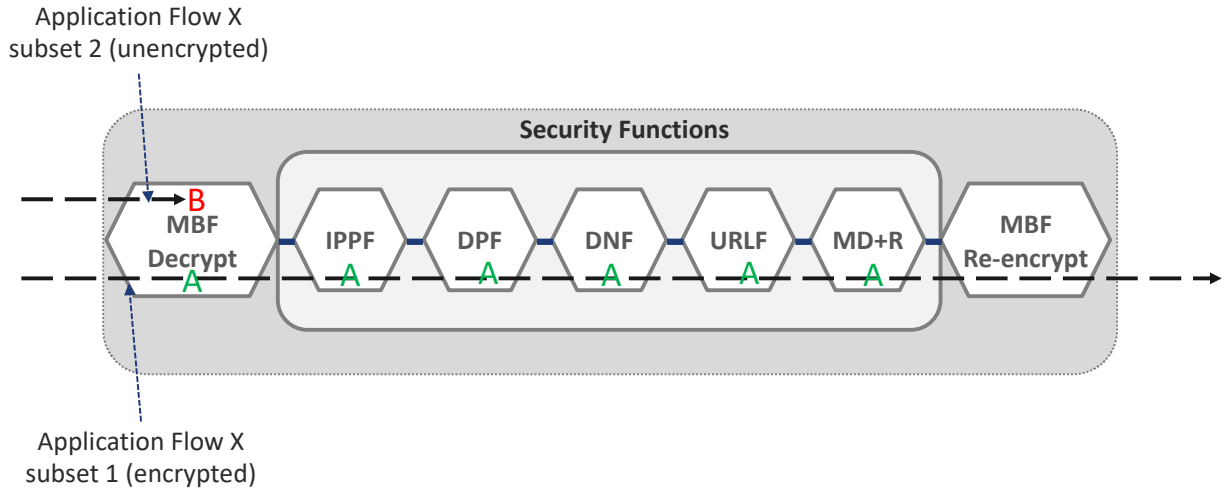


Figure 16 – Use Case 8: Commerce Website, mix of Unencrypted and Encrypted Traffic

In this case, all Security Functions are enabled. Subset 1 of the Application Flow is encrypted, and all Security Function checks are good. Subset 1 is Allowed. Subset 2 of the Application Flow is unencrypted. MBF is configured to Block unencrypted traffic and so subset 2 is Blocked.

A.9 Use Case 9: Internet Website, Unencrypted Traffic

In Use Case 9, the Subscriber is running an unencrypted application to an Internet website. All Security Functions are enabled. By policy, unencrypted traffic is Allowed on this Application Flow.

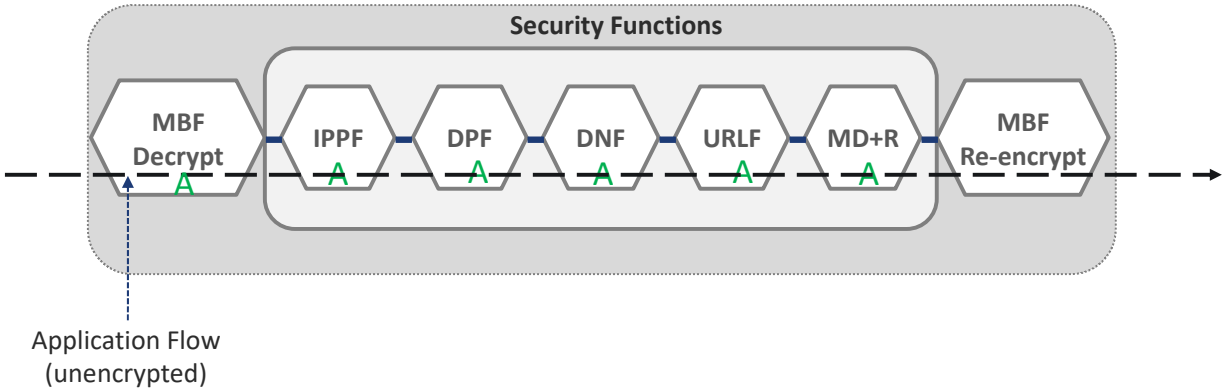


Figure 17 – Use Case 9: Internet Website, Unencrypted Traffic

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

A.10 Use Case 10: Social Media Website, URL Filtering

In Use Case 10, the Subscriber is running applications to a social media website. All Security Functions are enabled. Subset 1 of the Application Flow consists of requests to site.qaz.com/user-site, while subset 2 of the Application Flow consists of requests to site.qaz.com/advertisement/adultcontent. By policy, access to URLs containing adult content is Blocked.

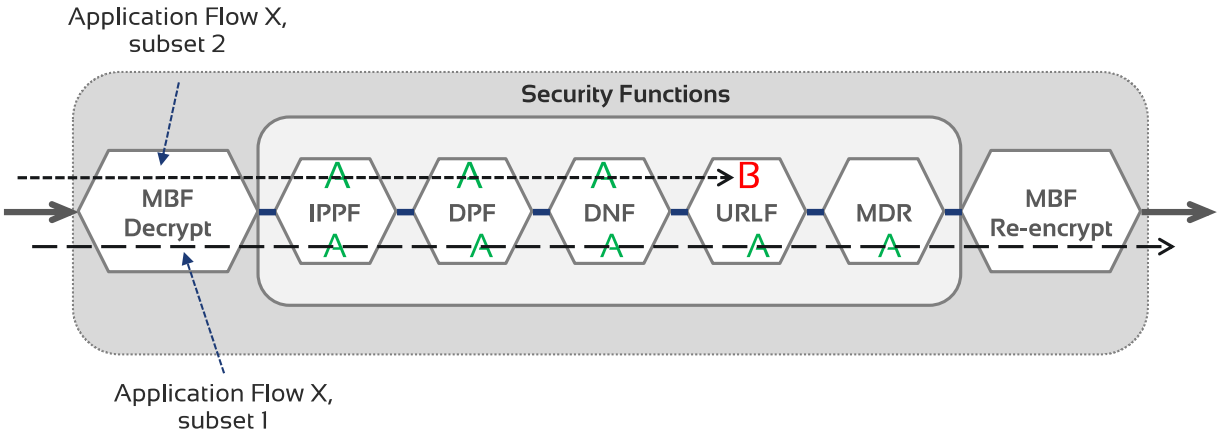


Figure 18 – Use Case 10: Internet Website, Unencrypted traffic

All Security Function checks for subset 1 are good, and that subset is Allowed. The URLF Security Function detects that subset 2 has a URL that is on the Block List, and therefore subset 2 is Blocked.

Appendix B Examples of Malware Detection (Informative)

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

Figure 19 shows an example of a clean file going through the Malware Detection and Removal process.

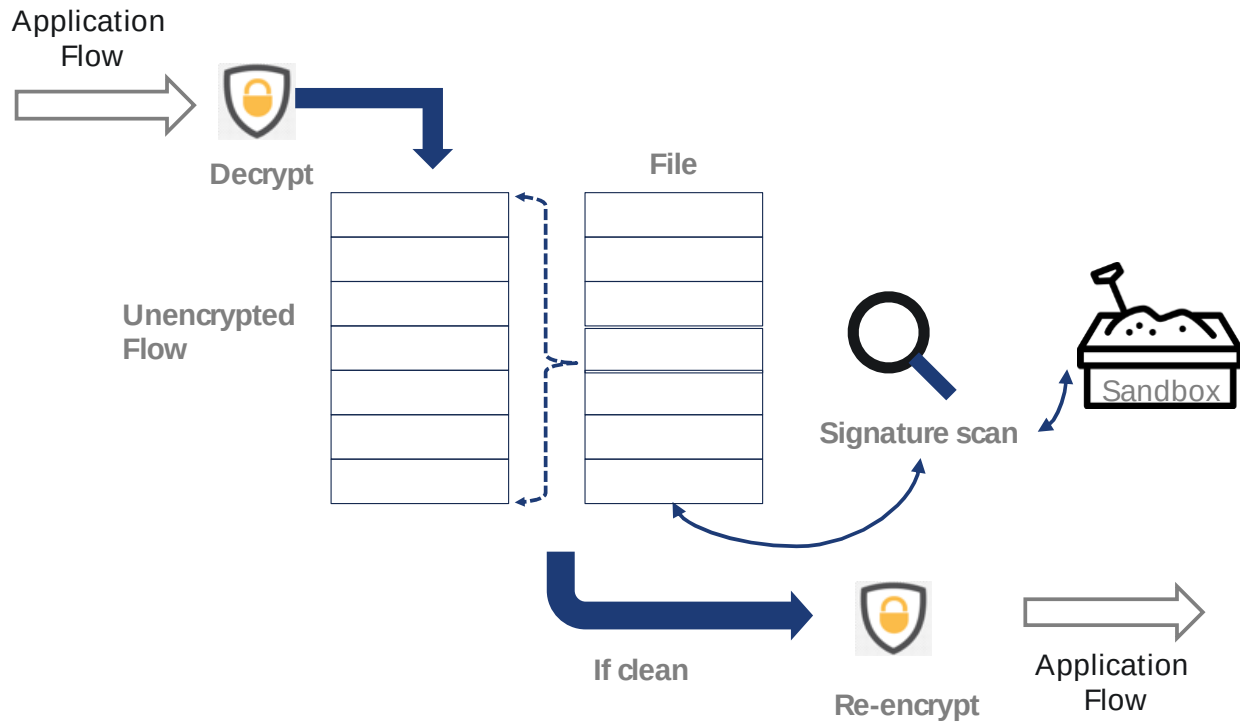


Figure 19 – Example of a Clean File

In the example shown in Figure 19, 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 Application Flow is re-encrypted and forwarded.

Figure 20 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.

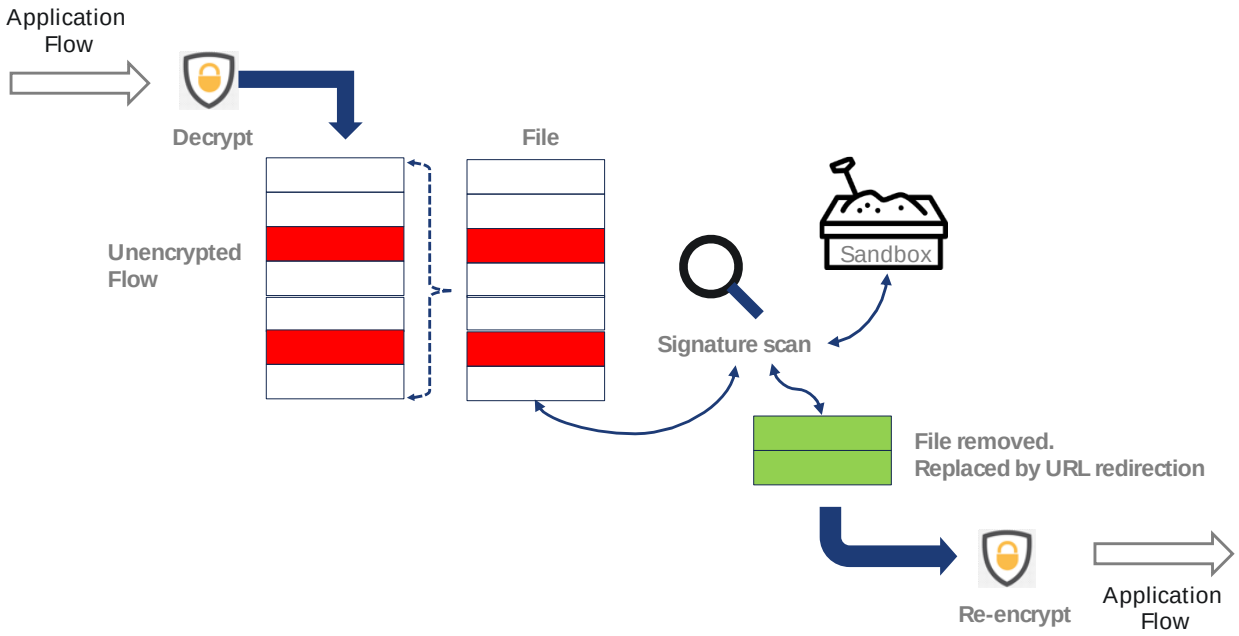


Figure 20 – Example of Malware Detected and File Removed using a Signature Scan

In the example shown in Figure 20, 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 21 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 [9], is used for transferring data into and back from the sandbox.

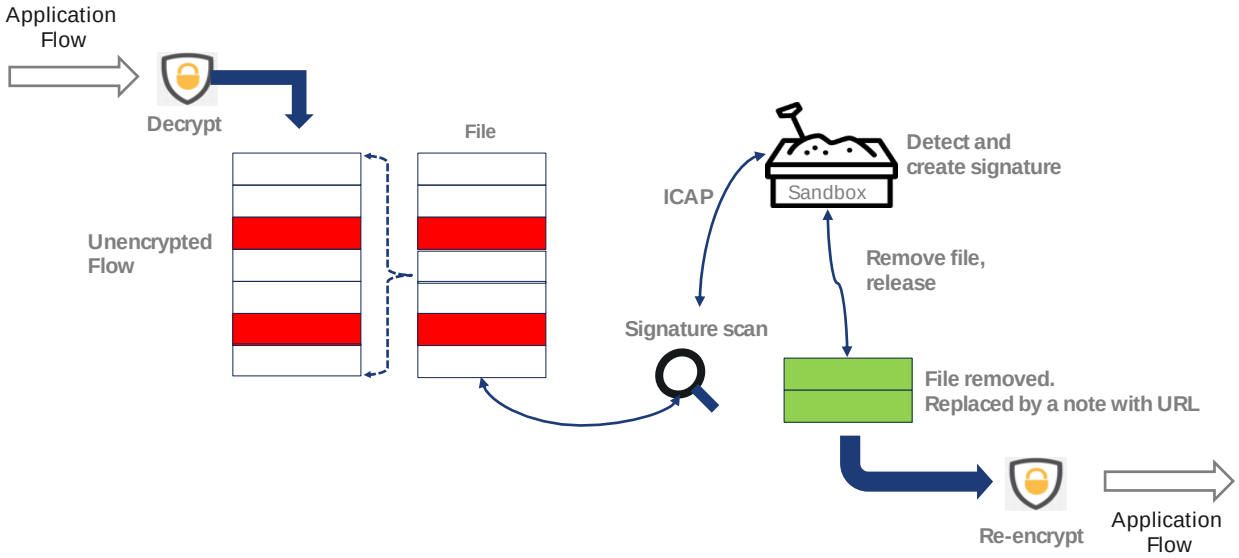


Figure 21 – Example of Malware Detected and File Removed using a Sandbox

In the example shown in Figure 21, 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 22 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.

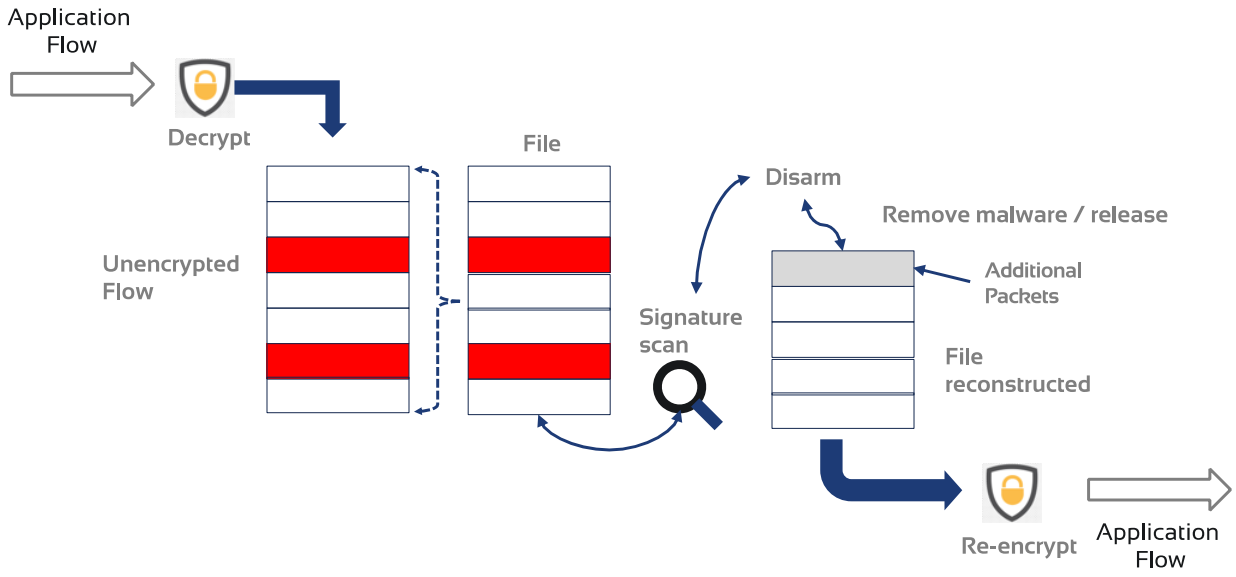


Figure 22 – Example of Malware Detected and File Reconstructed

In 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 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 C Threat Modeling (Informative)

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 an MBF to enable security policy enforcement, the likelihood of the above categories of threat above can be critically affected by how the MBF 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, MBF 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 MBF 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: [R35], [R36], [R37], [R38], [R39], [R40], [R41], [R42], [R43], [R44], [R45], [D2], [D3], [R46], [R47] and [R48] deals with transport security as it relates to the MBF.

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 application flow itself, particularly if the application places additional trust on the flow because of the presence of the MBF.
- Mitigation: [R39] 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 application flow itself, particularly if the application places additional trust on the flow because of the presence of the MBF. Review authorization settings.
- Mitigation: [R39] 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 Application Flow itself. Consider using a standard authentication mechanism to identify the source data store.
- Mitigation: [R39] 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 [R16], [R17] and [R18].

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 Application Flows are covered by [D1].

The threat model is available at this link:

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

Appendix D Example of a Security Policy (Informative)

This Appendix describes an example of a Security Policy, where each of the parameter values are specified, i.e., not *None*. The value of the Security Policy, when included in the SWVC List of Security Policies Service Attribute defined in MEF 70.1 [2] is shown below and expanded on in Table 3. In both cases, angle brackets (<>) are used to denote tuples, square brackets ([]) are used to denote lists, and braces ({}) are used to denote sets.

```
<Blue,
 <IT Security List, UTC-5>,
 <[<1.2, {0xC0,0x2C, 0xC0,0x30, 0x00,0xA8}>],
 [<1.3, {Any}>],
 [<1.0, {Any}>,
  <1.1, {Any}>],
 [<1.2, {0xC0,0x2C, 0xC0,0x30, 0x00,0xA}>,
  <1.3, {Any}>],
 Block,
 [TRUST-A, TRUST-B],
 Block>,
 <[<Any, Any, UDP, Any, Any, UDP, Any, 443>],
 [<Any, Any, TCP, Any, Any, TCP, Any, 443>],
 [],
 Allow,
 4 hours>,
 <[< Any, Any, Any, Any, DNS Query>],
 [<Any, Any, Any, Any, DNS Response>],
 [],
 Allow,
 4 hours>,
 <[*.domain.tld],
 [],
 [],
 Allow,
 4 hours>,
 <[host.domain.tld/section/*],
 [],
 [],
 Allow,
 4 hours>,
 <[‘ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c’],
 [],
 [],
 Allow,
 4 hours,
 Signature,
 Remove Malware from the Object and Allow the associated subset of the
                              Application Flow>
>
```

Security Policy Parameter	Parameter Element	Parameter Element Value
ID	Security Policy Identifier	*Blue*
SEN	*R* (Recipient list)	*IT Security List*
	T (Time format)	*UTC-5*
MBF	*S* (MBF Supported List)	*[<1.2, {0xC0,0x2C, 0xC0,0x30, 0x00,0xA8}[2]>}*
	U (MBF Unsupported List)	*[<1.3, {Any}>]*
	B (MBF Block List)	*[<1.0, {Any}>, <1.1, {Any}>]*
	A (MBF 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*
IPPF	*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*
DPF	*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*
DNF	*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*

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

Security Policy Parameter	Parameter Element	Parameter Element Value
URLF	*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*
MD+R	*B* (MDR Block List)	[‘ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c’[3]]
	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 Application Flow*

Table 3 – Example of a Security Policy

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