DOTS
INTERNET-DRAFT
Intended Status: Standard Track
Expires: December 25, 2016

J. Francois
Inria
A. Lahmadi
University of Lorraine - LORIA
Marco Davids
SIDN Labs
Giovane C. M. Moura
SIDN Labs

June 23, 2016

IPv6 DOTS Signal Option
draft-francois-dots-ipv6-signal-option-00

Abstract

This document specifies an optional fall-back opportunistic method that employs the IPv6 Hop-by-Hop options extension header type. It allows a DOTS client to send a signaling message over a congested network due to a DDoS attack by ''tagging'' bypassing outgoing IPv6 packets to reach a DOTS server or relay.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright and License Notice

Table of Contents

1 Introduction

1.1 Overview

A distributed denial-of-service (DDoS) attack aims at rendering machines or network resources unavailable. These attacks have grown in frequency, intensity and target diversity [I-D.draft-ietf-dots-requirements]. Moreover, several protocols have been utilized to amplify the intensity of the attacks [kuhrer2014exit], peaking at several hundred gigabits per second.

The DOTS aims at defining a common and open protocol to signal DDoS attacks to facilitate a coordinated response to these attacks. This document specifies a signalling mechanism that instead of designing a new application-layer protocol, it utilizes the IPv6 Hop-by-Hop header [RFC2460]. This header has the advantage to be fully inspected by all network devices and it is the first header in IPv6 extension headers [RFC7045].

The new option containing the attributes of the signalling message is included in an opportunistic way in available IPv6 packets leaving a network element until the message reaches a DOTS server. It thus constitutes an additional signalling channel but MUST NOT replace the original signalling channel used between DOTS client and servers as the one defined in [I-D.draft-reddy-dots-transport]. The DOTS client will thus embed the signalling attributes into outgoing IPv6 packets not necessarily going to the DOTS server. Intermediate routers receiving such a packet will examine it and embed the same information into other IPv6 packets. domain in this opportunistic way to increase the probability that such a packet will be finally forwarded to a DOTS Relay or Server, but also in controlled way to avoid that the mechanism is exploited for a malicious purposes.

Only the Hop-by-Hop options header allows such behavior and using Destination options header is not enough to make the DOTS signal going through the network in an opportunistic way. Each network element recognizing this new option will select the best fitted IPv6 packets to deliver the signal to the DOTS server or relay. For this reason the Hop-by-Hop header option is essential to make such behavior compared to other existing IPv6 extension headers [RFC6564].

1.2 Motivation

The traffic generated by a DDoS can be characterized according to various parameters, such as the layer (IP/ICMP or application), maximum and instant throughput, among others. Regardless its nature, we assume that for most cases, a DOTS client will be able to signal back one or few messages, during the attack, to the DOTS phase.

We have the same behavior in other DDoS attacks. For instance, on November 30th and December 1st, 2015, the Root DNS system was hit by an application layer (DNS) attack [rootops-ddos]. Each one of the 13 root server letters (A--M) was hit by attacks peaking at 5 million queries per second. By utilizing the RIPE Atlas DNSMON infrastructure, we can see that during the DDoS attacks, most of the root server letters remained reachable and able to respond to the DNS request sent by the probes employed by the DNSMON [ripe-dnsmon-ddos]. Few letters, however, had a packet loss rate of more than 99%. The DNSMON probes, however, experience mostly delays in their DNS requests instead.

Our signalling mechanism operates in an opportunistic way it is designed for DDoS as the ones on the Root DNS system.

1.3 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The terms DOTS client, DOTS server, DOTS relay, DOTS agents, signal channel, DOTS signal and DOTS signal refers to the terminology introduced in [I-D.draft-ietf-dots-requirements].

The following terms are introduced:

Opportunistic DOTS signal:
an IPv6 packet containing the signalling attributes of an attack within the Hop-by-Hop extension header. The purpose is the same as the DOTS signal. It is used to request help for mitigating the attack.

DOTS opportunistic-capable router:
a router with the capacity to decode the opportunistic DOTS signal and re-embed such an information in other IPv6 packets.

All DOTS opportunistic-capable agents are defined as the DOTS agents supporting the opportunistic DOTS signal processing.

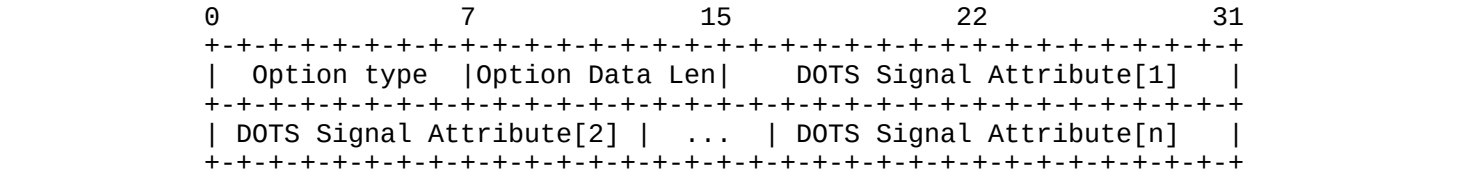
2. Opportunistic DOTS signal option

The goal is to provide an efficient mechanism where nodes in a IPv6 network facing a DDoS attack can deliver a DOTS signal message sent by a DOTS client to the DOTS server. The specified mechanism does not generate transport packets to carry the DOST signal message but it only relies on existing IPv6 packets in the

network to include inside them a hop-by-hop extension header which contains an encoded DOTS signal message. The solution defines a new IPv6 Hop-by-Hop header option with the semantic that the network node SHOULD include the option content within one or multiple outgoing IPv6 packets available in that network node.

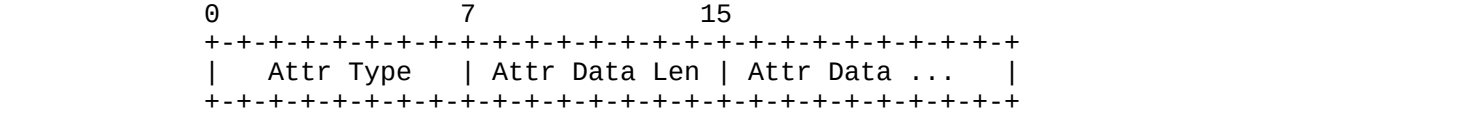
2.1 Hop-by-Hop option encoding

According to [RFC2460], options encoded into the IPv6 Hop-by-Hop header are formatted as Type-Length-Values (TLVs). The option for opportunistic DOTS signal is thus defined as follows:



The first byte defines the Hop-by-Hop Option type number allocated to the DOTS opportunistic signalling. This number is not yet fixed but the first three bits MUST be set to 0. The first two zero bits indicate that routers which cannot handle the DOTS signal option will continue to process other options. The third 0 bit means that the option processing will not change the packet’s final destination [RFC2460].

The second byte contains the length of the option content. The content of the DOTS Signal option is a variable-length field that contains one or more type-length-values (TLV) encoded DOTS signal attributes, and has the following format:



The Attr Type is 8-bit identifier of a DOTS signal attribute.

The Attr Data Len is 8-bit unsigned integer which is the length of Attr Data in bytes.

The Attr Data is variable-length field that contains the data of the attribute.

2.2 DOTS signal Option attributes

The first attribute embedded into the opportunistic DOTS signal is a TTL (Time-to-Live) field which indicates the maximum number of retransmission of the signal into another IPv6 packets until it MUST be discarded. Remaining attributes are similar to the header fields described in [I-D.draft-reddy-dots-transport] (section 5.1.1) used to convey a DOTS signal through a HTTP POST.

The sequence of attributes to be inserted within the header MUST start with fixed-length attributes which are defined in the following order:

TTL: Time-to-Live. This is a mandatory attribute encoded in one byte.

Flags: one byte is reserved for flags.
The first bit indicates the type of the IP address of the host: 0 for IPv4, 1 for IPv6. The second bit indicate if the protocol to use is TCP (1) or UDP (0). The third bit indicates if the message is signed The remaining bit are not used yet.

host: the IP address of the DOTS server where the signal option SHOULD be delivered. Depending on the flags, this field is encoded in 4 or 16 bytes.

port: the listening port of the DOTS server.
It is encoded in 2 bytes.

The remaining attributes MUST be TLV encoded, and they are defined in the following order:

policy-id: defined in [I-D.draft-reddy-dots-transport].

target-ip: defined in [I-D.draft-reddy-dots-transport].
However, each address or prefix is encoded in its own TLV element. The distinction between IPv4 and IPv6 is done over the length of the value.

target-port: defined in [I-D.draft-reddy-dots-transport].
However, each target port is encoded in its own TLV element.

target-protocol: defined in [I-D.draft-reddy-dots-transport].
However each target protocol is encoded in its own TLV element.

lifetime (lt): lifetime of the mitigation request defined in [I-D.draft-reddy-dots-transport].

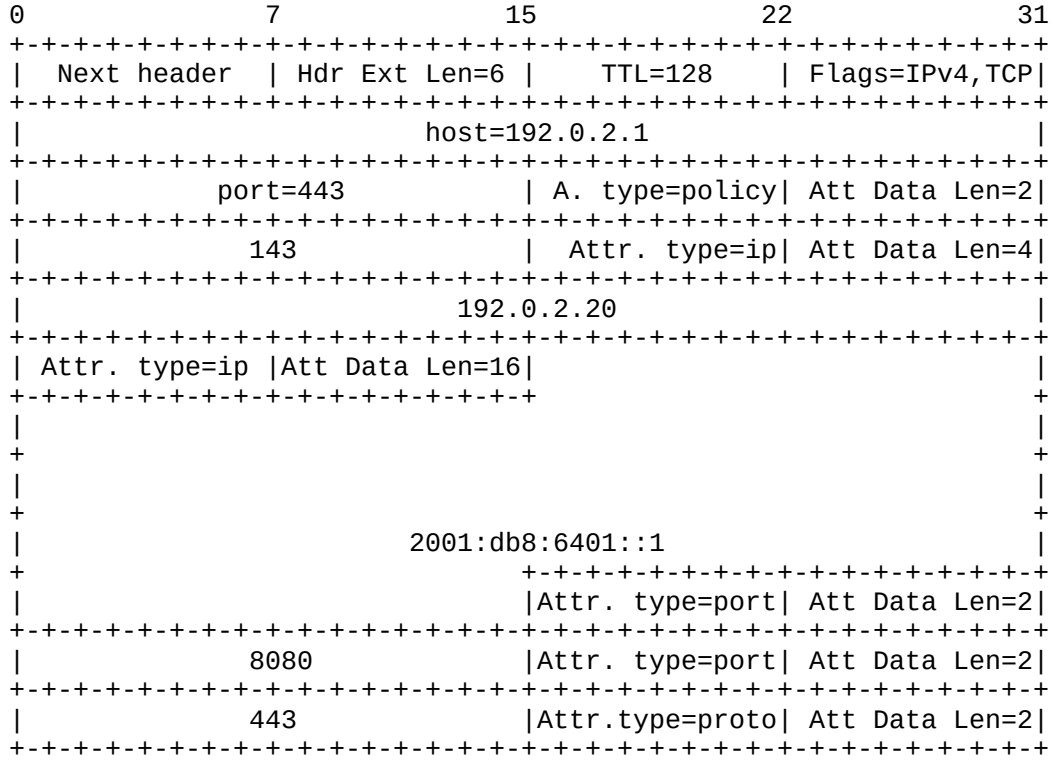
The encoded attributes MUST be included in the option header in the order defined above.

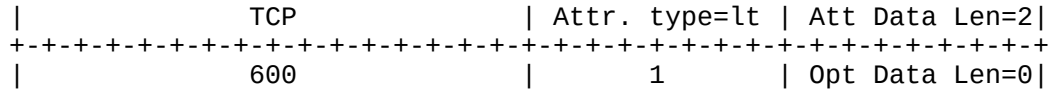
The following table provides the value of types that are used by the TLV encoded attributes.

Attribute type	value
policy-id	0
target-ip	1
target-port	2
target-protocol	3
lifetime	4

2.3 Example

Following is an example of an encoded Hop-by-Hop Option header to signal that a web service is under attack.





In the previous example, the message is not signed and terminates with padding. If it is the case, then the signature MUST BE added at the end such that the integrity and authenticity can be checked by the DOTS server or relay. The TTL attributes MUST be excluded from the signature calculation (see section 6).

3 Option Processing

3.2 Opportunistic DOTS signal initialization by a DOTS client

When a DOTS client needs to inform the DOTS server that it is under attack, it firstly makes a connection attempt and applies the mechanisms described in [I-D.draft-reddy-dots-transport].

In addition, it MAY activates an opportunistic mechanism to include the Hop-by-Hop header option specified in this document in one or multiple available IPv6 packets leaving the node.

The selection of packets has to be configured a priori. The configuration is composed of a sequence of rules defined in a hierarchical order such that they are triggered in a sequential manner.

Each rule is defined by:

- o a set of filters over the IPv6 packet headers. Only packets matching those filters are selected for opportunistic signalling. For instance, only packets heading to a given subnetwork or to specific address close to a DOTS server can be selected to increase the chance to reach the latter.

- o a ratio to select only a proportion of packets matching the filters in order to limit the induced overhead of the opportunistic signalling.

- o a timeout until the rule is active and selected IPv6 packets embed the DOTS opportunistic signal.

The objective is to apply each ordered rule after another according to their timeouts. The first rule is triggered immediately after the opportunistic signalling is activated.

Although the definition of rules MUST be configured by the user. It is RECOMMENDED to order them inversely related to the number of packets that would be selected. This can be approximated regarding the definition of filters. The core idea is to benefit from the first instants of the attack before losing connectivity by using a maximum number of outgoing packets to include the DOTS signalling option. It is thus RECOMMENDED to define the first as matching all IPv6 packets with a ratio equals one to rapidly disseminate the information but with a short timeout to limit the implied overhead.

Here is the an example of rules:

1: all outgoing IPv6 packets with a 10 second timeout
2: all outgoing IPv6 packets with a ratio of 10% and a 1 minute timeout
3: all outgoing multicast IPv6 packets with a ratio of 10% and a 1 minute timeout
4: all outgoing anycast IPv6 packets with a ratio of 10% and a 5 minute timeout
5: all outgoing IPv6 packets heading to the DOTS server with a ratio of 100% and a one hour timeout

3.2 Processing by a non DOTS opportunistic-capable router

When receiving an opportunistic DOTS signal encoded in a IPv6 packet, a non DOT opportunistic capable router simply skips the Hop-by-Hop option and continue the normal processing of the IPv6 packet because the option type MUST start with three zero bits.

3.3 Processing by a DOTS opportunistic-capable router

A DOTS opportunistic-capable router MUST store DOTS signalling information whose it is aware of. If a router processes an IPv6 DOTS opportunistic signal and supports this option, it first checks if it has already stored the associated information. In that case, the router simply skips the option and continues the normal processing otherwise it stores the encoded information in order to embed it again in other IPv6 packets similarly to the DOTS client. Hence, a set of rules are also defined in advance and are triggered upon the reception of a new opportunistic DOTS signal. Once all rule have been applied, signalling information MUST be discarded by the router. When embedding the information into other IPv6 packets, the router MUST decrease the TTL by one since opportunistic signalling does not prevent loops in the dissemination of signalling.

3.4 Processing by a DOTS opportunistic-capable relay

If a DOTS relay has DOTS capabilities, it will apply the same strategy as a DOTS client by making attempts of direct connections to the DOST server and in addition it inserts the Hop-by-Hop header DOTS signalling option in leaving IPv6 packets using the strategy specified above.

3.5 Processing by a DOTS opportunistic-capable server

When the IP layer of the host where the DOTS server is running receives an IPv6 packet carrying a Hop-by-Hop DOTS signal option header it MUST extracts the content of the option and provides the attributes data to the server program.

4 Deployment considerations

This mechanism will be potentially used by networks with IPv6 capable elements and requires that of IPv6 traffic exist in the network during the attack. The existing IPv6 traffic to be used could be of any type from management or user levels. It is also important to emphasize that while our mechanism utilizes an IPv6 header field, it can also be used to signal IPv4 attacks as well - given that the network devices are dual stacked.

IPv6 extension headers are often rate-limited or dropped entirely [HBH-HEADER]. To be able to use the mechanism specified in this document, network operators need to avoid discarding packets or ignoring the processing of the hop-by-hop option on their deployed network elements. However, instead of dropping or ignoring packets with hop-by-hop option carrying DOTS signal, they need to assign these packets to slow forwarding path, and be processed by the router’s CPU. This behavior will not affect the performance of the network devices since the network is already facing a DDoS attack and fast forwarding paths are saturated by the attacker traffic.

If the DOTS server, relay and the client are located in the same administrative domain, marking the IPv6 packets with the proposed hop-by-hop header option could be done in a straight forward way, while considering that an agreement exists inside the domain to avoid dropping or rate limiting of IPv6 extension headers as described above. The proposed mechanism becomes less practical and difficult to deploy when the DOST server is running on the Internet. In such scenario, the mechanism could be used in the intra-domain part to deliver the hop-by-hop option carrying the DOTS signal until it reaches a DOTS relay located in the same domain as the client, then the relay will apply mechanisms provided by the DOTS transport protocol [I-D.draft-reddy-dots-transport] to inform the server running on Internet about the attack. This deployment scenario

requires that at least one DOTS relay is deployed in the same domain than the DOTS client.

5 Impact on existing IP layer implementations

For this option to be applicable within an IP system, it requires modifications to existing IP layer implementation. At DOTS capable nodes (client, relay and server), it requires a service interface used by upper-layer protocols and application programs to ask the IP layer to insert and listen to the Hop-by-Hop header option in IPv6 packets with the content and strategies described in Section 3. A DOTS client invokes the service interface to insert the option, A DOTS relay invokes the service interface for listening and inserting the option, and finally a DOTS server only invokes the service interface to listen to the DOTS signalling option.

Intermediate nodes (routers or middle boxes) IP layer needs to be extended to perform processing of the new Hop-by-Hop header option as described in Section 3. They mainly parse the first host attribute of the option and make a selection of a leaving IPv6 packet where the option will be inserted.

Every node inserting the new proposed Hop-by-Hop option SHOULD only select IPv6 packets with enough left space to avoid fragmentation.

6 Security Considerations

Any IPv6 header option could be used by an attacker to create an attack on the routers and intermediate boxes that process packets containing the option. The proposed IPv6 option in this document MAY be abused by an attacker to create a covert channel at the IP layer where data is hidden inside the content of the option [RFC6564]. However, this attack is not specific to the proposed option and it is a known issue of IPv6 header extensions and options. The option MAY also be used by an attacker to forge or modify opportunistic DOTS signal leading to trigger additional processing on intermediate nodes and DOTS servers.

However the proposed option should be only initiated by a DOTS client and information embedded in new IPv6 messages by opportunistic DOTS capable routers. Defining proper policies to filter all messages with this option set and originated from other nodes would limit security issues since these DOTS opportunistic-capable agents SHOULD be trustworhy.

In addition, the message MAY be signed using techniques to enforce authenticity and integrity over the opportunistic DOTS signal

channel. The signalling message specification includes a flag to indicate if the message is signed by the choice of the signature algorithm is let to the users. This signature has to be computed by the DOTS opportunistic-capable client and checked by the DOTS opportunistic-capable relay or router. Hence, intermediate routers MUST NOT modify the message and its signature except the TTL, which so has not be considered during the signature computation.

Assuming a compromised router, the attacker could nevertheless replay the message or increase the TTL but thanks to the unique policy-id all intermediate-DOTS capable router will drop such messages and thus limiting their forwarding in the network.

Besides, an attacker can also listen opportunistic DOTS signals to monitor the impact of its own attack. These considerations are not specific to the proposed option and supposes that the attacker is able to compromise intermediate routers.

7 IANA Considerations

This draft defines a new IPv6 [RFC2460] hop-by-hop option. This requires an IANA RFC3692-style update of: http://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml and ultimately the assignment of a new hop-by-hop option according to the guidelines described in [RFC5237].

7 References

7.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, DOI 10.17487/RFC6564, April 2012, <http://www.rfc-editor.org/info/rfc6564>.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <http://www.rfc-editor.org/info/rfc7045>.

7.2 Informative References

[I-D.draft-ietf-dots-requirements]
A. Mortensen., R. Moskowitz., and T. Reddy., "DDoS Open Threat Signaling Requirements", draft-ietf-dots-requirements-00 (work in progress), October 2015.

[kuhrer2014exit]
Kuhrer, Marc and Hupperich, Thomas and Rossow, Christian and Holz, Thorsten. Exit from Hell? Reducing the Impact of Amplification DDoS Attacks. In: 23rd USENIX Security Symposium (USENIX Security 14).

[I-D.draft-reddy-dots-transport]
T. Reddy., D. Wing., P. Patil., M. Geller., M. Boucadair.,and R. Moskowitz., "Co-operative DDoS Mitigation", draft-reddy-dots-transport-03 (work in progress), March 2016.

[rootops-ddos]
rootops.: Events of 2015-11-30. Online: http://root-servers.org/news/events-of-20151130.txt

[ripe-dnsmon-ddos]
RIPE NCC DNS Monitoring Service (DNSMON). Online: https://atlas.ripe.net/dnsmon/

[HBH-HEADER]
Baker, F., "IPv6 Hop-by-Hop Options Extension Header",Work in Progress, draft-ietf-6man-hbh-header-handling-03,March 2016.

Acknowledgements

This work is partly funded by FLAMINGO, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Programme.

Authors' Addresses

Jerome Francois
Inria
615 rue du Jardin Botanique
54600 Villers-les-Nancy
France

Phone: +33 3 83 59 30 66
EMail: jerome.francois@inria.fr

Abdelkader Lahmadi
University of Lorraine - LORIA
615 rue du Jardin Botanique
54600 Villers-les-Nancy
France

Phone: +33 3 83 59 30 00
Email: Abdelkader.Lahmadi@loria.fr

Marco Davids
SIDN Labs
Meander 501
6825 MD Arnhem
The Netherlands

Email: marco.davids@sidn.nl

Giovane C. M. Moura
SIDN Labs
Meander 501
6825 MD Arnhem
The Netherlands

Email: giovane.moura@sidn.nl

DOTS
Internet Draft
Intended status: Standard Track
Expires: Nov 2016

T. Fu
Huawei
D. Zhang
Alibaba
L. Xia
M. Li
Huawei

June 14, 2016

IPFIX IE Extensions for DDoS Attack Detection
draft-fu-dots-ipfix-extension-01.txt

Status of this Memo

This Internet-Draft will expire on December 14, 201616.

Abstract

DDoS Open Threat Signaling (DOTS) Working Group is for developing the standard signaling mechanisms, together with the DDoS related telemetry and threat handling requests and data transmitted by them used in DDoS problem space. Although IP Flow Information Export (IPFIX), Packet Sampling (PSAMP), and Packet Selection methods are useful for network security inspection, there are still some gaps existing to identify some categories of DDoS attacks. To fill in the gaps, this document describes the connection sampling mechanism and explains why it is needed for detecting DDoS attacks. It also defines several new IPFIX Information Elements (IEs). Then, it presents some examples to show how to use these new IPFIX IEs together with the existing IPFIX IEs to detect specific DDoS attacks.

Table of Contents

1. Introduction

As network security issues arising dramatically nowadays, network administrators are eager to detect and identify attacks as early as possible, generate countermeasures with high agility. Due to the enormous amount of network attack types, metrics useful for attack detection are also enormous. Moreover, attacking methods are evolved rapidly, which brings challenges to designing detection mechanism.

Specifically, DOTS WG aims for developing the standard solution to fight against the DDoS attacks. The following sentence is from the DOTS WG charter:

"The aim of DDoS Open Threat Signaling (DOTS) is to develop a standards based approach for the realtime signaling of DDoS related telemetry and threat handling requests and data between elements concerned with DDoS attack detection, classification, traceback, and mitigation."

According to the above sentence, the signaling mechanisms and contents are all the essential parts of the solution, in which the contents refer to the realtime DDoS related telemetry information and threat handling messages.

The IPFIX Protocol [RFC7011] defines a generic exchange mechanism for flow information and events. It supports source-triggered exporting of information via the push model approach. The IPFIX Information Model [IPFIX-IANA] defines a list of standard Information Elements (IEs) which can be carried by the IPFIX protocol. The IPFIX requirement [RFC3917] points out that one of the target applications of IPFIX is attack and intrusion detection. Although the existing IPFIX/PSAMP protocol, packet selection methods, as well as the related standard IEs provide a rich source of data for security inspection by checking the status/events of the traffic, there are still some gaps existing to identify some categories of the DDoS attacks. More detailed gap analysis is given in the following section.

This document focuses on the DDoS related telemetry information part for DOTS, and proposes using the connection sampling method with a set of IPFIX IEs for the goal of inspecting mainly some connection-based and Zero-Day DDoS attacks, which normally are the kinds of the low & slow DDoS attack and not easy to be inspected as flood attacks. Some of these IPFIX IEs already exist; some are the new defined ones with their formats specified. The wise utilization of these IEs will improve the DDoS attack inspection and will support the offline analysis of data from different operators in the future with minimal resource consumption, which is very necessary for increasing the operators' intelligence of identifying new and unknown DDoS attacks.

This document is structured as following: Section 3 discusses the connection sampling mechanism and introduces the new IPFIX IEs derived from relevant use cases. Section 4 describes how to use these IEs to detect specific DDoS attacks.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

2.1. Terminology

IPFIX-specific terminology (Information Element, Template, Template Record, Options Template Record, Template Set, Collector, Exporter, Data Record, etc.) used in this document is defined in Section 2 of [RFC7011]. As in [RFC7011], these IPFIX-specific terms have the first letter of a word capitalized.

This document also makes use of the same terminology and definitions as [I-D.draft-ietf-dots-requirements] and [I-D.draft-ietf-dots-use-cases].

The following are the new terms in this document.

o Connection Sampling

Connection Sampling is a connection oriented sampling mechanism. If one connection is selected for DDoS attack detection, then all the packets (if possible) of this connection will be sampled during one detection period.

o Victim

The target that suffers DDoS attack.

o Observer

Devices or software deployed at observation point defined in IPFIX.

3. Connection Sampling and new IEs

3.1. Packet Sampling vs Connection Sampling

Packet sampling selection is a widely used method to select packets from network traffic for reporting. Its selection operations include time-based selection, count-based selection, random selection, probability-based selection and so on. Although it is easy and efficient, it still has a number of limitations in inspecting some types of DDoS attack:

- Several research projects [N. DUFFIELD, 2003], [D. BRAUCKHOFF 2006] show that packet sampling impacts greater on small volumetric flows (with only few packets) due to the smaller sampling probability compared with large volumetric flows, which means that packet sampling may impair the detection performance for small volumetric flow based DDoS attacks; Connection sampling can pay more attention to small flows than packet sampling.

- The communication is 2-way between source and destination. Current packet sampling is used to select a subset of packets from all the observed packets. One of its purposes is to select the appropriate packets to estimate the whole traffic. Although packet sampling can produce flows by using property matching or hash method, it does not consider semantics of the connection (i.e. whether two flows are belonging to one connection or not). In this scenario, packet sampling is applied independently in each direction. If packets are sampled only in one direction, then it will be difficult or inaccurate to detect specific DDoS attacks such as SNMP/DNS Reflected Amplification because of the loss of the information in the opposite direction. It cannot distinguish whether there are too much requests from the victim or not.

- Although packet sampling at full line rate, i.e. with probability 100%, is not excluded in principle, resource constraints may not permit it in practice [RFC5474]. For certain DDoS attack such as HTTP Slowloris, enough packets are needed to realize the detection. In this scenario, connection sampling can provide more meaningful packets than packet sampling because all the packets it samples are belonging to the target connection.

- o Connection sampling method uses some new connection related IEs for attack analysis because of their meaning/value available for judging the status of one connection, which current packet sampling method is lack of. For example, tcpControlStateBits is an IE to record the current state of TCP session. If a packet with erroneous flag is identified in any stage of three handshakes, it should be considered as a symptom of exception.

As a consequence from the above analysis, a connection oriented sampling method is more suitable for the security application: Rather than sampling a small part of packets in the traffic between the communication peers, the connection sampling records all (if possible) TCP/UDP connection packets (including packets during connection setup and close phase if there is) between them once that connection is selected to be sampled. In nature, the connection sampling method is able to track the complete working status of the connection state machine. So, it can identify the abnormal state of the connection or the attack easily and accurately. Although the IPFIX/PSAMP also supports the connection sampling mechanism (that is the packet filtering technology for packet selection [RFC5475]), it does not explicitly discuss how to use this method for the detection of connection-based and Zero-Day DDoS attacks in a systematical way. Furthermore, if the observer (e.g. device, middle box) supports the export of the new IPFIX IEs proposed in this document, the traffic volume between exporter and analyzer can be greatly reduced compared with PSAMP, which should export the detailed packet information for further attack analysis.

3.2. Use Cases for New IEs

In this section, several use cases are discussed to identify the requirements where new IEs are desirable for the network attacks detection.

3.2.1. Upstream/Downstream Counters

Take ICMP reflection attack as an example, ICMP flow model has features such as the ICMP Echo/Echo Reply dominate the whole traffic flow, ICMP packet interval is usually not too short (normally 1 pkt/s). Usually, the normal ratio between ICMP echo to ICMP echo reply packets is around 1:1. When a DDOS reflection attack happens, a sudden burst of messages to a destination endpoint can be detected. In turn, the ratio between echo reply and echo packets will be significantly biased from the normal ratio, i.e., exceed 20:1. So the proper way to distinguish an attack from the normal communication is to check this ratio.

However, the current IPFIX IEs for ICMP contain the ICMP type and code for both IPv4 and IPv6 only for a single ICMP packet rather than statistical property of the ICMP session. Further metrics like the cumulated sum of various counters should be calculated based on sampling method defined by the Packet SAMPling (PSAMP) protocol [RFC5477]. Similar problems occur in TCP, UDP, SNMP and DNS attacks. It would be useful to calculate the number of the upstream and downstream packets for one connection separately over time in order to detect the anomalies of the network. For ICMP reflection attack, a more generic approach is to define two basic metrics icmpEchoCount and icmpEchoReplyCount as new IPFIX IEs to represent the cumulated upstream and downstream packets counter within a ICMP connection.

Note that in some case, the asymmetric routing mainly caused by the wide application of multipath technologies (e.g., load balancing, link aggregation) in network will make the bidirectional connection sampling on some network devices over the multipath to be not possible. This problem can be avoided by strategically deploying and enabling the connection sampling function in the network devices which are not located over the multipath.

3.2.2. Fragment statistic

Fragment attack employs unexpected formats of fragmentation, e.g. without last fragment or incorrect fragment offset[RFC791], which result in errors such as fragmentation buffer overrun and fragment overlapped. Existing IPFIX fragmentation metrics includes fragmentOffset, fragmentIdentification, fragmentFlags, which only indicate the attributes of a single fragment, and are not suitable for attack detection. Instead, the network attack should be observed based upon a historic, integrated view of fragmented packets of a connection. For instances, if more than 500 out of 1000 fragmented packets have fragment errors, it is likely that a fragment attack happens.

Therefore, a number of new IEs associated with fragment statistics are proposed as follows:

- fragmentPacketCount: The number of the fragmented packets of the same connection should be checked, and this metric is proposed;

- fragmentFirstTooShortCount: Attacker might intent to exclude destination port from the first fragment so as to bypass detection from firewall. This metric is proposed to indicate the number of the invalid first fragments in the observed connection;

- o fragmentFlagErrorCount: This metric is proposed to detect early whether the fragment flags are incorrectly set on purpose.

- o fragmentOffestErrorCount: This metric is proposed to count the number of fragments with offset error, and the value can be used to indicate attack occurs;

3.2.3. Response Time Calculation

For other DDoS attacks such as Http slowloris, there will be too many connections that should be kept in the victim (server), which lead to excessive resource consumption. As a result, the response time between client and server will increase greatly. Challenge Collapasar(CC) attack can also exhaust the resources of the server and generate the similar results. Thus, the following IEs are proposed as a symptom of these kinds of attacks:

serverResponseTime: For tcp, it denotes the time difference between the time point that the observer views the SYN packet from client to server and the time point that the observer views the SYN-ACK packet from server to client.

clientResponseTime: For tcp, it denotes the time difference between the time point that the observer views the SYN-ACK packet from server to client and the time point that the observer views the ACK packet from client to server.

sessionResponseTime: The sum of serverResponseTime and clientResponseTime. It is the Round Trip Time (RTT) between client and server.

3.2.4. Symptoms of Exceptions

In http slowloris attack the client may send packets to victim periodically which can cause the performance lost on the server. The characteristic of the attack is that there are too many connections on the victim. However, the volume for these connections is small. In order to detect this attack, the first step is to get the packets that are belonging to the same connection. The second step is to find the periodicity. Thus the two indices pktTimeInterval and pktTimeIntervalVariance are needed. The index pktTimeInterval denotes the average time difference between two successive packets and the index pktTimeIntervalVariance denotes the variance of multiple time difference. Large pktTimeInterval and small pktTimeIntervalVariance can be a symptom of slow packet attack. On the other hand, the payload size of the packets in http slowloris

attack is very small and the size difference is also small. So the index octetVariance can be used to identify the characteristic.

To degrade the performance of the victim, the malicious clients may send too many out-of-order packets, which will consume too much memory on the server. Although out-of-order packets are permit in the TCP protocol, it is possible to be leveraged to cause DDoS attack. So the index tcpOutoforderTotalCount is helpful to detect this kind of exception. For observer, it maintains one counter for each tcp connection. The initial sequence number of the client is saved in the counter. The counter increases by the sequence number of the packets it sees from client to server. If the observer sees a packet with lower sequence number than the current counter value, then the packet will be considered as an out-of-order packet.

In IPFIX, the index tcpControlBits is used to record the corresponding status bits in TCP header of the packets[IPFIX-IANA]. In order to detect the application attacks which can cause the protocol exception such as the wrong use of the tcp status bits before and after the tcp connection establishment, another index called tcpControlStateBits is needed. For example, when the observer sees the SYN packet from client to server, it sets 15th bit of tcpControlStateBits to 1; when it sees the SYN-ACK packet from server to client, it sets 14th bit to 1, and so on. If one endpoint sends the packet with wrong bits during the establishment of the connection, then the observer will identify the exception by the value of tcpControlStateBits.

3.2.5. Extended Value of FlowEndReason

Refer to [IPFIX-IANA], there are 5 defined reasons for Flow termination, with values ranging from 0x01 to 0x05:

0x01: idle timeout

0x02: active timeout

0x03: end of Flow detected

0x04: forced end

0x05: lack of resources

There is an additional reason caused by state machine anomaly. When FIN/SYN is sent, but no ACK is replied after a waiting timeout, the existing five reasons do not match this case. Therefore, a new value

is proposed to extend the FlowEndReason, which is 0x06: protocol exception timeout.

3.3. Definition of New IEs

The following is the table of all the new IEs that a device would need to export for attack statistic analysis. The recommended registrations to IANA are described in the IANA considerations section.

Field Name	Size (bits)	IANA IPFIX ID	Description
fragmentPacketCount	32	TBD	Counter of session fragments
fragmentFirstTooShortCount	32	TBD	Number of packets with first fragment too short
fragmentFlagErrorCount	32	TBD	Number of fragments with erroneous flag
fragmentOffsetErrorCount	32	TBD	Number of fragments with erroneous offset
icmpEchoCount	32	TBD	The number of ICMP echo.
icmpEchoReplyCount	32	TBD	The number of ICMP echo reply
octetVariance	64	TBD	IP packet byte variance statistic
tcpControlStateBits	16	TBD	tcp states
tcpOutoforderTotalCount	64	TBD	out of order packets statistic
pktTimeInterval	64	TBD	the average time interval between two successive packets
pktTimeIntervalVariance	64	TBD	the variance of pktTimeInterval
serverResponseTime	16	TBD	the response time of a server
clientResponseTime	16	TBD	the response time of a client
sessionResponseTime	16	TBD	the response time of a session

Table 1: Information Element Table

4. Application of the New IEs for Attack Detection

This section presents a number of examples to help for the easy understanding of the application of these new IEs for attack detection.

4.1. Detect ICMP Reflection Attack

According to previous analysis, the template for detecting ICMP reflection attack should at least contain IEs shown in Table 2.

Set ID = 2	Length = 40 octets
Template ID TBD	Field Count = 8
0 sourceIPv4Address	Field Length = 4
0 destinationIPv4Address	Field Length = 4
0 protocolIdentifier	Field Length = 1
0 packetDeltaCount	Field Length = 8
0 icmpEchoCount	Field Length = 4
0 icmpEchoReplyCount	Field Length = 4
0 flowStartSeconds	Field Length = 4
0 flowEndSeconds	Field Length = 4

Table 2: Template example for detecting ICMP attack

An example of the actual ICMP event data record is shown below in a readable form as below:

{sourceIPv4Address = 192.168.0.101, destinationIPv4Address = 192.168.0.201, protocolIdentifier = 1, packetDeltaCount = 3000, icmpEchoCount = 120, icmpEchoReplyCount = 2880, flowStartSeconds = 100, flowEndSeconds = 200}

protocolIdentifier = 1 represents the ICMP proptocol. There are 30 ICMP messages transmited per second. The ICMP Echo Reply to ICMP

Echo packet ratio is 24:1, which indicates a high possibility of ICMP reflection attack.

4.2. Detect Fragment Attack

The template for detecting fragment attack should at least contain IEs shown in Table 3. It requires the observation point to trace complete fragmented packet and accumulate the errors.

Set ID = 2	Length = 48 octets
Template ID TBD	Field Count = 10
0 sourceIPv4Address	Field Length = 4
0 destinationIPv4Address	Field Length = 4
0 protocolIdentifier	Field Length = 1
0 packetDeltaCount	Field Length = 8
0 fragmentPacketCount	Field Length = 4
0 fragmentFirstTooShortCount	Field Length = 4
0 fragmentFlagErrorCount	Field Length = 4
0 fragmentOffsetErrorCount	Field Length = 4
0 flowStartSeconds	Field Length = 4
0 flowEndSeconds	Field Length = 4

Table 3: Template example for detecting fragment attack

An example of the actual fragment attack record is shown below in a readable form as below:

{sourceIPv4Address = 192.168.0.101, destinationIPv4Address = 192.168.0.201, protocolIdentifier = 6, packetDeltaCount = 5000, fragmentPacketCount = 4000, fragmentFirstTooShortCount = 0, fragmentFlagErrorCount = 0, fragmentOffestErrorCount = 3000, flowStartSeconds = 100, flowEndSeconds = 200}

In this case, fragment offset errors are used to exhaust resource at the receiver.

4.3. Detect Slowloris Attack

The template for detecting resource exhausting application attack such as http slowloris attack should contain a subnet of IEs shown in Table 4.

Set ID = 2	Length = 48 octets
Template ID TBD	Field Count = 10
0 sourceIPv4Address	Field Length = 4
0 destinationIPv4Address	Field Length = 4
0 protocolIdentifier	Field Length = 1
0 serverResponseTime	Field Length = 2
0 clientResponseTime	Field Length = 2
0 sessionResponseTime	Field Length = 2
0 pktTimeInterval	Field Length = 4
0 pktTimeIntervalVariance	Field Length = 4
0 flowStartSeconds	Field Length = 4
0 flowEndSeconds	Field Length = 4

Table 4: Template example for detecting slowloris attack

An example of the actual record is shown below in a readable form as below:

{sourceIPv4Address = 192.168.0.101, destinationIPv4Address = 192.168.0.201, protocolIdentifier = 6, serverResponseTime = 200, clientResponseTime = 10, sessionResponseTime = 210, pktTimeInterval = 500, pktTimeIntervalVariance = 1000, flowStartSeconds = 100, flowEndSeconds = 200}

4.4. Detect Out-of-order Packets Attack

The template for detecting out-of-order packets attack should contain IEs shown in Table 5.

Set ID = 2	Length = 32 octets
Template ID TBD	Field Count = 10
0 sourceIPv4Address	Field Length = 4
0 destinationIPv4Address	Field Length = 4
0 protocolIdentifier	Field Length = 1
0 packetDeltaCount	Field Length = 8
0 tcpOutoforderTotalCount	Field Length = 4
0 flowStartSeconds	Field Length = 4
0 flowEndSeconds	Field Length = 4

Table 5: Template example for detecting out-of-order attack

An example of the actual record is shown below in a readable form as below:

{sourceIPv4Address = 192.168.0.101, destinationIPv4Address = 192.168.0.201, protocolIdentifier = 6, packetDeltaCount =3000, tcpOutoforderTotalCount = 2000, flowStartSeconds = 100, flowEndSeconds = 200}

5. Security Considerations

No additional security considerations are introduced in this document. The same security considerations as for the IPFIX protocol [RFC7011] apply.

6. IANA Considerations

The following information elements are requested from IANA IPFIX registry.

Name: fragmentPacketCount

Description: This Information Element is the counter of session fragments.

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: fragmentFirstTooShortCount

Description: This Information Element indicates the number of packets with first fragment too short.

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: fragmentFlagErrorCount

Description: This Information Element specifies number of fragments with flag error. When the DF bit and MF bit of the fragment flag are set in the same fragment, there is an error at the fragment flag.

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: fragmentOffsetErrorCount

Description: This Information Element specifies number of fragments with offset error.

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: icmpEchoCount

Description: icmp Echo packets.

Abstract Data Type: unsigned32

Data Type Semantics: deltaCounter

Name: icmpEchoReplyCount

Description: icmp Echo Reply packets.

Abstract Data Type: unsigned32

Data Type Semantics: deltaCounter

Name: octetVariance

Description: IP packet byte variance statistic.

Abstract Data Type: unsigned64

Data Type Semantics: quantity

Name: tcpControlStateBits

Description: the current tcp states of the connection.

Abstract Data Type: unsigned16

Data Type Semantics: flags

Name: tcpOutoforderTotalCount

Description: out of order packets statistic.

Abstract Data Type: unsigned64

Data Type Semantics: totalCounter

Name: pktTimeInterval

Description: the average time interval between two successive packets in a flow.

Abstract Data Type: unsigned32

Data Type Semantics: quantity

Name: pktTimeIntervalVariance

Description: the variance of the time intervals between two successive packets in a flow.

Abstract Data Type: unsigned64

Data Type Semantics: quantity

Name: serverResponseTime

Description: the response time of a server.

Abstract Data Type: unsigned16

Data Type Semantics: quantity

Name: clientResponseTime

Description: the response time of a client.

Abstract Data Type: unsigned16

Data Type Semantics: quantity

Name: sessionResponseTime

Description: the response time of a session.

Abstract Data Type: unsigned16

Data Type Semantics: quantity

A new value is added to FlowEndReason:

0x06: protocol exception timeout

The flow was terminated due to protocol state machine anomaly and unexpected timeout.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC7011] Claise, B., Trammell, B., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, September 2013.

[RFC3917] Quittek, J., Zseby, T., Claise, B., Zander, S., "Requirements for IP Flow Information Export (IPFIX)", RFC 3917, October 2004.

[RFC5474] N. Duffield Ed., D. Chiou, B. Claise, A. Greenberg, M. Grossglauser, J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, March 2009.

[RFC5475] T. Zseby, M. Molina, N. Duffield, S. Niccolini, F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, March 2009.

[RFC5476] B. Claise, Ed., A. Johnson, J. Quittek, "Packet Sampling (PSAMP) Protocol Specifications", RFC 5476, March 2009.

[RFC5477] T. Dietz, B. Claise, P. Aitken, F. Dressler, G. Carle, " Information Model for Packet Sampling Exports ", RFC 5477, March 2009.

7.2. Informative References

[IPFIX-IANA]

IANA, "IPFIX Information Elements registry",

<http://www.iana.org/assignments/ipfix>.

[I-D.draft-ietf-dots-requirements]

Mortensen, A., Moskowitz, R., Reddy, T., "DDoS Open Threat Signaling Requirements", work in progress, October, 2015.

[I-D.draft-ietf-dots-use-cases]

Dobbins, R., Fouant, S., Migault, D., Moskowitz, R., Teague, N., Xia, L., " Use cases for DDoS Open Threat Signaling", work in progress, October, 2015.

[D. BRAUCKHOFF 2006]

Daniela Brauckhoff, Bernhard Tellenbach, Arno Wagner, Martin May, and Anukool Lakhina. 2006. Impact of packet sampling on anomaly detection metrics. In Proceedings of the 6th ACM SIGCOMM conference on Internet measurement (IMC ’06). ACM, New York, NY, USA, 159-164.

[N. DUFFIELD, 2003]

DUFFIELD, N., LUND, C., AND THORUP, M., Estimating Flow Distributions from Sampled Flow Statistics. In ACM SIGCOMM (Karlsruhe, August 2003).

8. Acknowledgments

The authors would thank Danping He and Yibo Zhang for their great help during the initial period of this draft.

The authors would also thank Tienan Wang for his explain about the implementation of DDoS attack solutions.

This document was prepared using 2-Word-v2.0.template.dot.

Authors’ Addresses

Tianfu Fu
Huawei
Q11, Huanbao Yuan, 156 Beiqing Road, Haidian District
Beijing 100095
China

Email: futianfu@huawei.com

DaCheng Zhang
Alibaba

Email: Dacheng.zdc@alibaba-inc.com

Liang Xia (Frank)
Huawei

101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: Frank.xialiang@huawei.com

Bo Zhang (Alex)
Huawei

101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: Alex.zhangbo@huawei.com

Min Li
Huawei

Huawei Technologies Duesseldorf GmbH, European Research Center,
Riesstr. 25, 80992 Muchen, Germany
Email: l.min@huawei.com

DOTS
Internet-Draft
Intended status: Informational
Expires: January 6, 2017

A. Mortensen
Arbor Networks, Inc.
F. Andreasen
T. Reddy
Cisco Systems, Inc.
C. Gray
Comcast, Inc.
R. Compton
Charter Communications, Inc.
N. Teague
Verisign, Inc.
July 05, 2016

Distributed-Denial-of-Service Open Threat Signaling (DOTS) Architecture
draft-ietf-dots-architecture-00

Abstract

This document describes an architecture for establishing and maintaining Distributed Denial of Service (DDoS) Open Threat Signaling (DOTS) within and between domains. The document does not specify protocols or protocol extensions, instead focusing on defining architectural relationships, components and concepts used in a DOTS deployment.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 6, 2017.

Table of Contents

1. Context and Motivation

Signaling the need for help defending against an active distributed denial of service (DDoS) attack requires a common understanding of mechanisms and roles among the parties coordinating defensive response. The signaling layer and supplementary messaging is the focus of DDoS Open Threat Signaling (DOTS). DOTS defines a method of coordinating defensive measures among willing peers to mitigate attacks quickly and efficiently, enabling hybrid attack responses coordinated locally at or near the target of an active attack, or anywhere in-path between attack sources and target.

This document describes an architecture used in establishing, maintaining or terminating a DOTS relationship within a domain or between domains.

1.1. Terminology

1.1.1. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.1.2. Definition of Terms

This document uses the terms defined in [I-D.ietf-dots-requirements].

1.2. Scope

In this architecture, DOTS clients and servers communicate using the DOTS signaling. As a result of signals from a DOTS client, the DOTS server may modify the forwarding path of traffic destined for the attack target(s), for example by diverting traffic to a mitigator or pool of mitigators, where policy may be applied to distinguish and discard attack traffic. Any such policy is deployment-specific.

The DOTS architecture presented here is applicable across network administrative domains - for example, between an enterprise domain and the domain of a third-party attack mitigation service - as well as to a single administrative domain. DOTS is generally assumed to be most effective when aiding coordination of attack response between two or more participating network domains, but single domain scenarios are valuable in their own right, as when aggregating intra-domain DOTS client signals for inter-domain coordinated attack response.

This document does not address any administrative or business agreements that may be established between involved DOTS parties. Those considerations are out of scope. Regardless, this document assumes necessary authentication and authorization mechanism are put in place so that only authorized clients can invoke the DOTS service.

1.3. Assumptions

This document makes the following assumptions:

- All domains in which DOTS is deployed are assumed to offer the required connectivity between DOTS agents and any intermediary network elements, but the architecture imposes no additional limitations on the form of connectivity.

- Congestion and resource exhaustion are intended outcomes of a DDoS attack [RFC4732]. Some operators may utilize non-impacted paths or networks for DOTS, but in general conditions should be assumed to be hostile and that DOTS must be able to function in all circumstances, including when the signaling path is significantly impaired.

- There is no universal DDoS attack scale threshold triggering a coordinated response across administrative domains. A network domain administrator, or service or application owner may arbitrarily set attack scale threshold triggers, or manually send requests for mitigation.

- Mitigation requests may be sent to one or more upstream DOTS servers based on criteria determined by DOTS client administrators. The number of DOTS servers with which a given DOTS client has established signaling sessions is determined by local policy and is deployment-specific.

- The mitigation capacity and/or capability of domains receiving requests for coordinated attack response is opaque to the domains sending the request. The domain receiving the DOTS client signal may or may not have sufficient capacity or capability to filter any or all DDoS attack traffic directed at a target. In either case, the upstream DOTS server may redirect a request to another DOTS server. Redirection may be local to the redirecting DOTS server’s domain, or may involve a third-party domain.

- DOTS client and server signals, as well as messages sent through the data channel, are sent across any transit networks with the same probability of delivery as any other traffic between the DOTS client domain and the DOTS server domain. Any encapsulation required for successful delivery is left untouched by transit

network elements. DOTS server and DOTS client cannot assume any preferential treatment of DOTS signals. Such preferential treatment may be available in some deployments, and the DOTS architecture does not preclude its use when available. However, DOTS itself does not address how that may be done.

- The architecture allows for, but does not assume, the presence of Quality of Service (QoS) policy agreements between DOTS-enabled peer networks or local QoS prioritization aimed at ensuring delivery of DOTS messages between DOTS agents. QoS is an operational consideration only, not a functional part of the DOTS architecture.

- The signal channel and the data channel may be loosely coupled, and need not terminate on the same DOTS server.

2. Architecture

The basic high-level DOTS architecture is illustrated in Figure 1:

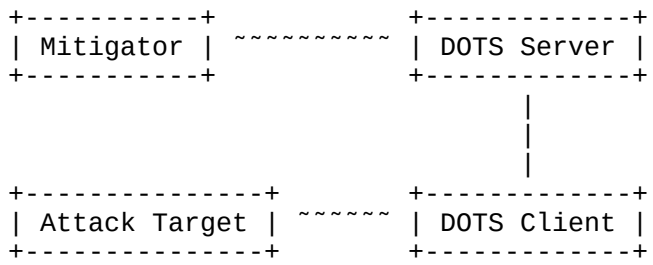


Figure 1: Basic DOTS Architecture

A simple example instantiation of the DOTS architecture could be an enterprise as the attack target for a volumetric DDoS attack, and an upstream DDoS mitigation service as the Mitigator. The enterprise (attack target) is connected to the Internet via a link that is getting saturated, and the enterprise suspects it is under DDoS attack. The enterprise has a DOTS client, which obtains information about the DDoS attack, and signals the DOTS server for help in mitigating the attack. The DOTS server in turn invokes one or more mitigators, which are tasked with mitigating the actual DDoS attack, and hence aim to suppress the attack traffic while allowing valid traffic to reach the attack target.

The scope of the DOTS specifications is the interfaces between the DOTS client and DOTS server. The interfaces to the attack target and the mitigator are out of scope of DOTS. Similarly, the operation of both the attack target and the mitigator are out of scope of DOTS.

Thus, DOTS neither specifies how an attack target decides it is under DDoS attack, nor does DOTS specify how a mitigator may actually mitigate such an attack. A DOTS client's request for mitigation is advisory in nature, and may not lead to any mitigation at all, depending on the DOTS server domain's capacity and willingness to mitigate on behalf of the DOTS client's domain.

As illustrated in Figure 2, there are two interfaces between the DOTS server and the DOTS client:

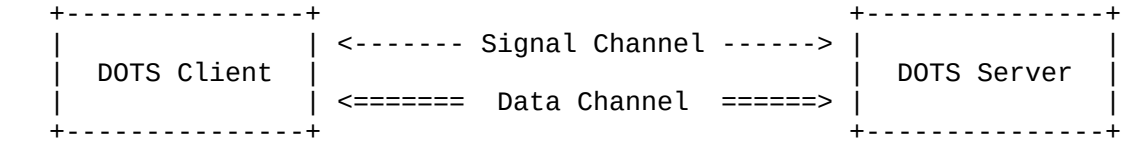


Figure 2: DOTS Interfaces

The DOTS client may be provided with a list of DOTS servers, each associated with one or more IP addresses. These addresses may or may not be of the same address family. The DOTS client establishes one or more signaling sessions by connecting to the provided DOTS server addresses.

[[EDITOR'S NOTE: We request feedback from the working group about the mechanism of server discovery.]]

The primary purpose of the signal channel is for a DOTS client to ask a DOTS server for help in mitigating an attack, and for the DOTS server to inform the DOTS client about the status of such mitigation. The DOTS client does this by sending a client signal, which contains information about the attack target or targets. The client signal may also include telemetry information about the attack, if the DOTS client has such information available. The DOTS server in turn sends a server signal to inform the DOTS client of whether it will honor the mitigation request. Assuming it will, the DOTS server initiates attack mitigation (by means outside of DOTS), and periodically informs the DOTS client about the status of the mitigation. Similarly, the DOTS client periodically informs the DOTS server about the client's status, which at a minimum provides client (attack target) health information, but it may also include telemetry information about the attack as it is now seen by the client. At some point, the DOTS client may decide to terminate the server-side attack mitigation, which it indicates to the DOTS server over the signal channel. A mitigation may also be terminated if a DOTS client-specified mitigation time limit is exceeded; additional considerations around mitigation time limits may be found below. Note that the signal channel may need to operate over a link that is

experiencing a DDoS attack and hence is subject to severe packet loss and high latency.

While DOTS is able to request mitigation with just the signal channel, the addition of the DOTS data channel provides for additional and more efficient capabilities; both channels are required in the DOTS architecture. The primary purpose of the data channel is to support DOTS related configuration and policy information exchange between the DOTS client and the DOTS server. Examples of such information include, but are not limited to:

- Creating identifiers, such as names or aliases, for resources for which mitigation may be requested. Such identifiers may then be used in subsequent signal channel exchanges to refer more efficiently to the resources under attack, as seen in Figure 3 below, using JSON to serialize the data:

```
{
    "https1": [
        "172.16.168.254:443",
        "172.16.169.254:443",
    ],
    "proxies": [
        "10.0.0.10:3128",
        "[2001:db9::1/128]:3128"
    ],
    "api_urls": "https://apiserver.local/api/v1",
}
```

Figure 3: Protected resource identifiers

- Black-list management, which enables a DOTS client to inform the DOTS server about sources to suppress.

- White-list management, which enables a DOTS client to inform the DOTS server about sources from which traffic should always be accepted.

- Filter management, which enables a DOTS client to install or remove traffic filters dropping or rate-limiting unwanted traffic.

- DOTS client provisioning.

Note that while it is possible to exchange the above information before, during or after a DDoS attack, DOTS requires reliable delivery of the this information and does not provide any special means for ensuring timely delivery of it during an attack. In

practice, this means that DOTS deployments SHOULD NOT rely on such information being exchanged during a DDoS attack.

2.1. DOTS Operations

The scope of DOTS is focused on the signaling and data exchange between the DOTS client and DOTS server. DOTS does not prescribe any specific deployment models, however DOTS is designed with some specific requirements around the different DOTS agents and their relationships.

First of all, a DOTS agent belongs to an domain, and that domain has an identity which can be authenticated and authorized. DOTS agents communicate with each other over a mutually authenticated signal channel and data channel. However, before they can do so, a service relationship needs to be established between them. The details and means by which this is done is outside the scope of DOTS, however an example would be for an enterprise A (DOTS client) to sign up for DDoS service from provider B (DOTS server). This would establish a (service) relationship between the two that enables enterprise A's DOTS client to establish a signal channel with provider B's DOTS server. A and B will authenticate each other, and B can verify that A is authorized for its service.

From an operational and design point of view, DOTS assumes that the above relationship is established prior to a request for DDoS attack mitigation. In particular, it is assumed that bi-directional communication is possible at this time between the DOTS client and DOTS server. Furthermore, it is assumed that additional service provisioning, configuration and information exchange can be performed by use of the data channel, if operationally required. It is not until this point that the mitigation service is available for use.

Once the mutually authenticated signal channel has been established, it will remain in place. This is done to increase the likelihood that the DOTS client can signal the DOTS server for help when the attack target is being flooded, and similarly raise the probability that DOTS server signals reach the client regardless of inbound link congestion. This does not necessarily imply that the attack target and the DOTS client have to be co-located in the same administrative domain, but it is expected to be a common scenario.

DDoS mitigation service with the help of an upstream mitigator will often involve some form of traffic redirection whereby traffic destined for the attack target is diverted towards the mitigator, e.g. by use of BGP [RFC4271] or DNS [RFC1034]. The mitigator in turn inspects and scrubs the traffic, and forwards the resulting (hopefully non-attack) traffic to the attack target. Thus, when a

DOTS server receives an attack mitigation request from a DOTS client, it can be viewed as a way of causing traffic redirection for the attack target indicated.

DOTS relies on mutual authentication and the pre-established service relationship between the DOTS client's domain and the DOTS server's domain to provide basic authorization. The DOTS server SHOULD enforce additional authorization mechanisms to restrict the mitigation scope a DOTS client can request, but such authorization mechanisms are deployment-specific.

Although co-location of DOTS server and mitigator within the same domain is expected to be a common deployment model, it is assumed that operators may require alternative models. Nothing in this document precludes such alternatives.

2.2. Components

2.2.1. DOTS Client

A DOTS client is a DOTS agent from which requests for help coordinating attack response originate. The requests may be in response to an active, ongoing attack against a target in the DOTS client's domain, but no active attack is required for a DOTS client to request help. Local operators may wish to have upstream mitigators in the network path for an indefinite period, and are restricted only by business relationships when it comes to duration and scope of requested mitigation.

The DOTS client requests attack response coordination from a DOTS server over the signal channel, including in the request the DOTS client's desired mitigation scoping, as described in [I-D.ietf-dots-requirements]. The actual mitigation scope and countermeasures used in response to the attack are up to the DOTS server and Mitigator operators, as the DOTS client may have a narrow perspective on the ongoing attack. As such, the DOTS client's request for mitigation should be considered advisory: guarantees of DOTS server availability or mitigation capacity constitute service level agreements and are out of scope for this document.

The DOTS client adjusts mitigation scope and provides available attack details at the direction of its local operator. Such direction may involve manual or automated adjustments in response to feedback from the DOTS server.

To provide a metric of signal health and distinguish an idle signaling session from a disconnected or defunct session, the DOTS client sends a heartbeat over the signal channel to maintain its half

of the signaling session. The DOTS client similarly expects a heartbeat from the DOTS server, and MAY consider a signaling session terminated in the extended absence of a DOTS server heartbeat.

2.2.2. DOTS Server

A DOTS server is a DOTS agent capable of receiving, processing and possibly acting on requests for help coordinating attack response from one or more DOTS clients. The DOTS server authenticates and authorizes DOTS clients as described in Signaling Sessions below, and maintains signaling session state, tracking requests for mitigation, reporting on the status of active mitigations, and terminating signaling sessions in the extended absence of a client heartbeat or when a session times out.

Assuming the preconditions discussed below exist, a DOTS client maintaining an active signaling session with a DOTS server may reasonably expect some level of mitigation in response to a request for coordinated attack response.

The DOTS server enforces authorization of DOTS clients' signals for mitigation. The mechanism of enforcement is not in scope for this document, but is expected to restrict requested mitigation scope to addresses, prefixes, and/or services owned by the DOTS client's administrative domain, such that a DOTS client from one domain is not able to influence the network path to another domain. A DOTS server MUST reject requests for mitigation of resources not owned by the requesting DOTS client's administrative domain. A DOTS server MAY also refuse a DOTS client's mitigation request for arbitrary reasons, within any limits imposed by business or service level agreements between client and server domains. If a DOTS server refuses a DOTS client's request for mitigation, the DOTS server SHOULD include the refusal reason in the server signal sent to the client.

A DOTS server is in regular contact with one or more mitigators. If a DOTS server accepts a DOTS client's request for help, the DOTS server forwards a translated form of that request to the mitigator or mitigators responsible for scrubbing attack traffic. Note that the form of the translated request passed from the DOTS server to the mitigator is not in scope: it may be as simple as an alert to mitigator operators, or highly automated using vendor or open application programming interfaces supported by the mitigator. The DOTS server MUST report the actual scope of any mitigation enabled on behalf of a client.

The DOTS server SHOULD retrieve available metrics for any mitigations activated on behalf of a DOTS client, and SHOULD include them in

server signals sent to the DOTS client originating the request for mitigation.

To provide a metric of signal health and distinguish an idle signaling session from a disconnected or defunct session, the DOTS server sends a heartbeat over the signal channel to maintain its half of the signaling session. The DOTS server similarly expects a heartbeat from the DOTS client, and MAY consider a signaling session terminated in the extended absence of a DOTS client heartbeat.

2.2.3. DOTS Gateway

Traditional client to server relationships may be expanded by chaining DOTS sessions. This chaining is enabled through "logical concatenation" [RFC7092] of a DOTS server and a DOTS client, resulting in an application analogous to the SIP logical entity of a Back-to-Back User Agent (B2BUA) [RFC3261]. The term DOTS gateway will be used here and the following text will describe some interactions in relation to this application.

A DOTS gateway may be deployed client-side, server-side or both. The gateway may terminate multiple discrete client connections and may aggregate these into a single or multiple DOTS signaling sessions.

The DOTS gateway will appear as a server to its downstream agents and as a client to its upstream agents, a functional concatenation of the DOTS client and server roles, as depicted in Figure 4:

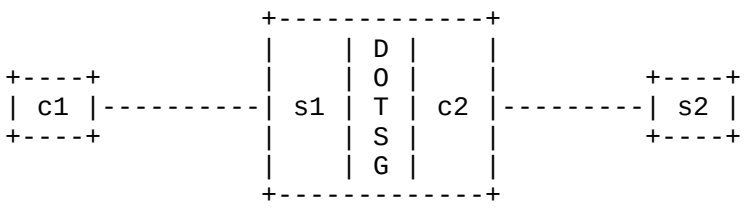


Figure 4: DOTS gateway

The DOTS gateway performs full stack DOTS session termination and reorigination between its client and server side. The details of how this is achieved are implementation specific. The DOTS protocol does not include any special features related to DOTS gateways, and hence from a DOTS perspective, whenever a DOTS gateway is present, the DOTS session simply terminates/originates there.

2.3. DOTS Agent Relationships

So far, we have only considered a relatively simple scenario of a single DOTS client associated with a single DOTS server, however DOTS supports more advanced relationships.

A DOTS server may be associated with one or more DOTS clients, and those DOTS clients may belong to different domains. An example scenario is a mitigation provider serving multiple attack targets (Figure 5):

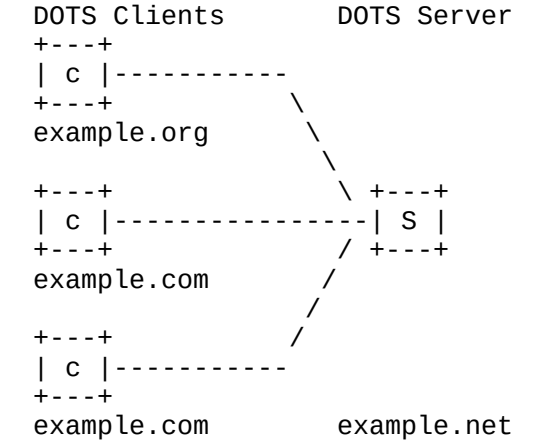


Figure 5: DOTS server with multiple clients

A DOTS client may be associated with one or more DOTS servers, and those DOTS servers may belong to different domains. This may be to ensure high availability or co-ordinate mitigation with more than one directly connected ISP. An example scenario is for an enterprise to have DDoS mitigation service from multiple providers, as shown in Figure 6 below. Operational considerations relating to co-ordinating multiple provider responses are beyond the scope of DOTS.

[[EDITOR'S NOTE: we request working group feedback and discussion of operational considerations relating to coordinating multiple provider responses to a mitigation request.]]

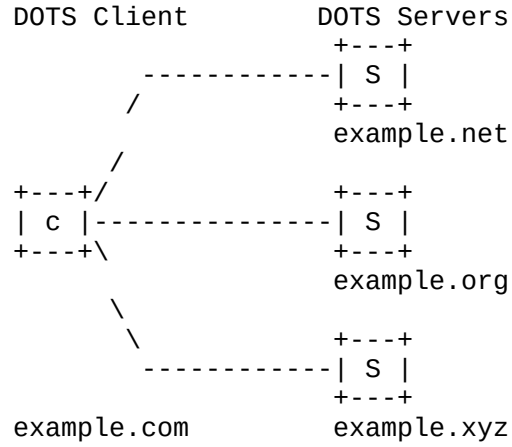


Figure 6: Multi-Homed DOTS Client

2.3.1. Gatewayed signaling

As discussed above in Section 2.2.3, a DOTS gateway is a logical function chaining signaling sessions through concatenation of a DOTS server and DOTS client.

An example scenario, as shown in Figure 7 and Figure 8 below, is for an enterprise to have deployed multiple DOTS capable devices which are able to signal intra-domain using TCP [RFC0793] on un-congested links to a DOTS gateway which may then transform these to a UDP [RFC0768] transport inter-domain where connection oriented transports may degrade; this applies to the signal channel only, as the data channel requires a connection-oriented transport. The relationship between the gateway and its upstream agents is opaque to the initial clients.

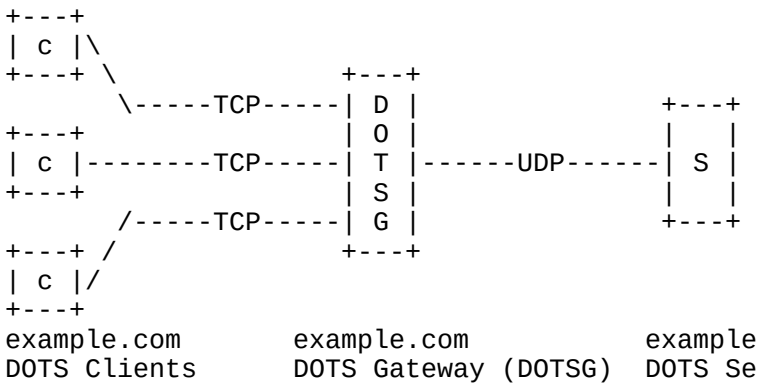


Figure 7: Client-Side Gateway with Aggregation

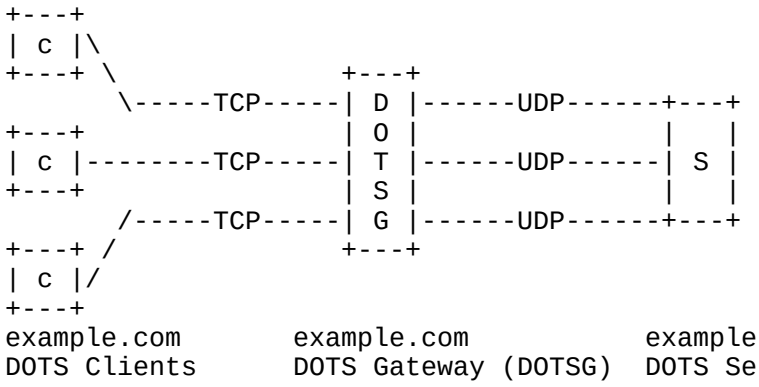


Figure 8: Client-Side Gateway without Aggregation

This may similarly be deployed in the inverse scenario where the gateway resides in the server-side domain and may be used to terminate and/or aggregate multiple clients to single transport as shown in figures Figure 9 and Figure 10 below.

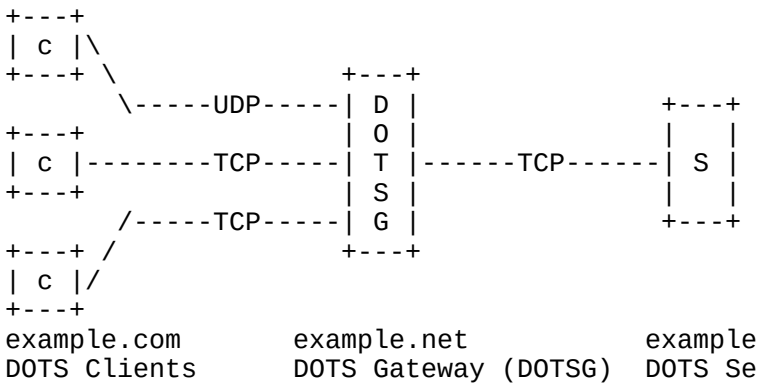


Figure 9: Server-Side Gateway with Aggregation

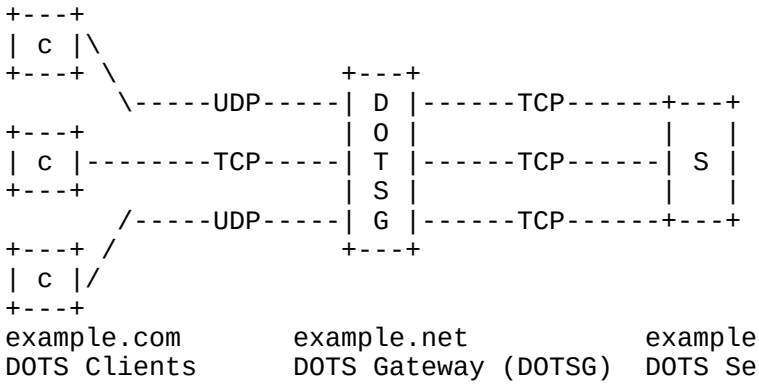


Figure 10: Server-Side Gateway without Aggregation

3. Concepts

3.1. Signaling Sessions

In order for DOTS to be effective as a vehicle for DDoS mitigation requests, one or more DOTS clients must establish ongoing communication with one or more DOTS servers. While the preconditions for enabling DOTS in or among network domains may also involve business relationships, service level agreements, or other formal or informal understandings between network operators, such considerations are out of scope for this document.

An established communication layer between DOTS agents is a Signaling Session. At its most basic, for a DOTS signaling session to exist both signal channel and data channel must be functioning between DOTS agents. That is, under nominal network conditions, signals actively

sent from a DOTS client are received by the specific DOTS server intended by the client, and vice versa.

3.1.1. Preconditions

Prior to establishing a signaling session between agents, the owners of the networks, domains, services or applications involved are assumed to have agreed upon the terms of the relationship involved. Such agreements are out of scope for this document, but must be in place for a functional DOTS architecture.

It is assumed that as part of any DOTS service agreement, the DOTS client is provided with all data and metadata required to establish communication with the DOTS server. Such data and metadata would include any cryptographic information necessary to meet the message confidentiality, integrity and authenticity requirement in [I-D.ietf-dots-requirements], and might also include the pool of DOTS server addresses and ports the DOTS client should use for signal and data channel messaging.

3.1.2. Establishing the Signaling Session

With the required business or service agreements in place, the DOTS client initiates a signal session by contacting the DOTS server over the signal channel and the data channel. To allow for DOTS service flexibility, neither the order of contact nor the time interval between channel creations is specified. A DOTS client MAY establish signal channel first, and then data channel, or vice versa.

The methods by which a DOTS client receives the address and associated service details of the DOTS server are not prescribed by this document. For example, a DOTS client may be directly configured to use a specific DOTS server address and port, and directly provided with any data necessary to satisfy the Peer Mutual Authentication requirement in [I-D.ietf-dots-requirements], such as symmetric or asymmetric keys, usernames and passwords, etc. All configuration and authentication information in this scenario is provided out-of-band by the domain operating the DOTS server.

At the other extreme, the architecture in this document allows for a form of DOTS client auto-provisioning. For example, the domain operating the DOTS server or servers might provide the client domain only with symmetric or asymmetric keys to authenticate the provisioned DOTS clients. Only the keys would then be directly configured on DOTS clients, but the remaining configuration required to provision the DOTS clients could be learned through mechanisms similar to DNS SRV [RFC2782] or DNS Service Discovery [RFC6763].

The DOTS client SHOULD successfully authenticate and exchange messages with the DOTS server over both signal and data channel as soon as possible to confirm that both channels are operational.

Once the DOTS client begins receiving DOTS server signals, the signaling session is active. At any time during the signaling session, the DOTS client MAY use the data channel to adjust initial configuration, manage black- and white-listed prefixes or addresses, leverage vendor-specific extensions, and so on. Note that unlike the signal channel, there is no requirement that the data channel remain operational in attack conditions (See Data Channel Requirements, [I-D.ietf-dots-requirements]).

3.1.3. Maintaining the Signaling Session

DOTS clients and servers periodically send heartbeats to each other over the signal channel, per Operational Requirements discussed in [I-D.ietf-dots-requirements]. DOTS agent operators SHOULD configure the heartbeat interval such that the frequency does not lead to accidental denials of service due to the overwhelming number of heartbeats a DOTS agent must field.

Either DOTS agent may consider a signaling session terminated in the extended absence of a heartbeat from its peer agent. The period of that absence will be established in the protocol definition.

3.2. Modes of Signaling

This section examines the modes of signaling between agents in a DOTS architecture.

3.2.1. Direct Signaling

A signaling session may take the form of direct signaling between the DOTS clients and servers, as shown in Figure 11 below:

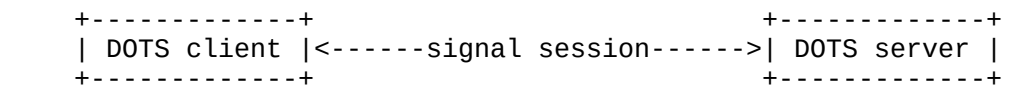


Figure 11: Direct Signaling

In a direct signaling session, DOTS client and server are communicating directly. A direct signaling session MAY exist inter- or intra-domain. The signaling session is abstracted from the underlying networks or network elements the signals traverse: in a direct signaling session, the DOTS client and server are logically peer DOTS agents.

3.2.2. Redirected Signaling

In certain circumstances, a DOTS server may want to redirect a DOTS client to an alternative DOTS server for a signaling session. Such circumstances include but are not limited to:

- Maximum number of signaling sessions with clients has been reached;
- Mitigation capacity exhaustion in the Mitigator with which the specific DOTS server is communicating;
- Mitigator outage or other downtime, such as scheduled maintenance;
- Scheduled DOTS server maintenance;
- Scheduled modifications to the network path between DOTS server and DOTS client.

A basic redirected signaling session resembles the following, as shown in Figure 12:

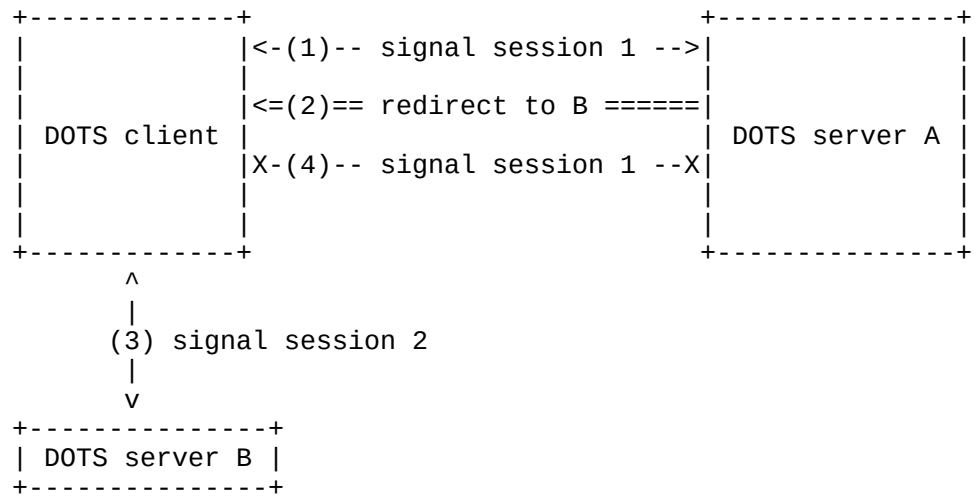


Figure 12: Redirected Signaling

1. Previously established signaling session 1 exists between a DOTS client and DOTS server with address A.
2. DOTS server A sends a server signal redirecting the client to DOTS server B.

3. If the DOTS client does not already have a separate signaling session with the redirection target, the DOTS client initiates and establishes a signaling session with DOTS server B as described above.

4. Having redirected the DOTS client, DOTS server A ceases sending server signals. The DOTS client likewise stops sending client signals to DOTS server A. Signal session 1 is terminated.

[[EDITOR'S NOTE: we request working group feedback and discussion of the need for redirected signaling.]]

3.2.3. Recursive Signaling

DOTS is centered around improving the speed and efficiency of coordinated response to DDoS attacks. One scenario not yet discussed involves coordination among federated domains operating DOTS servers and mitigators.

In the course of normal DOTS operations, a DOTS client communicates the need for mitigation to a DOTS server, and that server initiates mitigation on a mitigator with which the server has an established service relationship. The operator of the mitigator may in turn monitor mitigation performance and capacity, as the attack being mitigated may grow in severity beyond the mitigating domain's capabilities.

The operator of the mitigator has limited options in the event a DOTS client-requested mitigation is being overwhelmed by the severity of the attack. Out-of-scope business or service level agreements may permit the mitigating domain to drop the mitigation and let attack traffic flow unchecked to the target, but this is only encourages attack escalation. In the case where the mitigating domain is the upstream service provider for the attack target, this may mean the mitigating domain and its other services and users continue to suffer the incidental effects of the attack.

A recursive signaling model as shown in Figure 13 below offers an alternative. In a variation of the primary use case "Successful Automatic or Operator-Assisted CPE or PE Mitigators Request Upstream DDoS Mitigation Services" described in [I-D.ietf-dots-use-cases], an domain operating a DOTS server and mitigation also operates a DOTS client. This DOTS client has an established signaling session with a DOTS server belonging to a separate administrative domain.

With these preconditions in place, the operator of the mitigator being overwhelmed or otherwise performing inadequately may request mitigation for the attack target from this separate DOTS-aware

domain. Such a request recurses the originating mitigation request to the secondary DOTS server, in the hope of building a cumulative mitigation against the attack:

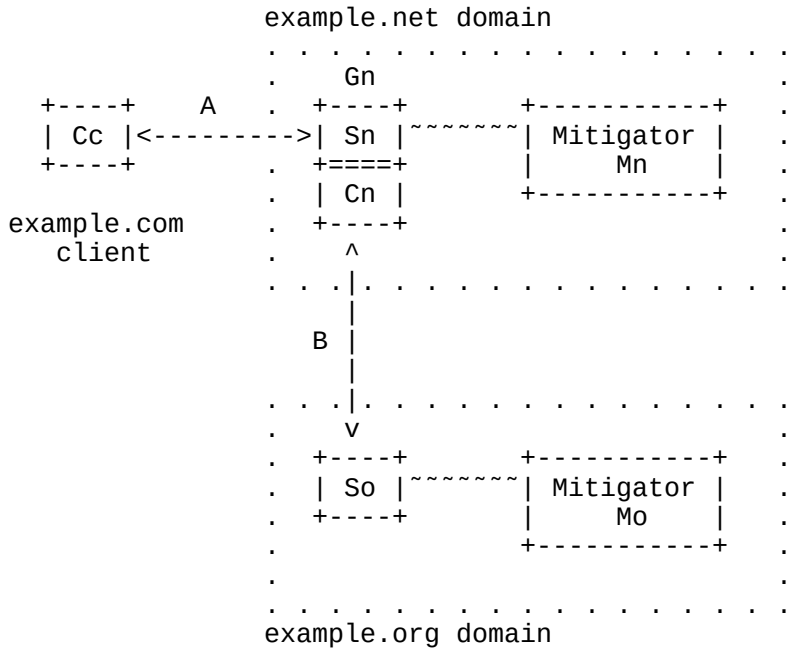


Figure 13: Recursive Signaling

In Figure 13 above, client Cc signals a request for mitigation across inter-domain signaling session A to the DOTS server Sn belonging to the example.net domain. DOTS server Sn enables mitigation on mitigator Mn. DOTS server Sn is half of DOTS gateway Gn, being deployed logically back-to-back with DOTS client Cn, which has pre-existing inter-domain signaling session B with the DOTS server So belonging to the example.org domain. At any point, DOTS server Sn MAY recurse an on-going mitigation request through DOTS client Cn to DOTS server So, in the expectation that mitigator Mo will be activated to aid in the defense of the attack target.

Recursive signaling is opaque to the DOTS client. To maximize mitigation visibility to the DOTS client, however, the recursing domain SHOULD provide recursed mitigation feedback in signals reporting on mitigation status to the DOTS client. For example, the recursing domain’s mitigator should incorporate into mitigation status messages available metrics such as dropped packet or byte counts from the recursed mitigation.

DOTS clients involved in recursive signaling MUST be able to withdraw requests for mitigation without warning or justification, per [I-D.ietf-dots-requirements].

Operators recursing mitigation requests MAY maintain the recursed mitigation for a brief, protocol-defined period in the event the DOTS client originating the mitigation withdraws its request for help, as per the discussion of managing mitigation toggling in the operational requirements ([I-D.ietf-dots-requirements]). Service or business agreements between recursing domains are not in scope for this document.

[[EDITOR'S NOTE: Recursive signaling raises questions about operational and data privacy, as well as what level of visibility a client has into the recursed mitigation. We ask the working group for feedback and additional discussion of these issues to help settle the way forward.]]

3.3. Triggering Requests for Mitigation

[I-D.ietf-dots-requirements] places no limitation on the circumstances in which a DOTS client operator may request mitigation, nor does it demand justification for any mitigation request, thereby reserving operational control over DDoS defense for the domain requesting mitigation. This architecture likewise does not prescribe the network conditions and mechanisms triggering a mitigation request from a DOTS client.

However, considering selected possible mitigation triggers from an architectural perspective offers a model for alternative or unanticipated triggers for DOTS deployments. In all cases, what network conditions merit a mitigation request are at the discretion of the DOTS client operator.

The interfaces required to trigger the mitigation request in the following scenarios are implementation-specific.

3.3.1. Manual Mitigation Request

A DOTS client operator may manually prepare a request for mitigation, including scope and duration, and manually instruct the DOTS client to send the mitigation request to the DOTS server. In context, a manual request is a request directly issued by the operator without automated decision-making performed by a device interacting with the DOTS client. Modes of manual mitigation requests include an operator entering a command into a text interface, or directly interacting with a graphical interface to send the request.

An operator might do this, for example, in response to notice of an attack delivered by attack detection equipment or software, and the alerting detector lacks interfaces or is not configured to use available interfaces to translate the alert to a mitigation request automatically.

In a variation of the above scenario, the operator may have preconfigured on the DOTS client mitigation request for various resources in the operator's domain. When notified of an attack, the DOTS client operator manually instructs the DOTS client to send the preconfigured mitigation request for the resources under attack.

A further variant involves recursive signaling, as described in Section 3.2.3. The DOTS client in this case is the second half of a DOTS gateway (back-to-back DOTS server and client). As in the previous scenario, the scope and duration of the mitigation request are pre-existing, but in this case are derived from the mitigation request received from a downstream DOTS client by the DOTS server. Assuming the preconditions required by Section 3.2.3 are in place, the DOTS gateway operator may at any time manually request mitigation from an upstream DOTS server, sending a mitigation request derived from the downstream DOTS client's request.

The motivations for a DOTS client operator to request mitigation manually are not prescribed by this architecture, but are expected to include some of the following:

- Notice of an attack delivered via e-mail or alternative messaging
- Notice of an attack delivered via phone call
- Notice of an attack delivered through the interface(s) of networking monitoring software deployed in the operator's domain
- Manual monitoring of network behavior through network monitoring software

3.3.2. Automated Threshold-Based Mitigation Request

Unlike manual mitigation requests, which depend entirely on the DOTS client operator's capacity to react with speed and accuracy to every detected or detectable attack, mitigation requests triggered by detected attack thresholds reduce the operational burden on the DOTS client operator, and minimize the latency between attack detection and the start of mitigation.

Mitigation requests are triggered in this scenario by violations of operator-specified attack thresholds. Attack detection is

deployment-specific, and not constrained by this architecture. Similarly the specifics of a threshold are left to the discretion of the operator, though common threshold types include the following:

- Detected attack exceeding a rate in packets per second (pps).
- Detected attack exceeding a rate in bytes per second (bps).
- Detected resource exhaustion in an attack target.
- Detected resource exhaustion in the local domain's mitigator.
- Number of open connections to an attack target.
- Number of attack sources in a given attack.
- Number of active attacks against targets in the operator's domain.
- Thresholds developed through arbitrary statistical analysis or deep learning techniques.

When automated threshold-based mitigation requests are enabled, violations of any of the above thresholds, or any additional operator-defined threshold, will trigger a mitigation request from the DOTS client to the DOTS server. The interfaces between the application detecting the threshold violation and the DOTS client are implementation-specific.

3.3.3. Automated Mitigation on Loss of Signal

To maintain a signaling session, the DOTS client and the DOTS server exchange regular but infrequent messages across the signaling channel. In the absence of an attack, the probability of message loss in the signaling channel should be extremely low. Under attack conditions, however, some signal loss may be anticipated as attack traffic congests the link, depending on the attack type.

While [I-D.ietf-dots-requirements] specifies the DOTS protocol be robust when signaling under attack conditions, there are nevertheless scenarios in which the DOTS signal is lost in spite of protocol best efforts. To handle such scenarios, a DOTS client operator may configure the signaling session to trigger mitigation when the DOTS server ceases receiving DOTS client signals (or vice versa) beyond the miss count or period permitted by the protocol.

The impact of mitigating due to loss of signal in either direction must be considered carefully before enabling it. Signal loss is not caused by links congested with attack traffic alone, and as such

mitigation requests triggered by signal channel degradation in either direction may incur unnecessary costs, in network performance and operational expense alike.

4. Security Considerations

This section describes identified security considerations for the DOTS architecture.

DOTS is at risk from three primary attack vectors: agent impersonation, traffic injection and signal blocking. These vectors may be exploited individually or in concert by an attacker to confuse, disable, take information from, or otherwise inhibit DOTS agents.

Any attacker with the ability to impersonate a legitimate client or server or, indeed, inject false messages into the stream may potentially trigger/withdraw traffic redirection, trigger/cancel mitigation activities or subvert black/whitelists. From an architectural standpoint, operators SHOULD ensure best current practices for secure communication are observed for data and signal channel confidentiality, integrity and authenticity. Care must be taken to ensure transmission is protected by appropriately secure means, reducing attack surface by exposing only the minimal required services or interfaces. Similarly, received data at rest SHOULD be stored with a satisfactory degree of security.

As many mitigation systems employ diversion to scrub attack traffic, operators of DOTS agents SHOULD ensure signaling sessions are resistant to Man-in-the-Middle (MitM) attacks. An attacker with control of a DOTS client may negatively influence network traffic by requesting and withdrawing requests for mitigation for particular prefixes, leading to route or DNS flapping.

Any attack targeting the availability of DOTS servers may disrupt the ability of the system to receive and process DOTS signals resulting in failure to fulfill a mitigation request. DOTS agents SHOULD be given adequate protections, again in accordance with best current practices for network and host security.

5. Acknowledgments

Thanks to Matt Richardson and Med Boucadair for their comments and suggestions.

6. Change Log

2016-03-18 Initial revision

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

7.2. Informative References

[I-D.ietf-dots-requirements] Mortensen, A., Moskowitz, R., and T. Reddy, "Distributed Denial of Service (DDoS) Open Threat Signaling Requirements", draft-ietf-dots-requirements-01 (work in progress), March 2016.

[I-D.ietf-dots-use-cases] Dobbins, R., Fouant, S., Migault, D., Moskowitz, R., Teague, N., and L. Xia, "Use cases for DDoS Open Threat Signaling", draft-ietf-dots-use-cases-01 (work in progress), March 2016.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <http://www.rfc-editor.org/info/rfc768>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <http://www.rfc-editor.org/info/rfc2782>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <http://www.rfc-editor.org/info/rfc4271>.

[RFC4732] Handley, M., Ed., Rescorla, E., Ed., and IAB, "Internet Denial-of-Service Considerations", RFC 4732, DOI 10.17487/RFC4732, December 2006, <http://www.rfc-editor.org/info/rfc4732>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <http://www.rfc-editor.org/info/rfc6763>.

[RFC7092] Kaplan, H. and V. Pascual, "A Taxonomy of Session Initiation Protocol (SIP) Back-to-Back User Agents", RFC 7092, DOI 10.17487/RFC7092, December 2013, <http://www.rfc-editor.org/info/rfc7092>.

Authors' Addresses

Andrew Mortensen
Arbor Networks, Inc.
2727 S. State St
Ann Arbor, MI 48104
United States

EMail: amortensen@arbor.net

Flemming Andreasen
Cisco Systems, Inc.
United States

EMail: fandreas@cisco.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

EMail: tireddy@cisco.com

Christopher Gray
Comcast, Inc.
United States

EMail: Christopher_Gray3@cable.comcast.com

Rich Compton
Charter Communications, Inc.

EMail: Rich.Compton@charter.com

Nik Teague
Verisign, Inc.
United States

EMail: nteague@verisign.com

DOTS
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

A. Mortensen
Arbor Networks, Inc.
R. Moskowitz
HTT Consulting
T. Reddy
Cisco Systems, Inc.
July 08, 2016

Distributed Denial of Service (DDoS) Open Threat Signaling Requirements
draft-ietf-dots-requirements-02

Abstract

This document defines the requirements for the Distributed Denial of Service (DDoS) Open Threat Signaling (DOTS) protocols coordinating attack response against DDoS attacks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2017.

Copyright Notice

Table of Contents

1. Introduction

1.1. Context and Motivation

Distributed Denial of Service (DDoS) attacks continue to plague networks around the globe, from Tier-1 service providers on down to enterprises and small businesses. Attack scale and frequency similarly have continued to increase, in part as a result of software vulnerabilities leading to reflection and amplification attacks. Once staggering attack traffic volume is now the norm, and the impact of larger-scale attacks attract the attention of international press agencies.

The greater impact of contemporary DDoS attacks has led to increased focus on coordinated attack response. Many institutions and enterprises lack the resources or expertise to operate on-premise attack mitigation solutions themselves, or simply find themselves constrained by local bandwidth limitations. To address such gaps,

security service providers have begun to offer on-demand traffic scrubbing services, which aim to separate the DDoS traffic from legitimate traffic and forward only the latter. Today each such service offers its own interface for subscribers to request attack mitigation, tying subscribers to proprietary implementations while also limiting the subset of network elements capable of participating in the attack response. As a result of incompatibility across services, attack responses may be fragmentary or otherwise incomplete, leaving key players in the attack path unable to assist in the defense.

The lack of a common method to coordinate a real-time response among involved actors and network domains inhibits the speed and effectiveness of DDoS attack mitigation. This document describes the required characteristics of a DOTS protocol enabling requests for DDoS attack mitigation, reducing attack impact and leading to more efficient defensive strategies.

DOTS communicates the need for defensive action in anticipation of or in response to an attack, but does not dictate the form any defensive action takes. DOTS supplements calls for help with pertinent details about the detected attack, allowing entities participating in DOTS to form ad hoc, adaptive alliances against DDoS attacks as described in the DOTS use cases [I-D.ietf-dots-use-cases]. The requirements in this document are derived from those use cases and [I-D.ietf-dots-architecture].

1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document adopts the following terms:

DDoS: A distributed denial-of-service attack, in which traffic originating from multiple sources are directed at a target on a network. DDoS attacks are intended to cause a negative impact on the availability of servers, services, applications, and/or other functionality of an attack target. Denial-of-service considerations are discussed in detail in [RFC4732].

DDoS attack target: A network connected entity with a finite set of resources, such as network bandwidth, memory or CPU, that is the focus of a DDoS attack. Potential targets include servers, services and applications.

DDoS attack telemetry: Collected behavioral characteristics defining the nature of a DDoS attack. This document makes no assumptions regarding telemetry collection methodology.

Countermeasure: An action or set of actions taken to recognize and filter out DDoS attack traffic while passing legitimate traffic to the attack target.

Mitigation: A set of countermeasures enforced against traffic destined for the target or targets of a detected or reported DDoS attack, where countermeasure enforcement is managed by an entity in the network path between attack sources and the attack target. Mitigation methodology is out of scope for this document.

Mitigator: An entity, typically a network element, capable of performing mitigation of a detected or reported DDoS attack. For the purposes of this document, this entity is a black box capable of mitigation, making no assumptions about availability or design of countermeasures, nor about the programmable interface between this entity and other network elements. The mitigator and DOTS server are assumed to belong to the same administrative entity.

DOTS client: A DOTS-aware software module responsible for requesting attack response coordination with other DOTS-aware elements.

DOTS server: A DOTS-aware software module handling and responding to messages from DOTS clients. The DOTS server SHOULD enable mitigation on behalf of the DOTS client, if requested, by communicating the DOTS client’s request to the mitigator and returning selected mitigator feedback to the requesting DOTS client. A DOTS server MAY also be a mitigator.

DOTS agent: Any DOTS-aware software module capable of participating in a DOTS siganling session.

DOTS gateway: A logical DOTS agent resulting from the logical concatenation of a DOTS server and a DOTS client, analogous to a SIP Back-to-Back User Agent (B2BUA) [RFC3261]. DOTS gateways are discussed in detail in [I-D.ietf-dots-architecture].

Signal channel: A bidirectional, mutually authenticated communication channel between DOTS agents characterized by resilience even in conditions leading to severe packet loss, such as a volumetric DDoS attack causing network congestion.

DOTS signal: A concise authenticated status/control message transmitted between DOTS agents, used to indicate client’s need

for mitigation, as well as to convey the status of any requested mitigation.

Heartbeat: A message transmitted between DOTS agents over the signal channel, used as a keep-alive and to measure peer health.

Client signal: A message sent from a DOTS client to a DOTS server over the signal channel, indicating the DOTS client's need for mitigation, as well as the scope of any requested mitigation, optionally including additional attack details to supplement server-initiated mitigation.

Server signal: A message sent from a DOTS server to a DOTS client over the signal channel. Note that a server signal is not a response to client signal, but a DOTS server-initiated status message sent to DOTS clients with which the server has established signaling sessions.

Data channel: A secure communication layer between DOTS clients and DOTS servers used for infrequent bulk exchange of data not easily or appropriately communicated through the signal channel under attack conditions.

Filter: A policy matching a network traffic flow or set of flows and rate-limiting or discarding matching traffic.

Blacklist: A filter list of addresses, prefixes and/or other identifiers indicating sources from which traffic should be blocked, regardless of traffic content.

Whitelist: A list of addresses, prefixes and/or other identifiers from indicating sources from which traffic should always be allowed, regardless of contradictory data gleaned in a detected attack.

Multi-homed DOTS client: A DOTS client exchanging messages with multiple DOTS servers, each in a separate administrative domain.

2. Requirements

This section describes the required features and characteristics of the DOTS protocol.

DOTS is an advisory protocol. An active DDoS attack against the entity controlling the DOTS client need not be present before establishing DOTS communication between DOTS agents. Indeed, establishing a relationship with peer DOTS agents during normal network conditions provides the foundation for more rapid attack

response against future attacks, as all interactions setting up DOTS, including any business or service level agreements, are already complete.

DOTS must at a minimum make it possible for a DOTS client to request a DOTS server's aid in mounting a coordinated defense against a suspected attack, signaling within or between domains as requested by local operators. DOTS clients should similarly be able to withdraw aid requests. DOTS requires no justification from DOTS clients for requests for help, nor do DOTS clients need to justify withdrawing help requests: the decision is local to the DOTS clients' domain. Regular feedback between DOTS clients and DOTS server supplement the defensive alliance by maintaining a common understanding of DOTS peer health and activity. Bidirectional communication between DOTS clients and DOTS servers is therefore critical.

Yet DOTS must also work with a set of competing operational goals. On the one hand, the protocol must be resilient under extremely hostile network conditions, providing continued contact between DOTS agents even as attack traffic saturates the link. Such resiliency may be developed several ways, but characteristics such as small message size, asynchronous, redundant message delivery and minimal connection overhead (when possible given local network policy) will tend to contribute to the robustness demanded by a viable DOTS protocol. Operators of peer DOTS-enabled domains may enable quality- or class-of-service traffic tagging to increase the probability of successful DOTS signal delivery, but DOTS requires no such policies be in place. The DOTS solution indeed must be viable especially in their absence.

On the other hand, DOTS must include protections ensuring message confidentiality, integrity and authenticity to keep the protocol from becoming another vector for the very attacks it's meant to help fight off. DOTS clients must be able to authenticate DOTS servers, and vice versa, for DOTS to operate safely, meaning the DOTS agents must have a way to negotiate and agree upon the terms of protocol security. Attacks against the transport protocol should not offer a means of attack against the message confidentiality, integrity and authenticity.

The DOTS server and client must also have some common method of defining the scope of any mitigation performed by the mitigator, as well as making adjustments to other commonly configurable features, such as listen ports, exchanging black- and white-lists, and so on.

Finally, DOTS should provide sufficient extensibility to meet local, vendor or future needs in coordinated attack defense, although this

consideration is necessarily superseded by the other operational requirements.

2.1. General Requirements

GEN-001 Extensibility: Protocols and data models developed as part of DOTS MUST be extensible in order to keep DOTS adaptable to operational and proprietary DDoS defenses. Future extensions MUST be backward compatible.

GEN-002 Resilience and Robustness: The signaling protocol MUST be designed to maximize the probability of signal delivery even under the severely constrained network conditions imposed by particular attack traffic. The protocol MUST be resilient, that is, continue operating despite message loss and out-of-order or redundant message delivery.

GEN-003 Bidirectionality: To support peer health detection, to maintain an open signal channel, and to increase the probability of signal delivery during attack, the signal channel MUST be bidirectional, with client and server transmitting signals to each other at regular intervals, regardless of any client request for mitigation. Unidirectional messages MUST be supported within the bidirectional signal channel to allow for unsolicited message delivery, enabling asynchronous notifications between agents.

GEN-004 Sub-MTU Message Size: To avoid message fragmentation and the consequently decreased probability of message delivery, signaling protocol message size MUST be kept under signaling path Maximum Transmission Unit (MTU), including the byte overhead of any encapsulation, transport headers, and transport- or message-level security.

GEN-005 Bulk Data Exchange: Infrequent bulk data exchange between DOTS agents can also significantly augment attack response coordination, permitting such tasks as population of black- or white-listed source addresses; address or prefix group aliasing; exchange of incident reports; and other hinting or configuration supplementing attack response.

As the resilience requirements for the DOTS signal channel mandate small signal message size, a separate, secure data channel utilizing an established reliable transport protocol MUST be used for bulk data exchange.

2.2. Operational Requirements

OP-001 Use of Common Transport Protocols: DOTS MUST operate over common widely deployed and standardized transport protocols. While the User Datagram Protocol (UDP) [RFC0768] SHOULD be used for the signal channel, the Transmission Control Protocol (TCP) [RFC0793] MAY be used if necessary due to network policy or middlebox capabilities or configurations. The data channel MUST use TCP; see Section 2.3 below.

OP-002 Session Health Monitoring: Peer DOTS agents MUST regularly send heartbeats to each other after mutual authentication in order to keep the DOTS session open. A session MUST be considered active until a DOTS agent explicitly ends the session, or either DOTS agent fails to receive heartbeats from the other after a mutually agreed upon timeout period has elapsed.

OP-003 Session Redirection: In order to increase DOTS operational flexibility and scalability, DOTS servers SHOULD be able to redirect DOTS clients to another DOTS server at any time. Due to the decreased probability of DOTS server signal delivery due to link congestion, it is RECOMMENDED DOTS servers avoid redirecting while mitigation is enabled during an active attack against a target in the DOTS client’s domain. Either the DOTS servers have to fate-share the security state, the client MUST have separate security state with each potential redirectable server, or be able to negotiate new state as part of redirection.

OP-004 Mitigation Status: DOTS MUST provide a means to report the status of an action requested by a DOTS client. In particular, DOTS clients MUST be able to request or withdraw a request for mitigation from the DOTS server. The DOTS server MUST acknowledge a DOTS client’s request to withdraw from coordinated attack response in subsequent signals, and MUST cease mitigation activity as quickly as possible. However, a DOTS client rapidly toggling active mitigation may result in undesirable side-effects for the network path, such as route or DNS [RFC1034] flapping. A DOTS server therefore MAY continue mitigating for a mutually negotiated period after receiving the DOTS client’s request to stop.

A server MAY refuse to engage in coordinated attack response with a client. To make the status of a client’s request clear, the server MUST indicate in server signals whether client-initiated mitigation is active. When a client-initiated mitigation is active, and threat handling details such as mitigation scope and statistics are available to the server, the server SHOULD include those details in server signals sent to the client. DOTS clients

SHOULD take mitigation statistics into account when deciding whether to request the DOTS server cease mitigation.

OP-005 Mitigation Lifetime: A DOTS client SHOULD indicate the desired lifetime of any mitigation requested from the DOTS server. As DDoS attack duration is unpredictable, the DOTS client SHOULD be able to extend mitigation lifetime with periodic renewed requests for help. When the mitigation lifetime comes to an end, the DOTS server SHOULD delay session termination for a protocol-defined grace period to allow for delivery of delayed mitigation renewals over the signal channel. After the grace period elapses, the DOTS server MAY terminate the session at any time.

If a DOTS client does not include a mitigation lifetime in requests for help sent to the DOTS server, the DOTS server will use a reasonable default as defined by the protocol. As above, the DOTS client MAY extend a current mitigation request’s lifetime trivially with renewed requests for help.

A DOTS client MAY also request an indefinite mitigation lifetime, enabling architectures in which the mitigator is always in the traffic path to the resources for which the DOTS client is requesting protection. DOTS servers MAY refuse such requests for any reason. The reasons themselves are not in scope.

OP-006 Mitigation Scope: DOTS clients MUST indicate the desired scope of any mitigation, for example by using Classless Internet Domain Routing (CIDR) [RFC1518],[RFC1519] prefixes, [RFC2373] for IPv6 [RFC2460] prefixes, the length/prefix convention established in the Border Gateway Protocol (BGP) [RFC4271], SIP URIs [RFC3261], E.164 numbers, DNS names, or by a resource group alias agreed upon with the server through the data channel.

If there is additional information available narrowing the scope of any requested attack response, such as targeted port range, protocol, or service, DOTS clients SHOULD include that information in client signals. DOTS clients MAY also include additional attack details. Such supplemental information is OPTIONAL, and DOTS servers MAY ignore it when enabling countermeasures on the mitigator.

As an active attack evolves, clients MUST be able to adjust as necessary the scope of requested mitigation by refining the scope of resources requiring mitigation.

OP-007 Mitigation Efficacy: When a mitigation request by a DOTS client is active, DOTS clients SHOULD transmit a metric of perceived mitigation efficacy to the DOTS server, per "Automatic

or Operator-Assisted CPE or PE Mitigators Request Upstream DDoS Mitigation Services" in [I-D.ietf-dots-use-cases]. DOTS servers MAY use the efficacy metric to adjust countermeasures activated on a mitigator on behalf of a DOTS client.

OP-008 Conflict Detection and Notification: Multiple DOTS clients controlled by a single administrative entity may send conflicting mitigation requests for pool of protected resources , as a result of misconfiguration, operator error, or compromised DOTS clients. DOTS servers attempting to honor conflicting requests may flap network route or DNS information, degrading the networks attempting to participate in attack response with the DOTS clients. DOTS servers SHALL detect such conflicting requests, and SHALL notify the DOTS clients in conflict. The notification SHOULD indicate the nature and scope of the conflict, for example, the overlapping prefix range in a conflicting mitigation request.

OP-009: Network Address Translator Traversal: The DOTS protocol MUST operate over networks in which Network Address Translation (NAT) is deployed. As UDP is the recommended transport for DOTS, all considerations in "Middlebox Traversal Guidelines" in [RFC5405] apply to DOTS. Regardless of transport, DOTS protocols MUST follow established best common practices (BCPs) for NAT traversal.

2.3. Data Channel Requirements

The data channel is intended to be used for bulk data exchanges between DOTS agents. Unlike the signal channel, which must operate nominally even when confronted with despite signal degradation due to packet loss, the data channel is not expected to be constructed to deal with attack conditions. As the primary function of the data channel is data exchange, a reliable transport is required in order for DOTS agents to detect data delivery success or failure.

The data channel must be extensible. We anticipate the data channel will be used for such purposes as configuration or resource discovery. For example, a DOTS client may submit to the DOTS server a collection of prefixes it wants to refer to by alias when requesting mitigation, to which the server would respond with a success status and the new prefix group alias, or an error status and message in the event the DOTS client’s data channel request failed. The transactional nature of such data exchanges suggests a separate set of requirements for the data channel, while the potentially sensitive content sent between DOTS agents requires extra precautions to ensure data privacy and authenticity.

DATA-001 Reliable transport: Messages sent over the data channel MUST be delivered reliably, in send order.

DATA-002 Data privacy and integrity: Transmissions over the data channel is likely to contain operationally or privacy-sensitive information or instructions from the remote DOTS agent. Theft or modification of data channel transmissions could lead to information leaks or malicious transactions on behalf of the sending agent (see Section 4 below). Consequently data sent over the data channel MUST be encrypted and authenticated using current industry best practices. DOTS servers MUST enable means to prevent leaking operationally or privacy-sensitive data. Although administrative entities participating in DOTS may detail what data may be revealed to third-party DOTS agents, such considerations are not in scope for this document.

DATA-003 Session configuration: To help meet the general and operational requirements in this document, DOTS servers MUST provide methods for DOTS client operators to configure DOTS session behavior using the data channel. DOTS server implementations MUST have mechanisms to configure the following:

* Acceptable signal lossiness, as described in GEN-002.

* Heartbeat intervals, as described in OP-002.

* Maximum mitigation lifetime, as described in OP-005.

* Resource identifiers, as described in OP-006.

DOTS server implementations MAY expose additional configurability. Additional configurability is implementation-specific.

DATA-004 Black- and whitelist management: DOTS servers SHOULD provide methods for DOTS clients to manage black- and white-lists of traffic destined for resources belonging to a client.

For example, a DOTS client should be able to create a black- or whitelist entry; retrieve a list of current entries from either list; update the content of either list; and delete entries as necessary.

How the DOTS server determines client ownership of address space is not in scope.

2.4. Security requirements

DOTS must operate within a particularly strict security context, as an insufficiently protected signal or data channel may be subject to abuse, enabling or supplementing the very attacks DOTS purports to mitigate.

SEC-001 Peer Mutual Authentication: DOTS agents MUST authenticate each other before a DOTS session is considered valid. The method of authentication is not specified, but should follow current industry best practices with respect to any cryptographic mechanisms to authenticate the remote peer.

SEC-002 Message Confidentiality, Integrity and Authenticity: DOTS protocols MUST take steps to protect the confidentiality, integrity and authenticity of messages sent between client and server. While specific transport- and message-level security options are not specified, the protocols MUST follow current industry best practices for encryption and message authentication.

In order for DOTS protocols to remain secure despite advancements in cryptanalysis and traffic analysis, DOTS agents MUST be able to negotiate the terms and mechanisms of protocol security, subject to the interoperability and signal message size requirements above.

SEC-003 Message Replay Protection: In order to prevent a passive attacker from capturing and replaying old messages, DOTS protocols MUST provide a method for replay detection.

2.5. Data Model Requirements

The value of DOTS is in standardizing a mechanism to permit elements, networks or domains under or under threat of DDoS attack to request aid mitigating the effects of any such attack. A well-structured DOTS data model is therefore critical to the development of a successful DOTS protocol.

DM-001: Structure: The data model structure for the DOTS protocol may be described by a single module, or be divided into related collections of hierarchical modules and sub-modules. If the data model structure is split across modules, those distinct modules MUST allow references to describe the overall data model’s structural dependencies.

DM-002: Versioning: To ensure interoperability between DOTS protocol implementations, data models MUST be versioned. The version number of the initial data model SHALL be 1. Each published change to the initial published DOTS data model SHALL increment the data model version by 1.

How the protocol represents data model versions is not defined in this document.

DM-003: Mitigation Status Representation: The data model MUST provide the ability to represent a request for mitigation and the withdrawal of such a request. The data model MUST also support a representation of currently requested mitigation status, including failures and their causes.

DM-004: Mitigation Scope Representation: The data model MUST support representation of a requested mitigation's scope. As mitigation scope may be represented in several different ways, per OP-006 above, the data model MUST be capable of flexible representation of mitigation scope.

DM-005: Mitigation Lifetime Representation: The data model MUST support representation of a mitigation request's lifetime, including mitigations with no specified end time.

DM-006: Mitigation Efficacy Representation: The data model MUST support representation of a DOTS client's understanding of the efficacy of a mitigation enabled through a mitigation request. TBD: how do we represent the efficacy?

DM-007: Relationship to Transport: The DOTS data model MUST NOT depend on the specifics of any transport to represent fields in the model.

3. Congestion Control Considerations

3.1. Signal Channel

As part of a protocol expected to operate over links affected by DDoS attack traffic, the DOTS signal channel MUST NOT contribute significantly to link congestion. To meet the operational requirements above, DOTS signal channel implementations MUST support UDP. However, UDP when deployed naively can be a source of network congestion, as discussed in [RFC5405]. Signal channel implementations using UDP MUST therefore include a congestion control mechanism. The form of that congestion control is implementation-specific.

Signal channel implementations using TCP may rely on built-in TCP congestion control support.

3.2. Data Channel

As specified in DATA-001, the data channel requires reliable, in-order message delivery. Data channel implementations using TCP may rely on the TCP implementation's built-in congestion control mechanisms.

4. Security Considerations

DOTS is at risk from three primary attacks:

- DOTS agent impersonation
- Traffic injection
- Signaling blocking

The DOTS protocol MUST be designed for minimal data transfer to address the blocking risk. Impersonation and traffic injection mitigation can be managed through current secure communications best practices. See Section 2.4 above for a detailed discussion.

5. Contributors

Med Boucadair Flemming Andreasen

6. Acknowledgments

Thanks to Roman Danyliw and Matt Richardson for careful reading and feedback.

7. Change Log

7.1. 02 revision

7.2. 01 revision

2016-03-21

- Reconciled terminology with -00 revision of [I-D.ietf-dots-use-cases].
- Terminology clarification based on working group feedback.
- Moved security-related requirements to separate section.
- Made resilience/robustness primary general requirement to align with charter.
- Clarified support for unidirectional communication within the bidirection signal channel.
- Added proposed operational requirement to support session redirection.

- Added proposed operational requirement to support conflict notification.
- Added proposed operational requirement to support mitigation lifetime in mitigation requests.
- Added proposed operational requirement to support mitigation efficacy reporting from DOTS clients.
- Added proposed operational requirement to cache lookups of all kinds.
- Added proposed operational requirement regarding NAT traversal.
- Removed redundant mutual authentication requirement from data channel requirements.

7.3. 00 revision

2015-10-15

7.4. Initial revision

2015-09-24 Andrew Mortensen

8. References

8.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <http://www.rfc-editor.org/info/rfc768>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, DOI 10.17487/RFC5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

8.2. Informative References

[I-D.ietf-dots-architecture]
Mortensen, A., Andreasen, F., Reddy, T., christopher_gray3@cable.comcast.com, c., Compton, R., and N. Teague, "Distributed-Denial-of-Service Open Threat Signaling (DOTS) Architecture", draft-ietf-dots-architecture-00 (work in progress), July 2016.

[I-D.ietf-dots-use-cases]
Dobbins, R., Fouant, S., Migault, D., Moskowitz, R., Teague, N., and L. Xia, "Use cases for DDoS Open Threat Signaling", draft-ietf-dots-use-cases-01 (work in progress), March 2016.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC1518] Rekhter, Y. and T. Li, "An Architecture for IP Address Allocation with CIDR", RFC 1518, DOI 10.17487/RFC1518, September 1993, <http://www.rfc-editor.org/info/rfc1518>.

[RFC1519] Fuller, V., Li, T., Yu, J., and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, DOI 10.17487/RFC1519, September 1993, <http://www.rfc-editor.org/info/rfc1519>.

[RFC2373] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, DOI 10.17487/RFC2373, July 1998, <http://www.rfc-editor.org/info/rfc2373>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <http://www.rfc-editor.org/info/rfc4271>.

[RFC4732] Handley, M., Ed., Rescorla, E., Ed., and IAB, "Internet Denial-of-Service Considerations", RFC 4732, DOI 10.17487/RFC4732, December 2006, <http://www.rfc-editor.org/info/rfc4732>.

Authors' Addresses

Andrew Mortensen
Arbor Networks, Inc.
2727 S. State St
Ann Arbor, MI 48104
United States

Email: amortensen@arbor.net

Robert Moskowitz
HTT Consulting
Oak Park, MI 42837
United States

Email: rgm@htt-consult.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tireddy@cisco.com

DOTS WG
Internet-Draft
Intended status: Informational
Expires: May 4, 2017

R. Dobbins, Ed.
Arbor Networks
S. Fouant
Corero Network Security
D. Migault
Ericsson
R. Moskowitz
HTT Consulting
N. Teague
Verisign Inc
L. Xia
Huawei

October 31, 2016

Use cases for DDoS Open Threat Signaling
draft-ietf-dots-use-cases-02.txt

Abstract

The DDoS Open Threat Signaling (DOTS) effort is intended to provide a protocol that facilitates interoperability between multivendor solutions/services. This document presents use cases to evaluate the interactions expected between the DOTS components as well as the DOTS exchanges. The purpose of the use cases is to identify the interacting DOTS component, how they collaborate and what are the type of informations to be exchanged.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 4, 2017.

Table of Contents

1. Introduction

Currently, distributed denial-of-service (DDoS) attack mitigation solutions/services are largely based upon siloed, proprietary communications paradigms which result in vendor/service lock-in, and as a side-effect make the configuration, provisioning, operation, and activation of these solutions a highly manual and often time-consuming process. Additionally, coordination of multiple DDoS mitigation solutions/services simultaneously engaged in defending the same organization against DDoS attacks is fraught with both technical and process-related hurdles which greatly increase operational complexity and often result in suboptimal DDoS attack mitigation efficacy.

The DDoS Open Threat Signaling (DOTS) effort is intended to provide a protocol that facilitates interoperability between multivendor solutions/services. As DDoS solutions/services are broadly heterogeneous among different vendor, the primary goal for DOTS is to provide a high level interaction with these DDoS solutions/services such as initiating or terminating the the service/solution. In addition, DOTS is limited to DDoS and may be used by a node under attack. More specifically, DOTS does not intend to become a generic purpose used to orchestrate different DDoS mitigation services/ solutions and the use of DOTS by node under a DDoS attack is expected to impact the design of the DOTS protocol. As a result, although DOTS may be used in the future for further signaling, the current document limits DOTS to a DDoS signaling protocol. It should be noted that DOTS is not in and of itself intended to perform orchestration functions duplicative of the functionality being developed by the [I2NSF] WG; rather, DOTS is intended to allow devices, services, and applications to request mitigation assistance and receive mitigation status updates from systems of this nature.

This document provides use cases where DDoS mitigation is handled using DOTS. The use case presented in the document are intended to clarify what interactions are envisioned with DOTS, as well as the

nodes interacting using DOTS. In both cases, the use cases are expected to provide inputs for the design of DOTS.

2. Terminology and Acronyms

2.1. Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.2. Acronyms

This document makes use of the same terminology and definitions as [I-D.ietf-dots-requirements], except where noted.

3. Use Cases Scenarios

This section provides a high level description of scenarios addressed by DOTS. These scenarios are described in more details in Appendix A. In both sections, the scenarios are provided in order to illustrate the purpose of DOTS. They are not limitative and other use cases are expected to appear during the deployment of DOTS.

All scenarios presents a coordination between the DDoS target, the DDoS attack telemetry and the mitigator. The coordination and communication between these entity depends, for example on the characteristic or functionality of the equipment, the reliability of the information provided by DDoS attack telemetry, the business relationship between the DDoS target domain and the mitigator.

More explicitly, in some cases, the DDoS telemetry attack may simply activate a DDoS mitigation, whereas in some case, it may collaborate by providing some information. In some cases, the DDoS mitigation may be orchestrated, which includes selecting an specific appliance as well as starting/ending a mitigation.

3.1. CPE Intra-domain DDoS Mitigation

The most elementary scenario considers a equipment such as a CPE that when overloaded sends an alert to specific equipment located upstream. In most cases, these very basic equipment are unlikely to diagnose whether an DDoS attack is ongoing or not and detection as well as potential mitigation is left to the upstream equipment.

In most deployment, the upstream equipment belong to the same domain as the CPE. In such case, it is not expected that a specific contract is established between the CPE and the DDoS mitigation

service. The CPE and concerned traffic is likely to be identified by the source of the alert, which also imply the mitigator is aware of the nature of the equipment as well as the architecture of the domain.

The DDoS mitigation service may be for example an equipment that is located on path or a controller that will configure the network to the traffic to be analyzed and mitigated is redirected to a dedicated vendor specific equipment or solution. The DDoS mitigation service may be activated only for the traffic associated to the CPE sending the alert or instead to the traffic associated to all CPE. Such decision are not part of DOTS, but instead depends on the policies of the network administrator.

The DDoS mitigation service is expected to acknowledge the reception of the alert in order to avoid retransmission. This may become an issue for example if an ISP receives alerts from all CPEs multiple time. However, it is unlikely that in such cases the CPE will follow the status of the mitigation. Instead, as the DDoS mitigation service and the CPE belongs to the same administrative domain, it is expected that the decision of mitigating or not, as well as the decision to end an ongoing mitigation will be left to DDoS mitigation service without notice to the CPEs.

3.2. Service/System Intra-domain DDoS Mitigation

This section considers that some more specialized equipment are sending the DDoS alert. As opposed to the CPE, these equipment are likely to provide reliable information about the ongoing attack. Such equipment could typically be a telemetry system, or a specific target service such as a specific instance of web server, or a specific web application detecting application specific attacks.

Such information is likely to be carried in the alert and taken into account by the DDoS mitigation service to proceed to further action. Typically a telemetry system may indicate selectors of the suspicious traffic as well indicators or qualification of the detected attack. As the telemetry system is expected to monitor multiple aspect of the traffic. Similarly when an attack is detected by the target service. The destination of the alert is likely to receive alert from multiple different services (DNS, HTTP, TCP, UDP, application layer specific...). Such information is likely to be trusted and considered by the mitigator to apply the appropriated security appliance.

Note that within a single domain it likely that the service or the telemetry system are most accurate equipment to qualify the attack. As a result, not providing the information is likely to re-do the

analysis phase. Providing the information while sending the alert avoid re-processing the analysis. Instead the mitigator uses directly the information to redirect the traffic to the appropriated specialized appliance.

For the same reasons as the CPE, as mitigation of the DDoS Service is performed in a single administrative domain, the source of the alert may not manage the end of the mitigation service and leave such decision to the administrator of domain or the DDoS mitigation service.

3.3. Orchestrating Intra-domain DDoS Mitigation

This section presents a generalization of the Service/System intra-domain scenario. Orchestration goes one step further and considers that the information carried by the alert could have some management purpose. This includes explicitly starting / ending a mitigation as well as selecting a specific DDoS mitigation service. This differs from the previous case in that the source of the alert does not leave anymore the decision on how to mitigate the attack by the mitigator. Instead the mitigator is orchestrated.

Typical example of orchestrators could be a network administrator that monitors the traffic and initiates manually a DDoS mitigation from its web portal. Orchestration may also applied automatically by an orchestrator.

3.4. Inter-domain DDoS Mitigation

In the case of inter-domain mitigation, it is expected that the DDoS mitigation service has more resource, know-how than the target domain. As a result, there is little benefit of sharing the information collected in the target domain. In addition, the relation between the two domains are also expected to be described into a pre-agreed contract. In that sense, the alert can be restraint to an activation of the DDoS mitigation service.

On the other hand, has there is a contract agreement, it is also expected that target domain is able to stop the DDoS mitigation service itself, and that the end of the mitigation is not unilaterally provided to the DDoS mitigation service.

4. Use Cases Taxonomy

The purpose of DDoS Open Threat Signaling DOTS is to enable the coordination of multiple vendor DDoS mitigation services/systems. DOTS communication is a communication between a DOTS Client and a DOTS Server. A DOTS Client or DOTS Server can be hosted on different

nodes which are associated to different functionalities, and thus leading to different expectations from DOTS. This section provides a classification of the DOTS Client, the DOTS Servers as well as the different type of exchanges.

The high level classification is then illustrated on concrete nodes and examples. Appendix also illustrate the current classification with scenario and complete description of the process.

4.1. DOTS Client Taxonomy

DOTS Client initiates a DOTS communication in order to alert an DDoS attack is ongoing or to coordinate a DDoS mitigation. Coordination of a DDOS Mitigation with DOTS includes initiating/terminations of an DDoS mitigation service/system as well as controlling the status of an ongoing DDoS mitigation.

Note that the section only considers DOTS Client that are actually initiating an exchange with a DOTS Server, and nodes that simply relay DOTS messages are not considered here.

Here are the categories of DOTS Client envisioned in this document:

(a) DOTS Client alerting a DDoS attack is ongoing

  i) hosted on the target attack

  ii) hosted on a monitoring service/system

(b) DOTS Client coordinating an DDoS attack mitigation

  i) hosted on an orchestrator

  ii) hosted on administrative GUI

When the DOTS Client is hosted on the attack target. The DOTS Client mostly raised an alert to the DDoS Mitigation service/system. When a alert is raised by the node under attack, very little information is expected to be provided by DOTS Client to the DDoS mitigation service/system. More particularly telemetric information or characteristics of the attack are likely to be unreliable as the host is already overload. As a result, such DOTS Client may raise an alert without any additional information. Eventually, information such as the asset under attack which can simply be configured. The asset under attack is especially useful for the DDoS mitigation service/system to indicate the origin of the alert. It is not necessary, for example, if the origin of the alert is implicit. The origin of the alert my be implicit, for example when DOTS Clients are

authenticated or when the device is identified by the links (i.e when the host is a CPE). Note also that the asset to protect is only informational and optional. This information may be spoofed, and the DDoS mitigation is likely to be derived from the authentication of the alert. In most cases, the DDoS mitigation has been pre-agreed between the host under attack and the DDoS mitigation service/system.

When the DOTS Client is hosted on a monitoring system, the monitoring system may raise an alert an attack is ongoing. Unlike the host under attack, the monitoring system is expected to have sufficient resource so it is not itself overload and impacted by the ongoing attack. As a result, the DOTS Client is more likely to provide additional information associated to the alert, as this information is expected to be reliable. The type of information associated may be associated to the asset to protect and eventually some information qualifying the attack. On the other hand, the information associated also depends on how the what has been agreed with DDoS mitigation service/system. In most cases, when a DDoS attack is detected all the traffic is redirected to the DDoS mitigation procedure has been agreed between the DDoS mitigation service/system and the entity hosting the monitoring service. In such cases, very few information is needed.

When the DOTS Client is hosted on an orchestrator, the DOTS Client contacts the DDoS mitigation service/system to initiates a DDoS mitigation. The orchestrator is responsible for setting the network to redirect the traffic to the DDoS mitigation service/system. If the DDoS mitigation service/system is not available, the orchestrator is responsible to find an alternative. Again the orchestrator is likely to provide additional information to the DDoS mitigation service/system. For example, typical information may be the asset to protect, as well as the specific mitigation function requested. On the other hand, the service is usually expected to be associated to the mitigation service, and so may not be explicitly specified. In addition, the DOTS Client is also expected to control how the DDoS mitigation is performed. More specifically, it is expected that the DOTS Client can terminate the DDoS mitigation. In addition, the DOTS Client should have sufficient information to decide how to operate next. For example, it should be able to check if the mitigation is ongoing as well as the efficiency of the mitigation.

When the DOTS client is hosted on an administrative system, the DOTS Client may be triggered by the network administrator to initiate a DDoS mitigation. In this case, the DOTS Server is likely to be an orchestrator, and all necessary information may be provided so the DDoS mitigation can be initiated. This includes, the asset to be protected, the action expected to be performed by the orchestrator, the DDoS mitigation service/system to contact...

Note that information associated by the DOTS Client to a request for mitigation is not limited. However, as DDoS mitigation systems are highly heterogeneous, if there is a need to provide interoperability between the vendors and DDoS mitigation services/systems, that actions provided by a DOTS Clients remains small and accepted by all services/systems. As a result here are the envisioned optional information provided by the DOTS Client.

(a) recommended asset to protect (IP, port). This information specifies the expected action from the DDoS mitigation service/ system.

(b) optional DDoS Mitigation Contract ID: which references the contract agreed out-of-band. This information specifies the expected action from the DDoS mitigation service/system.

(c) optional Requested Service: which designates the function or service associated to the DDoS mitigation service/system. This information specifies the expected action from the DDoS mitigation service/system.

(d) optional DDoS attack information (suspected attack, telemetry): This information is expected to help the mitigation service/ system to diagnose the ongoing attack.

In both cases, the DOTS Client sends a request for DDoS mitigation to the DOTS Server, and expects the DDoS mitigation service/system mitigates the DDoS attack. The difference between sending a request for DDoS mitigation as an alert or for coordinating an DDoS mitigation is that an alert is a request to completely outsource the mitigation, whereas the coordination requires additional control over the DDoS mitigation. An alert may be acknowledged by the DOTS Server to acknowledge the reception whereas during the coordination, the the DOTS server may acknowledge the initiation of the DDoS mitigation.

4.2. DOTS Server Taxonomy

DOTS Servers terminate the DOTS communication. The DOTS Server is typically hosted on a DDoS mitigation service/system or an intermediary node such as an orchestrator.

The DOTS Server is expected to be the entry point of a DDoS mitigation service/system. Some DOTS Client do not expect any interaction from the DOTS Server, once a DDoS mitigation has been requested. This is especially true for DOTS Client hosted on attack target. Other DOTS Client hosted on orchestrators or DDoS mitigation service/systems are likely to expect for the DOTS Server a confirmation the system accepts the DDoS mitigation task.

Respectively, these DOTS Client are also likely to expect a confirmation when a DDoS mitigation termination has been requested. In addition, DOTS Server are also expected to provide information related to the mitigation status when requested by the DOTS Client. In addition, it is also expected that the DOTS Server could provide some status report of the DDoS mitigation on a push basis.

4.3. DOTS Message Taxonomy

The core essential messages to coordination an heterogeneous set of DDoS mitigation services/system needs to be small and enable future options. Here are the different exchanges envisioned in this document between a DOTS Client and a DOTS Server.

(a) DOTS MITIGATION CONTROL messages are used by the DOTS Client to initiate or terminate a DDoS mitigation. The initiator the termination can be specified by the action type START or STOP. Such message can carry some additional options that specify additional information such as the asset under attack for example. These DOTS MITIGATION CONTROL messages are expected to be ACKed by the DOTS Server, in order to indicate the DOTS Server will perform the requested action. In any other case an error is expected to be returned. ven in the case of a DOTS Client sends an alert, ACK is recommended so the DOTS Client stop sending the alert.

(b) DOTS MITIGATION INFORMATIONAL message are left for any additional interaction between a DOTS Client and DOTS Server regarding an ongoing request. INFORMATIONAL message can be ignored by the receiver if it does not not understand the the requested information or options. In the current document an informational message can be the status of the ongoing mitigation.

(c) DOTS ERROR contains the errors associated to a request.

DOTS OPTIONS: options can be used to indicate some optional information. The option is expected to specify whether the DOTS Server can ignore it or must return an error if it is not understood. Options are not message, but part of the message.

5. Security Considerations

DOTS is at risk from three primary attacks: DOTS agent impersonation, traffic injection, and signaling blocking. The DOTS protocol MUST be designed for minimal data transfer to address the blocking risk.

Impersonation and traffic injection mitigation can be managed through current secure communications best practices. DOTS is not subject to anything new in this area. One consideration could be to minimize the security technologies in use at any one time. The more needed, the greater the risk of failures coming from assumptions on one technology providing protection that it does not in the presence of another technology.

Additional details of DOTS security requirements may be found in [I-D.ietf-dots-requirements].

6. IANA Considerations

No IANA considerations exist for this document at this time.

7. Acknowledgments

TBD

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[APACHE] "Apache mod_security", <https://www.modsecurity.org>.

[I-D.ietf-dots-requirements] Mortensen, A., Moskowitz, R., and T. Reddy, "Distributed Denial of Service (DDoS) Open Threat Signaling Requirements", draft-ietf-dots-requirements-03 (work in progress), October 2016.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <http://www.rfc-editor.org/info/rfc6335>.

[RRL] "BIND RRL", <https://deepthought.isc.org/article/AA-00994/0/Using-the-Response-Rate-Limiting-Feature-in-BIND-9.10.html>.

Appendix A. Use Cases

This section provides a high-level overview of likely use cases and deployment scenarios for DOTS-enabled DDoS mitigation services. It should be noted that DOTS servers may be standalone entities which, upon receiving a DOTS mitigation service request from a DOTS client, proceed to initiate DDoS mitigation service by communicating directly or indirectly with DDoS mitigators, and likewise terminate the service upon receipt of a DOTS service termination request; conversely, the DDoS mitigators themselves may incorporate DOTS servers and/or DOTS clients. The mechanisms by which DOTS servers initiate and terminate DDoS mitigation service with DDoS mitigators is beyond the scope of this document.

All of the primary use cases described in this section are derived from current, real-world DDoS mitigation functionality, capabilities, and operational models.

The posited ancillary use cases described in this section are reasonable and highly desirable extrapolations of the functionality of baseline DOTS capabilities, and are readily attainable in the near term.

Each of the primary and ancillary use cases described in this section may be read as involving one or more DDoS mitigation service providers; DOTS makes multi-provider coordinated DDoS defenses much more effective and practical due to abstraction of the particulars of a given DDoS mitigation service/solution set.

Both the primary and ancillary use cases may be facilitated by direct DOTS client - DOTS server communications or via DOTS relays deployed in order to aggregate DOTS mitigation service requests/responses, to mediate between stateless and stateful underlying transport protocols, to aggregate multiple DOTS requests and/or responses, to filter DOTS requests and/or responses via configured policy mechanisms, or some combination of these functions.

All DOTS messages exchanged between the DOTS clients and DOTS servers in these use cases may be communicated directly between DOTS clients and servers, or mediated by one or more DOTS relays residing on the network of the originating network, the network where upstream DDoS mitigation service takes place, an intervening network or networks, or some combination of the above.

DOTS is intended to apply to both inter- and intra-domain DDoS attack mitigation scenarios. The technical and operational requirements for inter- and intra-domain DOTS communications are identical. The main difference is administrative in nature; although it should be noted

that provisioning challenges which are typically associated with inter- domain DOTS communications relationships may also apply in intra- domain deployment scenarios, based upon organizational factors. All of the same complexities surrounding authentication and authorization can apply in both contexts, including considerations such as network access policies to allow DOTS communications, DOTS transport selection (including considerations of the implications of link congestion if a stateful DOTS transport option is selected), etc. Registration of well-known ports for DOTS transports per [RFC6335] should be considered in light of these challenges.

It should also be noted that DOTS does not directly ameliorate the various administrative challenges required for successful DDoS attack mitigation. Letters of authorization, RADB updates, DNS zone delegations, alteration of network access policies, technical configurations required to facilitate network traffic diversion and re-injection, etc., are all outside the scope of DOTS. DOTS may, however, prove useful in automating the registration of DOTS clients with DOTS servers, as well as in the automatic provisioning of situationally- appropriate DDoS defenses and countermeasures. This ancillary DOTS functionality is described in Appendix A.2.

Many of the 'external' administrative challenges associated with establishing workable DDoS attack mitigation service may be addressed by work currently in progress in the I2RS and I2NSF WGs. Interested parties may wish to consider tracking those efforts, and coordination with both I2RS and I2NSF is highly desirable.

Note that all the use-cases in this document are universal in nature. They apply equally to endpoint networks, transit backbone providers, cloud providers, broadband access providers, ASPs, CDNs, etc. They are not specific to particular business models, topological models, or application types, and are deliberately generalizable. Both networks targeted for attack as well as any adjacent or topologically distant networks involved in a given scenario may be either single- or multi-homed. In the accompanying vector illustrations incorporated into draft-ietf-dots-use-cases-01.pdf, specific business and topological models are described in order to provide context.

Likewise, both DOTS itself and the use cases described in this document are completely independent of technologies utilized for the detection, classification, traceback, and mitigation of DDoS attacks. Flow telemetry such as NetFlow and IPFIX, direct full-packet analysis, log-file analysis, indirection manual observation, etc. can and will be enablers for detection, classification and traceback. Intelligent DDoS mitigation systems (IDMSes), flowspec, S/RTBH, ACLs, and other network traffic manipulation tools and techniques may be used for DDoS attack mitigation. BGP, flowspec, DNS, inline

deployment, and various 'NFV' technologies may be used for network traffic diversion into mitigation centers or devices in applicable scenarios; GRE, MPLS, 'NFV', inline deployment and other techniques may be utilized for 'cleaned' traffic re-injection to its intended destination.

The scope, format, and content of all DOTS message types cited in this document must be codified by the DOTS WG.

The following use cases are intended to inform the DOTS requirements described in [I-D.ietf-dots-requirements].

A.1. Primary Use Cases

A.1.1. Automatic or Operator-Assisted CPE or PE Mitigators Request Upstream DDoS Mitigation Services

One or more CPE or PE mitigators with DOTS client capabilities may be configured to signal to one or more DOTS servers in order to request upstream DDoS mitigation service initiation during an attack when DDoS attack volumes and/or attack characteristics exceed the capabilities of such CPE mitigators. DDoS mitigation service may be terminated either automatically or manually via a DOTS mitigation service termination request initiated by the mitigator when it has been determined that the DDoS attack has ended.

(a) A DDoS attack is initiated against online properties of an organization which has deployed DOTS-client-capable DDoS mitigators.

(b) CPE or PE DDoS mitigators detect, classify, and begin mitigating the DDoS attack.

(c) CPE or PE DDoS mitigators determine that their capacity and/or capability to mitigate the DDoS attack is insufficient, and utilize their DOTS client functionality to send a DOTS mitigation service initiation request to one or more DOTS servers residing on one or more upstream transit networks, peer networks, or overlay MSSP networks. This DOTS mitigation service initiation request may be automatically initiated by the CPE or PE DDoS mitigators, or may be manually triggered by personnel of the requesting organization in response to an alert from the mitigators (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been configured to honor requests from the requesting CPE or PE mitigators, and

initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(e) The DOTS servers transmit a DOTS service status message to the requesting CPE or PE mitigators indicating that upstream DDoS mitigation service has been initiated.

(f) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the requesting CPE or PE mitigators.

(g) While DDoS mitigation services are active, the CPE or PE mitigators may optionally regularly transmit DOTS mitigation efficacy updates to the relevant DOTS servers.

(h) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(i) The DOTS servers transmit a DOTS mitigation status update to the CPE or PE mitigators indicating that the DDoS attack has ceased.

(j) The CPE or PE DDoS mitigators transmit a DOTS mitigation service termination request to the DOTS servers. This DOTS mitigation service termination request may be automatically initiated by the CPE or PE DDoS mitigators, or may be manually triggered by personnel of the requesting organization in response to an alert from the mitigators or a management system which monitors them (the mechanism by which this process takes place is beyond the scope of this document).

(k) The DOTS servers terminate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(l) The DOTS servers transmit a DOTS mitigation status update to the CPE or PE mitigators indicating that DDoS mitigation services have been terminated.

(m) The CPE or PE DDoS mitigators transmit a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.2. Automatic or Operator-Assisted CPE or PE Network Infrastructure Element Request to Upstream Mitigator

CPE or PE network infrastructure elements such as routers, switches, load-balancers, firewalls, 'IPSes', etc. which have the capability to detect and classify DDoS attacks and which have DOTS client capabilities may be configured to signal to one or more DOTS servers in order to request upstream DDoS mitigation service initiation during an attack. DDoS mitigation service may be terminated either automatically or manually via a DOTS mitigation service termination request initiated by the network element when it has been determined that the DDoS attack has ended.

In this use-case, the network elements involved are not engaged in mitigating DDoS attack traffic. They are signaling for upstream attack mitigation assistance. This can be an inter- or intra- domain use-case.

(a) A DDoS attack is initiated against online properties of an organization with DOTS-client-capable network infrastructure elements deployed.

(b) The network infrastructure elements utilize their DOTS client functionality to send a DOTS mitigation service initiation request to one or more DOTS servers residing on one or more upstream transit networks, peer networks, or overlay MSSP networks, either directly or via intermediate DOTS relays residing upon the requesting organization's network, the upstream mitigation provider's network, or both. The scope, format, and content of these messages must be codified by the DOTS WG. This DOTS mitigation service initiation request may be automatically initiated by the network infrastructure elements, or may be manually triggered by personnel of the requesting organization in response to an alert from the network elements or a management system which monitors them (the mechanism by which this process takes place is beyond the scope of this document).

(c) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been configured to honor requests from the requesting network infrastructure elements, and initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers transmit a DOTS service status message to the requesting network infrastructure elements indicating that upstream DDoS mitigation service has been initiated.

(e) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the requesting requesting network infrastructure elements.

(f) While DDoS mitigation services are active, the network infrastructure elements may optionally regularly transmit DOTS mitigation efficacy updates to the relevant DOTS servers.

(g) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(h) The DOTS servers transmit a DOTS mitigation status update to the network infrastructure elements indicating that the DDoS attack has ceased.

(i) The network infrastructure elements transmit a DOTS mitigation service termination request to the DOTS servers. This DOTS mitigation service termination request may be automatically initiated by the network infrastructure elements, or may be manually triggered by personnel of the requesting organization in response to an alert from the mitigators (the mechanism by which this process takes place is beyond the scope of this document).

(j) The DOTS servers terminate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(k) The DOTS servers transmit a DOTS mitigation status update to the network infrastructure elements indicating that DDoS mitigation services have been terminated.

(l) The network infrastructure elements transmit a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.3. Automatic or Operator-Assisted CPE or PE Attack Telemetry Detection/Classification System Request to Upstream Mitigator

CPE or PE Attack Telemetry Detection/Classification Systems which have DOTS client capabilities may be configured so that upon detecting and classifying a DDoS attack, they signal one or more DOTS servers in order to request upstream DDoS mitigation service initiation. DDoS mitigation service may be terminated either automatically or manually via a DOTS mitigation service termination request initiated by the Attack Telemetry Detection/Classification System when it has been determined that the DDoS attack has ended.

In this use-case, the Attack Telemetry Detection/Classification does not possess any inherent capability to mitigate DDoS attack traffic, and is signaling for upstream mitigation assistance. This can be an inter- or intra-domain use-case.

(a) A DDoS attack is initiated against online properties of an organization with DOTS-client-capable CPE or PE Attack Telemetry Detection/Classification Systems deployed.

(b) The CPE or PE Attack Telemetry Detection/Classification Systems utilize their DOTS client functionality to send a DOTS mitigation service initiation request to one or more DOTS servers residing on one or more upstream transit networks, peer networks, or overlay MSSP networks, either directly or via intermediate DOTS relays residing upon the requesting organization’s network, the upstream mitigation provider’s network, or both. This DOTS mitigation service initiation request may be automatically initiated by the CPE or PE Attack Telemetry Detection/Classification Systems, or may be manually triggered by personnel of the requesting organization in response to an alert from the CPE or PE Attack Telemetry Detection/Classification Systems (the mechanism by which this process takes place is beyond the scope of this document).

(c) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been configured to honor requests from the requesting CPE or PE Attack Telemetry Detection/Classification Systems, and initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers transmit a DOTS service status message to the requesting CPE or PE Attack Telemetry Detection/Classification Systems indicating that upstream DDoS mitigation service has been initiated.

(e) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the requesting CPE or PE Attack Telemetry Detection/Classification Systems.

(f) While DDoS mitigation services are active, the CPE or PE Attack Telemetry Detection/Classification Systems may optionally regularly transmit DOTS mitigation efficacy updates to the relevant DOTS servers.

(g) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(h) The DOTS servers transmit a DOTS mitigation status update to the CPE or PE Attack Telemetry Detection/Classification Systems indicating that the DDoS attack has ceased.

(i) The CPE or PE Attack Telemetry Detection/Classification Systems transmit a DOTS mitigation service termination request to the DOTS servers. This DOTS mitigation service termination request may be automatically initiated by the CPE or PE Attack Telemetry Detection/Classification Systems, or may be manually triggered by personnel of the requesting organization in response to an alert from the CPE or PE Attack Telemetry Detection/ Classification Systems (the mechanism by which this process takes place is beyond the scope of this document).

(j) The DOTS servers terminate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(k) The DOTS servers transmit a DOTS mitigation status update to the CPE or PE Attack Telemetry Detection/Classification Systems indicating that DDoS mitigation services have been terminated.

(l) The CPE or PE Attack Telemetry Detection/Classification Systems transmit a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.4. Automatic or Operator-Assisted Targeted Service/ Application Request to Upstream Mitigator

A service or application which is the target of a DDoS attack and which has the capability to detect and classify DDoS attacks (i.e, Apache mod_security [APACHE], BIND RRL [RRL], etc.) as well as DOTS client functionality may be configured so that upon detecting and classifying a DDoS attack, it signals one or more DOTS servers in order to request upstream DDoS mitigation service initiation. DDoS mitigation service may be terminated either automatically or manually via a DOTS mitigation service termination request initiated by the service/application when it has been determined that the DDoS attack has ended.

In this use-case, the service/application does not possess inherent DDoS attack mitigation capabilities, and is signaling for upstream

mitigation assistance. This can be an inter- or intra-domain use-case.

(a) A DDoS attack is initiated against online properties of an organization which include DOTS-client-capable services or applications that are the specific target(s) of the attack.

(b) The targeted services or applications utilize their DOTS client functionality to send a DOTS mitigation service initiation request to one or more DOTS servers residing on the same network as the services or applications, one or more upstream transit networks, peer networks, or overlay MSSP networks, either directly or via intermediate DOTS relays residing upon the requesting organization's network, the upstream mitigation provider's network, or both. This DOTS mitigation service initiation request may be automatically initiated by the targeted services or applications, or may be manually triggered by personnel of the requesting organization in response to an alert from the targeted services or applications or a system which monitors them (the mechanism by which this process takes place is beyond the scope of this document).

(c) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been provisioned to honor requests from the requesting services or applications, and initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers transmit a DOTS service status message to the services or applications indicating that upstream DDoS mitigation service has been initiated

(e) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the requesting services or applications.

(f) While DDoS mitigation services are active, the requesting services or applications may optionally regularly transmit DOTS mitigation efficacy updates to the relevant DOTS servers.

(g) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(h) The DOTS servers transmit a DOTS mitigation status update to the requesting services or applications indicating that the DDoS attack has ceased.

(i) The targeted services or applications transmit a DOTS mitigation service termination request to the DOTS servers. This DOTS mitigation service termination request may be automatically initiated by the targeted services or applications, or may be manually triggered by personnel of the requesting organization in response to an alert from a system which monitors them (the mechanism by which this process takes place is beyond the scope of this document).

(j) The DOTS servers terminate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(k) The DOTS servers transmit a DOTS mitigation status update to the targeted services or applications indicating that DDoS mitigation services have been terminated.

(l) The targeted services or applications transmit a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.5. Manual Web Portal Request to Upstream Mitigator

A Web portal which has DOTS client capabilities has been configured in order to allow authorized personnel of organizations which are targeted by DDoS attacks to manually request upstream DDoS mitigation service initiation from a DOTS server. When an organization has reason to believe that it is under active attack, authorized personnel may utilize the Web portal to manually initiate a DOTS client mitigation request to one or more DOTS servers. DDoS mitigation service may be terminated manually via a DOTS mitigation service termination request through the Web portal when it has been determined that the DDoS attack has ended.

In this use-case, the organization targeted for attack does not possess any automated or operator-assisted mechanisms for DDoS attack detection, classification, traceback, or mitigation; the existence of an attack has been inferred manually, and the organization is requesting upstream mitigation assistance. This can theoretically be an inter- or intra-domain use-case, but is more typically an inter-domain scenario.

(a) A DDoS attack is initiated against online properties of an organization have access to a Web portal which incorporates DOTS

client functionality and can generate DOTS mitigation service requests upon demand.

(b) Authorized personnel utilize the Web portal to send a DOTS mitigation service initiation request to one or more upstream transit networks, peer networks, or overlay MSSP networks, either directly or via intermediate DOTS relays residing upon the requesting organization's network, the upstream mitigation provider's network, or both. This DOTS mitigation service initiation request is manually triggered by personnel of the requesting organization when it is judged that the organization is under DDoS attack (the mechanism by which this process takes place is beyond the scope of this document).

(c) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been provisioned to honor requests from the Web portal, and initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers transmit a DOTS service status message to the Web portal indicating that upstream DDoS mitigation service has been initiated.

(e) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the Web portal.

(f) While DDoS mitigation services are active, the Web portal may optionally regularly transmit manually-triggered DOTS mitigation efficacy updates to the relevant DOTS servers.

(g) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(h) The DOTS servers transmit a DOTS mitigation status update to the Web portal indicating that the DDoS attack has ceased.

(i) The Web portal transmits a manually-triggered DOTS mitigation service termination request to the DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(j) The Web portal transmits a manually-triggered DOTS mitigation service termination request to the DOTS servers (the mechanism

by which this process takes place is beyond the scope of this document).

(k) The DOTS servers transmit a DOTS mitigation status update to the Web portal indicating that DDoS mitigation services have been terminated.

(l) The Web portal transmits a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.6. Manual Mobile Device Application Request to Upstream Mitigator

An application for mobile devices such as smartphones and tablets which incorporates DOTS client capabilities has been made available to authorized personnel of an organization. When the organization has reason to believe that it is under active DDoS attack, authorized personnel may utilize the mobile device application to manually initiate a DOTS client mitigation request to one or more DOTS servers in order to initiate upstream DDoS mitigation services. DDoS mitigation service may be terminated manually via a DOTS mitigation service termination request initiated through the mobile device application when it has been determined that the DDoS attack has ended.

This use-case is similar to the one described in Appendix A.1.5; the difference is that a mobile application provided by the DDoS mitigation service provider is used to request upstream attack mitigation assistance. This can theoretically be an inter- or intra-domain use-case, but is more typically an inter-domain scenario.

(a) A DDoS attack is initiated against online properties of an organization have access to a Web portal which incorporates DOTS client functionality and can generate DOTS mitigation service requests upon demand.

(b) Authorized personnel utilize the mobile application to send a DOTS mitigation service initiation request to one or more DOTS servers residing on the same network as the targeted Internet properties, one or more upstream transit networks, peer networks, or overlay MSSP networks, either directly or via intermediate DOTS relays residing upon the requesting organization’s network, the upstream mitigation provider’s network, or both. This DOTS mitigation service initiation request is manually triggered by personnel of the requesting organization when it is judged that the organization is under DDoS attack (the mechanism by which this process takes place is beyond the scope of this document).

(c) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been provisioned to honor requests from the mobile application, and initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers transmit a DOTS service status message to the mobile application indicating that upstream DDoS mitigation service has been initiated.

(e) While DDoS mitigation services are active, the DOTS servers regularly transmit DOTS mitigation status updates to the mobile application.

(f) While DDoS mitigation services are active, the mobile application may optionally regularly transmit manually-triggered DOTS mitigation efficacy updates to the relevant DOTS servers.

(g) When the upstream DDoS mitigators determine that the DDoS attack has ceased, they indicate this change in status to their respective DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(h) The DOTS servers transmit a DOTS mitigation status update to the mobile application indicating that the DDoS attack has ceased.

(i) The mobile application transmits a manually-triggered DOTS mitigation service termination request to the DOTS servers (the mechanism by which this process takes place is beyond the scope of this document).

(j) The DOTS servers terminate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(k) The DOTS servers transmit a DOTS mitigation status update to the mobile application indicating that DDoS mitigation services have been terminated.

(l) The mobile application transmits a DOTS mitigation termination status acknowledgement to the DOTS servers.

A.1.7. Unsuccessful Automatic or Operator-Assisted CPE or PE Mitigators Request Upstream DDoS Mitigation Services

One or more CPE or PE mitigators with DOTS client capabilities may be configured to signal to one or more DOTS servers in order to request upstream DDoS mitigation service initiation during an attack when DDoS attack volumes and/or attack characteristics exceed the capabilities of such CPE mitigators. DDoS mitigation service may be terminated either automatically or manually via a DOTS mitigation service termination request initiated by the mitigator when it has been determined that the DDoS attack has ended.

This can theoretically be an inter- or intra-domain use-case, but is more typically an inter-domain scenario.

(a) A DDoS attack is initiated against online properties of an organization which has deployed DOTS-client-capable DDoS mitigators.

(b) CPE or PE DDoS mitigators detect, classify, and begin mitigating the DDoS attack.

(c) CPE or PE DDoS mitigators determine that their capacity and/or capability to mitigate the DDoS attack is insufficient, and utilize their DOTS client functionality to send a DOTS mitigation service initiation request to one or more DOTS servers residing on one or more upstream transit networks, peer networks, or overlay MSSP networks. This DOTS mitigation service initiation request may be automatically initiated by the CPE or PE DDoS mitigators, or may be manually triggered by personnel of the requesting organization in response to an alert from the mitigators (the mechanism by which this process takes place is beyond the scope of this document).

(d) The DOTS servers which receive the DOTS mitigation service initiation requests determine that they have been configured to honor requests from the requesting CPE or PE mitigators, and attempt to initiate situationally-appropriate DDoS mitigation service on their respective networks (the mechanism by which this process takes place is beyond the scope of this document).

(e) The DDoS mitigators on the upstream network report back to the DOTS servers that they are unable to initiate DDoS mitigation service for the requesting organization due to mitigation capacity constraints, bandwidth constraints, functionality constraints, hardware casualties, or other impediments (the mechanism by which this process takes place is beyond the scope of this document).

(f) The DOTS servers transmit a DOTS service status message to the requesting CPE or PE mitigators indicating that upstream DDoS mitigation service cannot be initiated as requested.

(g) The CPE or PE mitigators may optionally regularly re-transmit DOTS mitigation status request messages to the relevant DOTS servers until acknowledgement that mitigation services have been initiated.

(h) The CPE or PE mitigators may optionally transmit a DOTS mitigation service initiation request to DOTS servers associated with a configured fallback upstream DDoS mitigation service. Multiple fallback DDoS mitigation services may optionally be configured.

(i) The process describe above cyclically continues until the DDoS mitigation service request is fulfilled; the CPE or PE mitigators determine that the DDoS attack volume has decreased to a level and/or complexity which they themselves can successfully mitigate; the DDoS attack has ceased; or manual intervention by personnel of the requesting organization has taken place.

A.2. Ancillary Use Cases

A.2.1. Auto-registration of DOTS clients with DOTS servers

An additional benefit of DOTS is that by utilizing agreed-upon authentication mechanisms, DOTS clients can automatically register for DDoS mitigation service with one or more upstream DOTS servers. The details of such registration are beyond the scope of this document.

A.2.2. Auto-provisioning of DDoS countermeasures

The largely manual tasks associated with provisioning effective, situationally-appropriate DDoS countermeasures is a significant barrier to providing/obtaining DDoS mitigation services for both mitigation providers and mitigation recipients. Due to the 'self-descriptive' nature of DOTS registration messages and mitigation requests, the implementation and deployment of DOTS has the potential to automate countermeasure selection and configuration for DDoS mitigators. The details of such provisioning are beyond the scope of this document.

This can theoretically be an inter- or intra-domain use-case, but is more typically an inter-domain scenario.

A.2.3. Informational DDoS attack notification to interested and authorized third parties

In addition to its primary role of providing a standardized, programmatic approach to the automated and/or operator-assisted request of DDoS mitigation services and providing status updates of those mitigations to requesters, DOTS may be utilized to notify security researchers, law enforcement agencies, regulatory bodies, etc. of DDoS attacks against attack targets, assuming that organizations making use of DOTS choose to share such third-party notifications, in keeping with all applicable laws, regulations, privacy and confidentiality considerations, and contractual agreements between DOTS users and said third parties.

This is an inter-domain scenario.

Authors' Addresses

Roland Dobbins (editor)
Arbor Networks
30 Raffles Place
Level 17 Chevron House
Singapore 048622
Singapore

Email: rdobbins@arbor.net

Stefan Fouant
Corero Network Security

Email: Stefan.Fouant@corero.com

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Phone: +1 514-452-2160
Email: daniel.migault@ericsson.com

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
USA

Email: rgm@labs.htt-consult.com

Nik Teague
Verisign Inc
12061 Bluemont Way
Reston, VA 20190
USA

Phone: +44 791 763 5384
Email: nteague@verisign.com

Liang Xia
Huawei
No. 101, Software Avenue, Yuhuatai District
Nanjing
China

Email: Frank.xialiang@huawei.com

RFC Beautification Working Group
Internet-Draft
Intended status: Informational
Expires: December 30, 2016

D. Migault
A. Ranjbar
Ericsson

June 28, 2016

Collaboration Agreement for Security Service Function draft-mglt-i2nsf-ssf-collaboration-00.txt

Abstract

This document specifies a collaboration agreement protocol. The collaboration agreement makes possible individual security services functions (SSF) to collaborate with each other. The collaboration is mostly intended for SSF located in different administrative domains, in which case the collaboration cannot be performed by a shared orchestrator.

The collaboration between SSF in different domains assumes the traffic is steered through the two domains.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 30, 2016.

Copyright Notice

Table of Contents

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

Security Service Function (SSF) has been deployed to mitigate and detect malicious traffic and security threats in networks.

A typical use case would consider today's cloud-based services were a data flows is forwarded from the Internet Service Provider to the cloud which hosts the destination service or any on-path services. The services deployed in the cloud are at least partly implemented using a combination of one or more SSF. Similarly, the ISP may also implement a set of on path SSF. The purpose of the collaboration is to enable a SSF running in the cloud administrative domain to take advantage of specific SSF running in the ISP administrative domain. The SSF may be of same type or of different type.

As the SSFs belong to different administrative domains, collaboration between these two SSFs is unlikely to happen through a common shared

orchestrator. To enable the collaboration between individual SSFs, a collaboration agreement protocol is proposed in this document. This protocol is expected to provide: better detection by exchanging real-time information about the detected attacks between SSFs, better mitigation by enforcing mitigation strategies on more effective network segments (e.g. cloud vs ISP), and better resource usage by eliminating the need for frequent deployment of similar service functions and by spreading the tasks among different SSFs.

3. Collaboration Agreement

The SSFs initiating and accepting the collaboration are called, respectively, 'initiator' and 'provider'. The initiator sends to potential providers a Collaboration Agreement (CA), which defines the necessary attributes involved in the collaboration. Such attributes are expected not to be SSF specific. However attributes that characterize the SSF, such as the SFF type, input and output flows, may be part of the CA simply to allow collaborators to define whether or not they are eligible to provide the corresponding services.

For management purposes, the collaboration agreement should also include an 'agreement ID' and a 'duration' indicating its lifespan. It is the responsibility of the 'initiator' to renew the agreement before it expires, although the 'provider' should also be able to notify the former that the agreement needs to be revised or interrupted earlier due to some unexpected event.

Two collaboration modes are envisioned:

1) 'Resilient', in which the provider is expected to handle the whole load of that traffic; and

2) 'Best Effort', which indicates that the provider supplies the service for a fraction of the load. When the Best Effort mode is chosen, an 'alternate path' indicates where non-treated traffic is forwarded, and 'resource' indicates the resources allocated for the service. The use of 'alternate path' enhances the collaboration between SSFs by allowing the provider to temporarily assign specific amount of resources for handling the packets and send the non-treated traffic through the alternate path to be processed by the initiator. The resources assigned in the Best Effort mode can be expressed in specific ways, such as a combination of various computational resources e.g., CPU, I/O, bandwidth, packet rate, or maximum latency. The manner by which such resources are controlled is left for the provider's implementation (e.g., by leveraging containers and micro services technologies).

Here are the parameters associated to the collaboration agreement:

- ca_id: identifies the collaboration agreement and it can be used later to refer to a specific collaboration agreement.
- initiator: designates the locators (e.g. IP address or FQDN) as well as the authentication credentials associated to the initiator.
- provider: see initiator
- collaboration type: indicates the type of collaboration (Best Effort and Resilient).
- resource: designates the resources agreed on between the initiator and the provider. Note that this parameter is optional as resources are only negotiated when collaboration is in a best effort mode.
- SSF type: designates the type of the security instances running.
- expiration time: designates the expiration of the collaboration.
- interconnections: defines how the interconnection between the initiator and the provider is performed. This includes the definition of the alternate path.
- direction: defines if the provider is expected to be in front of the initiator or behind it.

4. Collaboration Agreement Protocol

The purpose of the collaboration agreement protocol is to negotiate between the initiator and provider and make an agreement for cooperation between SSFs placed in a single or multiple domains. Currently, the collaboration agreement protocol is always originated from the initiator. In other words, the provider is not initiating the exchanges as to announce what it can provide.

The collaboration agreement protocol should include the following attributes:

- ca_id: the is the collaboration agreement identifier. In a case the value is not acceptable, an ERROR_UNACCEPTABLE_CA_ID MUST be returned. There are, however, little reasons such a collision occurs. If such a collision occurs, the negotiation is aborted and must be restarted with a new ca_id.

- initiator: it includes information that the initiator offers to the provider. Upon receiving the request for collaboration, the provider may reject the collaboration agreement by sending a ERROR_UNACCEPTABLE_INITIATOR.

- provider: it consists of information about the provider to be verified by the initiator. Upon receiving this information, the initiator my abort the negotiation with ERROR_UNACCEPTABLE_PROVIDER. The reason for refusing the provider, may be that the provider is not in a white list or that the provider has been explicitly banned by the initiator.

- resource: it represents allocated resources by the provider for collaboration with the initiator. When the collaboration mode is set to Resilient, the resource is not expected to be provided by the provider. For the best effort mode, the resource provided by the provider may consider the indication provided by the initiator or not. Given the resource provided by the provider, the initiator is likely to close the collaboration or to accept it.

In order to define a flexible framework, the negotiation steps between the initiator and provider is designed as mentioned below:

1. The initiator provides a list of proposals to the provider

2. A proposal may contain multiple proposition for a given attribute. For example, let P1 be a proposal offered by the initiator. In this case, the initiator may be willing to make an agreement with the providers either in Best Effort or Resilient modes. In this case, the initiator will set P1 with an object of collaboration type set to Best Effort AND an object of collaboration type set to Resilient.

3. When multiple proposals are received by the provider, the provider is expected to choose a single proposal. The chosen proposal is the one that contains the attributes that fits the provider.

4. When a proposal is chosen, the provider must select for each attribute the preferred value. More especially, when multiple values for a same attribute type are available, the provider selects the preferred value for that attribute. Also, the chosen proposal must have the same amount of attribute types which means the provider is not allowed to remove some attributes or selectively reject attributes.

5. The provider may send an acceptable proposal to the initiator. If none of the proposals are acceptable by the provider, the provider returns a ERROR_UNACCEPTABLE_PROPOSALS.

5. Collaboration Agreement Management operations

Once the collaboration agreement has been agreed between the initiator and provider, the following actions need to be considered during its life cycle.

- END_AGREEMENT: This action can be performed by either peers, that is to say the initiator or the provider. This action requires the ca_id and credentials to identify peers in the agreement.

- UPDATE_EXPIRATION_DATE: This action is initiated by either peers. It intends to update the expiration date. The expiration date can be extended or advanced. The input parameters are the ca_id and the new expiration date. The possible responses are to accept or reject this request. In case of rejection, ERROR_UNACCEPTABLE_NEW_EXPIRATION_DATE is sent to the requested peer.

- UPDATE_RESOURCE: This action is expected to be triggered by the provider. It indicates the amount of resources the provider offers for collaboration. This is an informative message. It may be useful for the initiator to know how much resource will be dedicated to the collaboration by the provider so it can adjust its strategy.

- REDIRECT_SSF: This action is triggered when peers change their location. This actions is initiated by either peers. It may result in changes in Alternate Path in case of Best Effort mode.

6. Error Message handling

The following Error message have been considered so far:

```
ERROR_UNACCEPTABLE_CA_ID
ERROR_UNACCEPTABLE_PROVIDER
ERROR_UNACCEPTABLE_INITIATOR
ERROR_UNACCEPTABLE_PROPOSALS
ERROR_UNACCEPTABLE_NEW_EXPIRATION_DATE
```

7. Payload Format

7.1. Collaboration Agreement Objects

This section represents the Collaboration Agreement object. The collaboration agreement is an object with properties. Some of these properties are object themselves. In order to enrich the object definition, the Collaboration is defined on different objects including 'peer' and 'resource' objects.

'Peer' object represents the necessary information associated to a peer. A peer can be either the initiator or provider. The description of a peer object is as follows:

```
{
  "peer":{
    "type": "object",
    "description": "provides different elements associated to the
                    initiator or the provider. This includes
                    location as well as authentication credentials",
    "properties": {
      "rsakey": {
        "type": "string",
        "description": "RSA public key used to identify the
                        initiator"
      },
      "cert": {
        "type": "array",
        "description": "list of certificates to authenticate the
                        initiator"
      },
      "fqdn": {
        "type": "string",
        "description": " FQDN associated to the initiator"
      },
      "ipv4": {
        "type": "string",
        "description": "IPv4 address used to reach the initiator"
      },
      "ipv6": {
        "type": "string",
        "description": "IPv6 address used to reach the initiator"
      }
    }
  }
}
```

The following object designates the resources agreed between the initiator and the provider.

```
{
  "resource": {
    "type":object,
    "description": "resource engaged into the collaboration",
    "properties": {
      "cpu": {
        "type": "number",
        "description": "cpu limit"
      },
      "memory": {
        "type": "number",
        "description": "memory limit"
      },
      "net": {
        "type": "number",
        "description": "net limit"
      },
      "blkio": {
        "type": "number",
        "description": "block limit"
      }
    }
  }
}
```

The collaboration type is defined as follow:

TYPE	CODE
Resilient	0
Best Effort	1

SSF instance types can be extended to any number of available services. We do not limit SSF types and we expect to extend this number in future. Some example SSFs can be defined as follows:

TYPE	CODE
Rate limiting	0
DNSoverTCP	1
PacketDropper	2

The following object is a Collaboration Agreement object which includes several properties to define an agreement between the provider and initiator.

```
{
  "type": "Collaboration Agreement",
  "description": "This object designates the Collaboration
                  Agreement properties",
  "properties": {
    "ca_id" : {
      "type": "number",
      "description" : "unique identifier of the Collaboration
                       agreement"
    },
    "initiator": {
      "type": "peer",
      "description": "provides the different elements associated
                      to the initiator. This includes location
                      as well as authentication credentials"
       },
    "provider": {
      "type": "peer",
      "description": "provides the different elements associated
                      to the provider. This includes location as
                      well as authentication credentials"
    },
    "collaboration_type": {
      "type": "number",
      "description": "defines whether the type of the
                      collaboration"
    },
    "security_service_instance_type": {
      "type": "number",
      "description": "the type of security service instance"
    },
    "interconnections":{
      "type": "interconnections",
      "description": "the type of security service instance"
    },
    "resource":{
      "type": "resource",
      "description": "the type of security service instance"
    },
    "direction":{
      "type": "direction",
      "description": "indicates whether the provider MUST
                      be placed downstream or upstream"
    }
  }
}
```

7.2. Collaboration Agreement Protocol

The collaboration agreement protocol can be defined as request or response objects.

7.2.1. Collaboration Agreement Protocol Request

The following object defines the request object which includes information about initiator, resources and a set of proposal objects.

```
{
  "type": "collaboration protocol agreement request",
  "description": "object",
  "properties": {
    "ca_id" : {
      "type": "number",
      "description" : "unique identifier for collaboration
                       agreement"
    },
    "initiator":{
      "type": "peer",
      "description": "provides different elements associated
                      to the initiator. This includes location
                      as well as authentication credentials"
    },
    "informative resource requested":{
      "type": "resource",
     "description": "the type of security service instance"
    },
    "proposals":{
      "type": "Array",
      "description": "Array of proposals offered by the initiator"
    }
  }
}
```

A proposal object can also be defined as follows:

```
{
  "type": "object",
  "description": "A single proposal with a set of attributes.
                  The expected attribute types are collaboration
                  type, security service instance type and
                  interconnections",
  "properties":{
    "proposal_id":{
      "type": "number"
    }
    "proposed-attribute":{
      "type": "object"
      "properties":{
        "attribute-type"{
          type: string
        }
        "attribute-values"{
          "type": "array"
          "items": {
            attribute-value
          }
        }
      }
    }
  }
}
```

7.2.2. Collaboration Agreement Protocol Response

The response object is similar to the request object except that:

- The response must include a provider object.
- The proposed list of attribute values must be of size one with the chosen value.

7.3. Collaboration Agreement Protocol Additional Operations

When the initiator and provider are placed in different domains, additional orchestration operations might be needed between domains to make an agreement. Moreover, in case of Best Effort mode, additional operations is needed to establish an alternate path and separate the treated traffic from non-treated traffic e.g. by deploying classifiers on the path.

8. Security Considerations

9. IANA Considerations

10. Acknowledgements

11. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

Authors' Addresses

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Phone: +1 514-452-2160
Email: daniel.migault@ericsson.com

Alireza Ranjbar
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Phone: +358-442992904
Email: alireza.ranjbar@ericsson.com

DOTS
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2017

K. Nishizuka
NTT Communications
L. Xia
J. Xia
Huawei Technologies Co., Ltd.
D. Zhang

L. Fang
Microsoft
C. Gray
Comcast, Inc.
R. Compton
Charter Communications, Inc.
July 1, 2016

Inter-domain cooperative DDoS protection mechanism
draft-nishizuka-dots-inter-domain-mechanism-01

Abstract

As DDoS attacks evolve rapidly in the aspect of volume and sophistication, cooperation among operators becomes very necessary because it will give us quicker and more sophisticated protection to cope with them. This document describes possible mechanisms which implement the cooperative inter-domain DDoS protection by DOTS protocol. The described data models are intended to cover intra-domain and inter-domain solutions.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 2, 2017.

Table of Contents

1. Introduction

These days, DDoS attacks are getting bigger and more sophisticated. Preliminary measures for minimizing damages caused by such attacks are indispensable to all organizations facing to the Internet. Due to the various types of UDP reflection attacks that can be executed, there are still large DDoS attacks being generated which consist of vulnerable servers, broadband routers and other network equipment distributed all over the world. Because of the amplification feature of the reflection attack, attackers can generate massive attacks with small resources. Moreover, there are many booters who are selling DDoS attacks as a service. DDoS attacks are commoditized, so the frequency of DDoS attacks are also increasing.

These trends of attacks could exceed the capacity of a protection system of one organization in the aspect of volume and frequency. Therefore, sharing the capacity and capability of protection systems with each other to cope with such attacks becomes very necessary.

By utilizing other organization’s resources, the burden of the protection is shared. The shared resources are not only CPU/memory resources of dedicated mitigation devices but also the capability of mitigation actions such as blackholing and filtering. We call the protections which utilize shared resources "cooperative DDoS protection".

Cooperative DDoS protection has numerous merits. First, as described above, it can leverage expanded capacity of protection by sharing the resources among organizations. Generally DDoS attacks happen unexpectedly, thus the capacity utilization ratio of a protection system is not constant. So, while the utilization ratio is low, it can be used by another organization which is under attack. Second, organizations can implement various countermeasures. If an attack is highly sophisticated and there is no countermeasure in the victim’s system, cooperative DDoS protection can offer an optimal countermeasure for all partners. Third, it can block malicious traffic nearer to the origin of the attack. Near source defense is ideal for the health of the Internet because it can reduce the total cost of forwarding packets which, in the case of DDoS attacks mostly consist of useless massive attack traffic. Moreover, it is also very effective to solve the inter-domain uplink congestion problem. Finally, it can reduce the time to respond to an attack. After getting attacked, prompt response is important because outage of service can cause significant loss to the victim organization. This cooperating channel between partner organizations is automated by DOTS protocol.

The proposed solutions are covering both intra-domain and inter-domain situations. This standardized approach can utilize various protection systems in automated manner which can afford it quicker and more sophisticated protection in its domain.

1.1. Scope

The solutions described in this draft are based on intra-domain and inter-domain usecases in [I-D.draft-ietf-dots-use-cases]. The DOTS protocols coordinating DDoS protection in inter-domain situations in this draft are compliant with requirements in [I-D.draft-ietf-dots-requirements]. Generally DOTS is assumed to be most effective when aiding coordination of attack response between two or more organizations, but single domain scenarios are also valuable[I-D.draft-mortensen-dots-architecture]. The data model described in this draft is mainly focusing on inter-domain coordination of DDoS protection because it also covers single domain scenarios. The information required in single domain scenarios is assumed to be a subset of the information required in inter-domain scenarios.

2. Terminology

2.1. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2.2. Definition of Terms

This document uses the terms defined in [I-D.draft-ietf-dots-requirements].

3. Cooperative DDoS Protection Requirements

In this section, problems regarding cooperative DDoS protection are described.

3.1. Provisioning Requirements

In inter-domain situation, a DOTS client is in a different organization from a DOTS server. To enable the protection in other organization, provisioning information should be informed to a DOTS server in advance. In the later section, the total scenario is divided into two stages: provisioning stage and signaling stage. In provisioning stage, a DOTS client is required to communicate registration messages with DOTS server which include the capacity building of protection. The data model of registration message is

defined in the protocol section of this draft. It is also required to find a way to provision other organization's DDoS protection service in secure manner. All of the messages should have confidentiality, integrity and authenticity. The requirments of the message protocol is following [I-D.draft-ietf-dots-requirements].

3.1.1. Automatic Provisioning vs Manual Provisioning

Manual provisioning is an easier way to utilize DDoS protection service of other organizations. An organization can establish trust with another organization that is going to use its DDoS protection service by many different means such as phone, e-mail, Web portal, etc,. However, it will take much time to manually provision the DDoS protection system. Attacks that occur before the DDoS system is provisioned make significant impact on the protected service. To reduce the time to start the protection, automatic provisioning is desirable. If an organization could acquire relevant information of the DDoS protection service of other organization and utilize it by DOTS signaling in a short time, the cooperative DDoS protection will succeed at a certain level. Other important work carried out in the bootstrapping process are auto-discovery and automatic capability building between the member DDoS protection service providers as the basis for the following coordination process.

3.2. Coordination Requirements

The number of the member DDoS protection service providers that will be providing cooperative DDoS protection is an important factor. If only two providers are involved, there is a bilateral relationship only. It is easy to negotiate the capacity of their own DDoS protection system. In a state of emergency, one can decide to ask for help from the other if the capacity of its own system is insufficient. When a lot of providers are joining cooperative DDoS protection, it is difficult to decide where to ask for help. They need to negotiate the capacity with every participant. It is needed to take into account all combinations to do appropriate protection. The coordination between the member providers cooperative DDoS protection is a complete process consisting of mitigation start/stop, status notification, mitigation policy updates and so on. The Inter-domain DOTS architectures described in the later section are intended to fulfill these requirements.

In addition, inter-domain uplink congestion problems can only be solved by coordinating protection services provided by the upstream operators.

3.2.1. Near Source Protection Problem

Stopping malicious traffic at the nearest point on the Internet will reduce the exhaustion of resources in all paths of the attack. To find the entry point of the attack, traceback of the attack traffic to its origin is needed. If there is a cooperative partner near the attack source, asking for help blocking the malicious traffic from the ISP is most effective.

However, the problem is that it is difficult to decide which ISP is nearest to the attack source because in many cases source addresses of attack packets are spoofed to prevent the true source from being discovered. Moreover, some topology information of an ISP's network will need to be uncovered in order to make a correct decision, however there could be privacy protection issues between ISPs. These problems can lead to difficulties locating the true attack source. These problems can be divided into two issues. The first is how to find the attacker. The second is how to decide who to ask for help.

3.3. Returning Path Requirements

As one of protection methods, some DDoS protection service provider announce BGP route to detour the attack traffic to their own network to deal with it. After scrubbing, cleaned traffic is returned to the original destination. The returning path is often called a "clean pipe". The DDoS service provider should be careful about causing routing loops because if the end point of a clean pipe is still included in the reach of the announced BGP route, the traffic will return to the mitigation path again and again. When thinking about cooperative DDoS protection, returning path information should be propagated to partners.

4. Inter-domain DOTS Architecture

With the fast growth of DDoS attack volume and sophistication, a global cooperative DDoS protection service is desirable. This service can not only address the inter-domain uplink congestion problem, but also take full advantage of global DDoS mitigation resources from different operators efficiently and enable mitigation near the source of the attack. Moreover, with providing DDoS mitigation as service, more customers will get the service flexibly they demand with maximized territory and resources. Together with on-premise DDoS protection appliances, the multiple layer DDoS protection system provides a comprehensive DDoS protection against all types of attacks, such as application layer attacks, network layer large traffic attacks and others.

The DOTS protocol is used among DOTS agents to facilitate the coordinated DDoS protection service as a whole. [I-D.draft-ietf-dots-use-cases] lists most options that DOTS agents could be used for, and describes their communication. Although this document is initiated to specify the DOTS protocol for inter-domain use cases, the final protocol would and should be the same since it is all about the signaling messages and their process between the DOTS clients and DOTS servers essentially. In other words, the protocol described here would also apply to all the intra-domain use cases. To support all the identified use cases and possibly new use cases in the future, the DOTS protocol must be extensible in terms of the message definition, protocol process, etc., which will be discussed in detail in the following section. The text below discusses the protocol mainly in respect to inter-domain use cases.

The inter-domain DDoS protection service is set up by the member operators' own DDoS protection systems and the coordination protocol among them. The inter-domain protocol for the goal of DDoS protection coordination is the main focus of this document. Note that both network operators and cloud based DDoS protection service providers can participate in the inter-domain DDoS protection service. In general, the member operator's own DDoS protection system should at least consist of an attack detector, a customer (DOTS client), a controller (DOTS server, and possible DOTS client for the inter-domain use cases) and a mitigator.

Attack Detector: responsible for attack detection and source traceback. An example is the flow analyzer

Customer: when a DDoS attack is detected, it requests mitigation service to the controller and exchanges status with the controller regularly

Controller: responsible for intra-domain DDoS mitigation controlling and communication with the customer and other operators' controllers for inter-domain coordination

Mitigator: responsible for mitigation and results reporting

here are two ways for operators to implement the inter-domain DDoS protection service: distributed or centralized. The following sections discuss these architectures, aligning with DOTS terms.

4.1. Distributed Architecture

Operators can set up bilateral cooperative relationships of DDoS protection between each other, thereby a distributed inter-domain DDoS protection service is realized, which has peer to peer

communication among all the participating operators. The distributed architecture is illustrated in the following diagram:

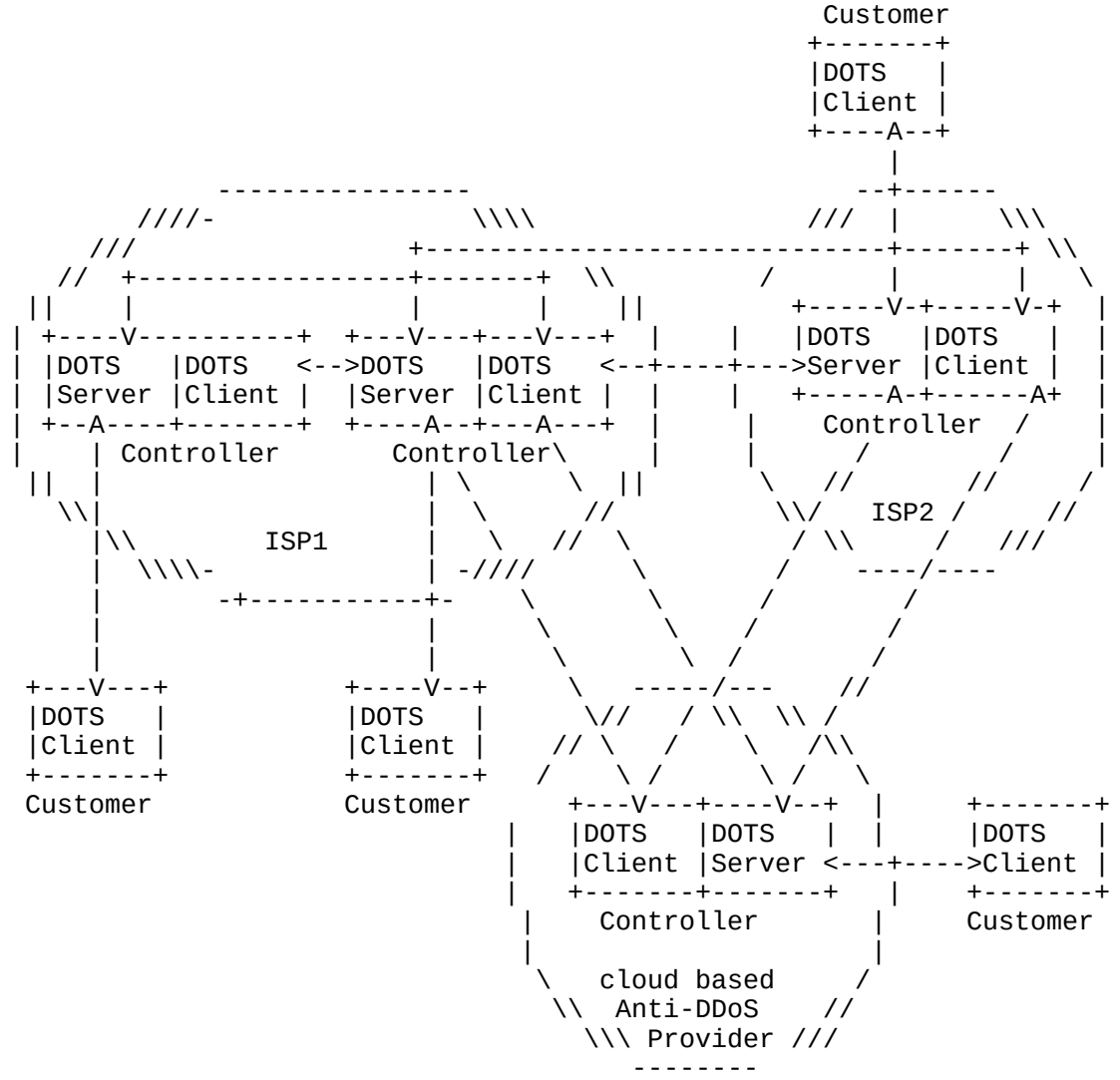


Figure 1: Distributed Architecture for Inter-domain DDoS Protection Service

As shown in the above diagram, when a customer is suffering a large traffic DDoS attack, it acts as the DOTS client to request DDoS protection service from its operator. The operator’s controller acts as the DOTS server to authenticate the customer’s validity and then

initiate the intra-domain DDoS mitigation service with its own resource for the customer. If the controller finds the attack volume exceeds its capacity, or the attack type is unknown type, or its inter-domain upstream link is congested, it should act as the DOTS client to request inter-domain coordinated DDoS Protection service to its upstream operators' controllers which it has cooperative relationship with. The operator's controller should support the functions of DOTS server and DOTS client at the same time in order to participate in the system of inter-domain DDoS protection service. In other words, as the representative for an operator's DDoS protection service, the controller manages and provides DDoS mitigation service to its customer on one hand, but on the other, it may require help from other operators under other situations, especially when the attack volume exceeds its capacity or the attack is from other operators. The inter-domain coordination can be a repeated process until the operator nearest the attack source receives the inter-domain coordination request and starts to mitigate the attack traffic.

In particular, each operator is able to decide its own responding actions to its peering operator's request flexibly by its internal policies, such as whether or not perform the mitigation function, or relay the request message to other operators. These other scenarios are out of the scope of this document.

The distributed architecture is straightforward and simple when the number of member operators are not too large. For deployment, all the work an operator needs to do is to configure other cooperative member operator's information (i.e., IP, port, DNS name, etc) and relevant policies for subsequent inter-domain communication. Regarding operation, each operator's controller only performs the mitigation service according to customer's request and possibly requests for inter-domain help to other operators if necessary. In the meantime, the mitigation report and statistics information is exchanged between the peering operators for the purpose of monitoring and accounting.

Some points for this architecture are noted below:

- Every operator controller only has the information of those operators which have cooperative relation with it, and does not necessarily have the information of all operators participating in the inter-domain DDoS protection service. The incomplete information may not lead to the most optimized operation.

- When the number of member operators is very large, a new joining operator will be required to configure and maintain a large number

of peering operators' information. This can be very complex and error-prone.

- Due to the exclusive repeating nature of the architecture mentioned above, it's possible that a really effective mitigation service by one upstream operator starts only after several rounds of repeating the inter-domain coordination process. This process may take a long time and is unacceptable.

4.2. Centralized Architecture

For the centralized architecture, the biggest difference from the distributed architecture is that a centralized orchestrator exists for controlling the inter-domain DDoS protection coordination centrally. This centralized architecture for the inter-domain DDoS protection service is illustrated in the following diagram:

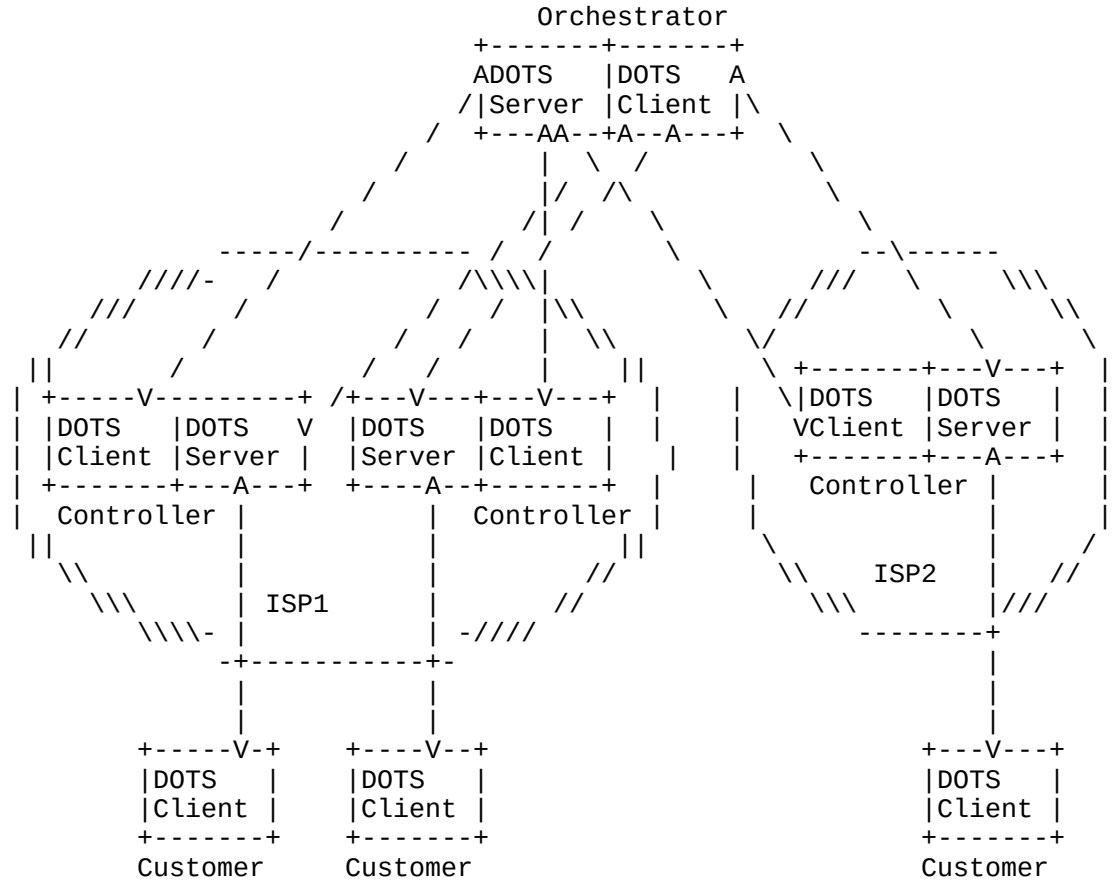


Figure 2: Centralized Architecture for Inter-domain DDoS Protection Service

As shown in above diagram, the orchestrator is the core component to the whole system. Each operator controller only communicates with it for the goal of registering, coordination requesting and reporting. When it receives the inter-domain coordination request message from the operator controller, a simple way is to notify all the other operator controllers that have registered to the orchestrator, to enable possible mitigation services. Another way is to choose a number of operators to notify them to enable the mitigation services according to the traceback result or other policies. The details about traceback are to be discussed in future. Based on the above analysis, the orchestrator is also a combination of a DOTS server and a DOTS client which supports both functions at the same time.

In addition to the orchestrator and its related functions, the signaling and operations of the centralized architecture are very similar to the distributed architecture.

The centralized architecture has its own unique characteristics described below:

- Since this is a centralized architecture, it is easy for the orchestrator to suffer a single failure problem like failure, congestion or performance downgrade, which would directly influence the availability of the whole system. This issue can be improved somewhat by implementing some redundancy mechanisms.

- A centralized orchestrator facilitates the auto-discovery mechanism for the member operators. And for each controller, its deployment and operation becomes easy since it is only required to communicate with the orchestrator during the whole process.

- Due to the direct communication between the orchestrator and all controllers, the inter-domain DDoS coordination is able to be finished in a short and fixed time period.

- Only the central orchestrator is required to support different transport protocols (e.g., TCP, UDP, CoAP) to communicate with all the controllers. The orchestrator is able to translate and relay different transport protocols among all the operators. So, the operator controller uses one transport protocol to communicate with orchestrator and is not required to support multiple kinds of transport protocols.

5. Inter-domain DOTS Protocol

According to [I-D.draft-ietf-dots-requirements], DOTS protocols MUST take steps to protect the confidentiality, integrity and authenticity of messages sent between the DOTS client and server, and provide a peer mutual authentication between the DOTS client and server before a DOTS session is considered active. The DOTS agents can use HTTPS (with TLS) for the goal of protocol security. The HTTP RESTful APIs are used in this section as the protocol channel, and the DOTS message content can be in JSON format.

With respect to the inter-domain DOTS protocol, all the DOTS messages are exchanged between DOTS client and server, no matter what the architecture (distributed or centralized) is. Therefore, the message formats and operations of the DOTS protocol ought to be the same for all architecture options. The DOTS messages can be categorized by which time period they are mainly required for DDoS protection, as below:

- Provisioning stage: Before being attacked by malicious traffic, a DOTS client should register itself to the DOTS server, as well as enable capacity building in advance.

- Signaling stage: Once the DOTS client has registered itself to the DOTS server, the DOTS session is created between client and server and the signaling stage begins. The signaling stage ends when the DOTS client cancels its registration to the DOTS server and the DOTS session is closed. During the signaling stage, the DOTS client should ask the DOTS server for DDoS mitigation service to the customer service under attack once an attack is detected. When an attack is over, the DOTS server should notify the DOTS client.

DOTS protocol can run on HTTPS (with TLS) and support several different ways for authentication:

- Employ bidirectional certificate authentication ([ITU-T X.509]) on the DOTS server and the client: Both DOTS server and client MUST verify the certificates of each other.

- Employ unidirectional certificate authentication ([ITU-T X.509]) on the DOTS server: Only the DOTS server needs to install the certificate. The DOTS client only needs to verify its certificate. In the opposite direction, the DOTS server can authenticate the DOTS client by the ways of a user/role:password, IP address white-list or digital signature.

- Employ bidirectional digital signature authentication on the DOTS server and client: In this scenario, the DOTS server and client must keep the customer’s private key safe. This private key is used to generate the digital signature.

Besides authenticating the DOTS client, the DOTS server also verifies the timestamp of the packets from the DOTS client. If the time difference between the timestamp and the current time of the DOTS server exceeds a specified threshold (60 seconds as an example), the DOTS server will consider the packet invalid and will not process it. Therefore, NTP must be configured on both the DOTS server and client to ensure time synchronization. This method can effectively protect the DOTS server against a replay attack.

The following sections present detailed description of all the DOTS messages for each stage, and the relevant DOTS protocol operations.

5.1. Provisioning Stage

In the provisioning stage, a DOTS client can be located either in the customer side, in the operator controller, or in the inter-domain orchestrator (for the centralized architecture). In any case, the DOTS client should register itself to its peering DOTS server which provides the intra/inter domain DDoS mitigation service to it in order to set up the DOTS protocol session. More importantly, the registration process also facilitates the auto-discovery, capacity building and configuration between the DOTS client and server.

5.1.1. Messages

In the provisioning stage, the messages of registration (DOTS client to server), registration response (DOTS server to client), registration cancelling (DOTS client to server) and registration cancelling response (DOTS server to client) are required. Since all the messages in this stage are not expected to be used under the DDoS attack conditions, transmitting all the messages through DOTS data channel over TLS is able to meet the requirements of reliability, privacy and integrity.

The HTTP POST method with the message body in JSON format is used for the registration and registration response messages as below:

```
METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/registration
registration body:
{
 "customer_name": string;
 "ip_version": string;
 "protected_zone": {
      "index": number;
      "need_alias": string;
      "ipv4_CIDR": string;
      "ipv6_address": string;
      "BGP_route": string;
      "SIP_URI": string;
      "E164_number": string;
      "DNS_name": string;
 }
 "protected_port": string;
 "protected_protocol": string;
 "countermeasures": string;
 "tunnel_information": string;
 "next_hop": string;
 "security_profile": {
     "TLS": string;
     "DTLS": string;
```

```
            "CoAP": string;
        }
        "white_list": {
            "name": string;
            "sequence_number": string;
            "source_ip": string;
            "destination_ip": string;
            "source_port": string;
            "destination_port": string;
            "protocol": string;
            "length": string;
            "TTL": string;
            "DSCP": number;
            "ip_flags": number;
            "tcp_flags": number;
        }
        "black_list": {
            "name": string;
            "sequence_number": string;
            "source_ip": string;
            "destination_ip": string;
            "source_port": string;
            "destination_port": string;
            "protocol": string;
            "length": string;
            "TTL": string;
            "DSCP": number;
            "ip_flags": number;
            "tcp_flags": number;
        }
       }
       registration response body:
       {
        "customer_name": string;
        "customer_id": string;
        "alias_of_mitigation_address": {
            "index": number;
            "alias": string;
        }
        "security_profile": string;
        "access_token": string;
        "thresholds_bps": number;
        "thresholds_pps": number;
        "duration": number;
        "capable_attack_type": string;
        "registration_time": string;
        "mitigation_status": string;
        }
```

Registration body:
customer_name: The name of the customer (DOTS client);
ip_version: Current IP version. It can be "v4" or "v6";
protected_zone: Limit the address range of protection. Especially it will be limited to the prefixes possessed by the customer;
index: index of the protected zone;
need_alias: the flag representing if this protected zone needs an alias. "true" represents that the alias is needed, "false" represents the opposite side;
ipv4_CIDR: ipv4 CIDR address or prefix scope of the protected zone;
ipv6_address: ipv6 address or prefix scope of the protected zone;
BGP_route: BGP route of the protected zone;
SIP_URI: SIP URI of the protected zone;
E164_number: E.164 number of the protected zone;
DNS_name: DNS name of the protected zone;
protected_port: Limit the port range of protection, "all" represents all the ports are to be protected;
protected_protocol: The protected protocol indicated by the protocol attribute in the IP packet header, "all" represents all the protocols are to be protected;
countermeasures: Some of the protection need mitigation and others need blackholing;
tunnel_information: The tunnel between the mitigation provider’s network and the customer’s network. Tunnel technologies such as GRE[RFC2784] can be used to
return normal traffic. "null" represents there is no tunnel information provided and the DOTS server can decide the return tunnel for the
normal traffic for itself;
next_hop: The returning path to customer’s network. "null" represents there is no next hop information provided and the DOTS server can decide it for itself;
security_profile: The security profile in transport layer for the DOTS signaling channel that DOTS client supports;
TLS: "true" represents that the DOTS client supports TLS over TCP, "false" represents the opposite side;
DTLS: "true" represents that the DOTS client supports DTLS over UDP, "false" represents the opposite side;
CoAP: "true" represents that the DOTS client supports CoAP, "false" represents the opposite side;
white_list: The white-list information provided to the DOTS server;
name: Name of the white-list;
sequence_number: Sequence number of the white-list;
source_ip: The source IP address attribute used in the white-list;
destination_ip: The destination IP address attribute used in the white-list;
source_port: The source port attribute used in the white-list;
destination_port": The destination port attribute used in the white-list;
protocol: The protocol attribute in the IP packet header used in the white-list;
length: The length attribute in the IP packet header used in the white-list;
TTL: The TTL attribute in the IP packet header used in the white-list;
DSCP: The DSCP attribute in the IP packet header used in the white-list;
ip_flags: The IP flags attribute used in the white-list;
tcp_flags: The TCP flags attribute used in the white-list;

black_list: The black-list information provided to the DOTS server;
name: Name of the black-list;
sequence_number: Sequence number of the black-list;
source_ip: The source IP address attribute used in the black-list;
destination_ip: The destination IP address attribute used in the black-list;

source_port: The source port attribute used in the black-list;
destination_port: The destination port attribute used in the black-list;
protocol: The protocol attribute in the IP packet header used in the black-list;
length: The length attribute in the IP packet header used in the black-list;
TTL: The TTL attribute in the IP packet header used in the black-list;
DSCP: The DSCP attribute in the IP packet header used in the black-list;

ip_flags: The IP flags attribute used in the black-list;
tcp_flags: The TCP flags attribute used in the black-list;

registration response body:
customer_name: The name of the customer (DOTS client);
customer_id: The unique id of the customer (DOTS client);
alias_of_mitigation_address:
index: index of the protected zone;
alias: The alias that the DOTS server assigns to this protected zone;
security_profile: The negotiated security profile for the DOTS session;
access_token: Authentication token (e.g. pre-shared nonce). "null" represe
nts there is no access token;
thresholds_bps: If an attack volume is over this threshold, the controller
will reject the protection in order to comply with the negotiated contract;
thresholds_pps: If an attack volume is over this threshold, the controller
will reject the protection in order to comply with the negotiated contract;
duration: If an attack is longer than this threshold, the controller will
reject the protection in order to comply with the negotiated contract;
capable_attack_type: Limit the protectable attack type;
registration_time: The time of registration;
mitigation_status: The status of current mitigation service of the ISP.

Similarly, another HTTP POST method with the message body in JSON format is used for the registration cancelling and registration cancelling response messages are described below:

METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/registration_cancelling
registration cancelling body:
{
"customer_id": string;
"reasons": string;
}
registration cancelling response body:
{
"customer_id": string;
"result": string;
}

Registration cancelling body:
customer_id: The unique id of the customer (DOTS client);
reasons: The reasons why the DOTS client cancelled the registration;

registration cancelling response body:
customer_id: The unique id of the customer (DOTS client);
result: The final result which defines whether or not the DOTS controller
accepts the registration cancelling request.

5.1.2. Operations

The main operations in the provisioning stage include:

- The customers (DOTS client) registers to the operator controller with the configuration and capability building including protection methods, process capacity, protected zone, security profile, white/black-list, etc;

- The DOTS client in operator controller registers to the DOTS server in inter-domain orchestrator (centralized architecture) or other operator controllers (distributed architecture) according to inter-domain DDoS protection requirements;

- The DOTS client can send the registration cancelling message to the DOTS server for cancelling its DDoS protection service.

The DOTS server indicates the result of processing the POST request using HTTP response codes:

- Response code 200 (OK) will be returned in the response if the DOTS server has accepted the mitigation request and will try to mitigate the attack. The HTTP response will include the JSON body of response messages specified above;

- If the request is missing one or more mandatory attributes then 400 (Bad Request) will be returned in the response or if the request contains invalid or unknown parameters then 500 (Invalid query) will be returned in the response. The HTTP response will include the JSON body received in the request, with an extra attribute to represent the specific error reason:

```
"error_reason": number;
   0: Bad Request;
   1: Invalid Query;
   2: Server Error;
   3: Protected Zone Confliction;
   4: Countermeasure Not Supported;
   5: Security Profile Not Supported;
   6: Confliction Exists for White-list or Black-list;
   255: Others;
```

5.2. Signaling Stage

During the signaling stage, the DOTS signaling channel created with the negotiated security profile in the provisioning stage is used for the DDoS attack mitigation coordination. Once the DOTS client detects the attack to the customer service, a mitigation initiation request message is created and sent to the provisioned DOTS server to call for the DDoS protection service. The DOTS server decides to protect the customer service based on the information from the request message and its configured policy. One operator’s DOTS

server may ask the co-located DOTS client to resume sending the mitigation initiation request message to other operators' DOTS server to request the inter-domain coordinated mitigation service while it isn't able to deal with the attack by itself. Meanwhile, some other messages are required to be communicated between the DOTS client and server for information updates about status, efficacy and scope. When the DOTS server is informed from the mitigator that the attack is over, it should notify the DOTS client to terminate the mitigation service.

5.2.1. Messages

In the signaling stage, the DOTS signaling channel is expected to transmit DOTS messages under extremely hostile network conditions such as link saturation. To meet the requirements of resilience and robustness, unidirectional messages MUST be supported within the bidirectional signal channel to allow for unsolicited message delivery, enabling asynchronous notifications between the DOTS client and server. So, the listed DOTS messages are required: mitigation initiation request (DOTS client to server), mitigation efficacy updates (DOTS client to server), mitigation status updates (DOTS server to client), mitigation termination (DOTS client to server), mitigation termination status acknowledgement (DOTS client to server) and heartbeat (bidirectional message).

Mitigation Request:

A HTTP POST method with the message body in JSON is used for the mitigation request message:

```
METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/mitigation_request
  mitigation request body:
  {
   "version": string;
   "type": string;
   "alert_id": string;
   "sender_id": string;
   "sender_asn": string;
   "mitigation_action": number;
   "lifetime": number;
   "max_bandwidth": number;
   "packet_header": {
       "dst_ip": string;
       "alias": string;
       "dst_ports": string;
       "src_ips": string;
       "src_ports": string;
       "protocols": string;
```

```
            "tcp_flags": string;
            "fragment": string;
            "pkt_len": string;
            "icmp_type": string;
            "icmp_code": string;
            "DSCP": string;
            "TTL": string;
          }
          "current_throughputs": {
            "bps": string;
            "pps": string;
          }
          "peak_throughputs": {
            "bps": string;
            "pps": string;
          }
          "average_throughputs": {
            "bps": string;
            "pps": string;
          }
          "info": {
            "attack_types": string;
            "started": number;
            "ongoing": number;
            "severity": number;
            "direction": number;
            "health": number;
          }
          "vendor": {
            "name": string;
            "version": string;
            "payload": {
                   "offset": number;
                   "content": string;
                   "hash": string;
            }
           }
         }
```

mitigation request body:
version: A 3 digit set, similar to Linux. (Major.Minor.Revision);
type: Only "attack" in scope for v1;
alert_id: A SHA-256 hash that is derived from DST_IP and started with some random nonce;
sender_id: A SHA-256 hash signature of the sender. This is used to validate who sent it;
sender_asn: ASN of the sender. This could be used to link back to sender_id to validate the sender of being a valid sender_id;
mitigation_action: The requested mitigation actions by DOTS client.

Possible value could be: 1 - mitigation, 2 - blackhole, 3 - flowspec, ...;
lifetime: The desired lifetime of the mitigation service from the DOTS client. Upon the expiry of this lifetime, and if the request is not refreshed, the mitigation service is stopped. The service can be refreshed by sending the message with the same "alert_id" again. A lifetime of zero indicates indefinite lifetime for the mitigation service. This is an optional attribute in the request message;
max_bandwidth: The max bandwidth the DOTS client can undertake. The unit is "G bytes";
packet_header: IP packet header contents used for a report. CSV (Comma Separated Values) format is used here when multiple values are possible. Note that no spaces between commas for CSV format, and the multiple values for every attribute should be in the same order as they are assigned.
dst_ip: A single IP under attack;
alias: The DOTS client's registered alias for the protected zone;
dst_ports: The destination port(s) used for the attack. CSV formatted;
src_ips: The list of source IPs of the attack. CSV formatted;
src_ports: The source port(s) used for the attack. CSV formatted;
protocols: The IP protocol numbers used for the attack. The list of IP protocol numbers are defined and maintained by IANA. CSV formatted;
tcp_flags: The TCP flags used for the attack. Possible value could be: SYN, FIN, ACK, PSH, RST, URG, NULL. CSV formatted;
fragment: The fragment flags in the IP header for the attack. Possible value could be: DF - Don't fragment, IsF - Is a fragment, FF - First fragment, LF - Last fragment. CSV formatted;
pkt_len: The packet length used for the attack. CSV formatted;
icmp_type: The icmp type used for the attack. CSV formatted;
icmp_code: The icmp code used for the attack. CSV formatted;
DSCP: The DSCP value used for the attack. CSV formatted;
TTL: The TTL value used for the attack. CSV formatted;
current_throughputs: Current throughput in bps/pps for the above attack flows
bps: bytes per second. CSV formatted;
pps: packets per second. CSV formatted;
peak_throughputs: The peak throughput in bps/pps for the above attack flows until the time the DOTS request message is sent
bps: bytes per second. CSV formatted;
pps: packets per second. CSV formatted;
average_throughputs: The calculated average throughput in bps/pps for the above attack flows until the time the DOTS request message is sent
bps: bytes per second. CSV formatted;
pps: packets per second. CSV formatted;
info: Other general information which may be useful
attack_types: List of attacks being used together for this attack, on this single DST_IP. CSV formatted;
started: Unix EPOCH when the attack is started;
ongoing: The value representing whether the attack is still ongoning. 1 - yes, 0 - no;
severity: The severity level of the attack. 1, 2, 3 - low, medium, high;
direction: The direction of the attack. in or out;
health: The health condition of the DOTS client. 0-100;
vendor:
name: Company name;
version: version of the DOTS client on the vendors device;
payload: The attack packet payload provided to DOTS server for further analysis

offset: The payload offset;
content: The payload content that is base64 encoded;
hash: A SHA-256 hash used as a checksum, of the original payload before being base64 encoded. This is to proof the payload is complete. Not to prove if
it has been tampered with;

Mitigation Status Exchange:

A HTTP POST method with the message body in JSON is used for the mitigation efficacy updates message:

```
      METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/mitigation_efficacy_u
pdates
        mitigation efficacy updates body:
        {
         "version": string;
         "alert_id": string;
         "sender_id": string;
         "sender_asn": string;
         "attack_status": string;
         "health": number;
        }

        mitigation efficacy updates body:
        version: A 3 digit set, similar to linux.  (Major.Minor.Revision);
        alert_id: A SHA-256 hash that is derived from DST_IP and started with s
ome random nonce;
        sender_id: A SHA-256 hash signature of the sender. This is used to vali
date who sent it;
        sender_asn: ASN of the sender. Could be used to link back to sender_id
to validate the sender of being a valid sender_id;
        attack_status: The current attack status of the DOTS client. Possible v
alue could be: 0 - in-process, 1 - terminated;
        health: The health condition of the DOTS client. 0-100;
```

A HTTP POST method with the message body in JSON is used for the mitigation status updates message:

```
     METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/mitigation_status_upda
tes
     mitigation status updates body:
    {
     "version": string;
     "alert_id": string;
     "sender_id": string;
     "sender_asn": string;
     "status": number;
     "error_reason": number;
     "lifetime": number;
     "source_ports": string;
     "destination_ports": string;
     "source_ips": string;
     "destination_ip": string;
     "TCP_flags": string;
     "start_time": number;
     "end_time": number;
     "forwarded_total_packets": number;
     "forwarded_total_bits": number;
     "forwarded_peak_pps": number;
     "forwarded_peak_bps": number;
     "forwarded_average_pps": number;
     "forwarded_average_bps": number;
```

```
      "malicious_total_packets": number;
      "malicious_total_bits": number;
      "malicious_peak_pps": number;
      "malicious_peak_bps": number;
      "malicious_average_pps": number;
      "malicious_average_bps": number;
      "record_time": string;
      }
```

mitigation status updates body:
version: A 3 digit set, similar to Linux. (Major.Minor.Revision);
alert_id: A SHA-256 hash that is derived from DST_IP and started with some random nonce;
sender_id: A SHA-256 hash signature of the sender. This is used to validate who sent it. The sender is the DOTS server for this message;
sender_asn: ASN of the sender. This could be used to link back to sender_id to validate the sender of being a valid sender_id;
status: Current mitigation status, such as: pending, ongoing, done, error;
error_reason: If status attribute is error, then this attribute expresses its reason, the possible value could be: 0 - Bad Request, 1 - Server Error, 3 - Mitigation Scope Confliction, 4 - Mitigation Action Not Support, 255 - Others;
lifetime: The lifetime of mitigation service that DOTS server has assigned to DOTS client. DOTS client MUST follow this value;
source_ports: For TCP or UDP or SCTP or DCCP: the source range of ports (e.g., 1024-65535) of the discarded traffic. CSV formatted;
destination_ports: For TCP or UDP or SCTP or DCCP: the destination range of ports (e.g., 1-443) of the discarded traffic. CSV formatted;
source_ips: The source IP addresses or prefixes of the discarded traffic. CSV formatted;
destination_ip: The destination IP addresses or prefixes of the discarded traffic;
TCP_flags: TCP flag of the discarded traffic. CSV formatted;
start_time: The start time for the duration of this mitigation status message;
end_time: The end time for the duration of this mitigation status message;
forwarded_total_packets: The total number of packets forwarded;
forwarded_total_bits: The total bits for all the packets forwarded;
forwarded_peak_pps: The peak pps of the traffic forwarded;
forwarded_peak_bps: The peak bps of the traffic forwarded;
forwarded_average_pps: The average pps of the traffic forwarded;
forwarded_average_bps: The average bps of the traffic forwarded;
malicious_total_packets: The total number of malicious packets;
malicious_total_bits: The total bits of malicious packets;
malicious_peak_pps: The peak pps of the malicious traffic;
malicious_peak_bps: The peak bps of the malicious traffic;
malicious_average_pps: The average pps of the malicious traffic;
malicious_average_bps: The average bps of the malicious traffic;
record_time: The time that the mitigation status updates message is created;

Mitigation Termination:

A HTTP POST method with the message body in JSON is used for the mitigation termination request message:

```
      METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/mitigation_termination
_request
         mitigation termination request body:
         {
          "version": string;
          "alert_id": string;
          "sender_id": string;
          "sender_asn": string;
         }

      mitigation termination request body:
      version: A 3 digit set, similar to linux.  (Major.Minor.Revision);
      alert_id: A SHA-256 hash that is derived from DST_IP and started with some
 random nonce;
      sender_id: A SHA-256 hash signature of the sender. This is used to validat
e who sent it;
      sender_asn: ASN of the sender. This could be used to link back to sender_i
d to validate the sender of being a valid sender_id;
```

A HTTP POST method with the message body in JSON is used for the mitigation termination status acknowledgement message:

```
        METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/mitigation_terminati
on_status_acknowledgement
         mitigation termination status acknowledgement body:
         {
          "version": string;
          "alert_id": string;
          "sender_id": string;
          "sender_asn": string;
         }

        mitigation termination status acknowledgement body:
        version: A 3 digit set, similar to Linux.  (Major.Minor.Revision);
        alert_id: A SHA-256 hash that is derived from DST_IP and started with so
me random nonce;
        sender_id: A SHA-256 hash signature of the sender. This is used to valid
ate who sent it;
        sender_asn: ASN of the sender. This could be used to link back to sender
_id to validate the sender of being a valid sender_id;
```

Heartbeat:

A HTTP POST method with the message body in JSON is used for the heartbeat message:

```
       METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/heartbeat
         heartbeat body
         {
          "version": string;
          "sender_id": string;
          "sender_asn": string;
         }

        heartbeat body:
        version: A 3 digit set, similar to Linux.  (Major.Minor.Revision);
        sender_id: A SHA-256 hash signature of the sender. This is used to valid
ate who sent it;
        sender_asn: ASN of the sender. This could be used to link back to sender
_id to validate the sender of being a valid sender_id;
```

5.2.2. Operations

The main operations in the signaling stage include:

- The customer (DOTS client) detects a malicious attack, requests mitigation service to its operator controller (DOTS server);
- DOTS server authenticates and provides its intra- domain mitigation service to the DOTS client;
- When the DOTS server is mitigating the attack but finds that the attack volume exceeds its capacity, or the attack type is an unknown type, or its upstream link is congested, it should request to other DOTS servers for inter-domain cooperation;
- Working DOTS server report their statistics results by mitigation status updates message to the DOTS client;
- The DOTS client can updates its mitigation scope to the DOTS server by resending the mitigation request message. It also can update its mitigation efficacy result to the DOTS server;
- When the DOTS server is informed from the mitigator that the attack is over, it should notify the DOTS client by the mitigation status updates message to terminate the mitigation service;
- When the DOTS client is notified by the DOTS server to terminate its mitigation service, it should send a DOTS termination request message to the DOTS server. The DOTS server stops its mitigation service and notifies it to the DOTS client by sending DOTS status updates message. At last, the DOTS client sends a DOTS mitigation termination acknowledgement message to finish the whole DOTS session;

- The heartbeat message is exchanged between the DOTS client and the DOTS server to check their respective status. If any side of the channel fails to receive the heartbeat message, then it will trigger an alert or further investigation into why the heartbeats never reached their destination.

6. Other Considerations

6.1. Billing Data

This is not a technical issue nor a part of the DOTS protocol but it is relevant to deployment models. If other organizations utilized the resources of a DDoS protection service, it is natural to charge it according to the amount of use. However, how does one count the amount of use among different DDoS protection service providers. For example, some DDoS protection service provider charges users by volume of the attack traffic or dropped packets. On the other hand, some of them use the volume of normal traffic. The number of executions can be also used. We cannot decide what information should be taken into account for billing purposes in advance, however information is needed to be exchanged while coordinating DDoS protection. This information could be also used to determine which service would be used when asking for help. Though it is out of the scope for DOTS, coordinating and optimizing this cooperation this business aspect is difficult to solve.

7. Security Considerations

TBD

8. IANA Considerations

No need to describe any request regarding number assignment.

9. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2784] D. Farinacci., T. Li., S. Hanks., D. Meyer., and P. Traina., "Generic Routing Encapsulation (GRE), March 2000".

[I-D.draft-ietf-dots-use-cases]
R. Dobbins, Ed., S. Fouant., D. Migault., R. Moskowitz., N. Teague., L. Xia, "Use cases for DDoS Open Threat Signaling, October 2015".

[I-D.draft-ietf-dots-requirements]
A. Mortensen., R. Moskowitz., and T. Reddy., "DDoS Open Threat Signaling Requirements, draft-ietf-dots-requirements-00, October 2015".

[I-D.draft-mortensen-dots-architecture]
A. Mortensen., F. Andreasen., T. Reddy., C. Gray., R. Compton., and N. Teague., "Distributed-Denial-of-Service (DDoS) Open Threat Signaling Architecture, March 2016".

[I-D.draft-reddy-dots-transport]
T. Reddy., D. Wing., P. Patil., M. Geller., M. Boucadair., and R. Moskowitz., "Co-operative DDoS Mitigation, June 2016".

Authors' Addresses

Kaname Nishizuka
NTT Communications
GranPark 16F
3-4-1 Shibaura, Minato-ku, Tokyo
108-8118,Japan

EMail: kaname@nttv6.jp

Liang Xia
Huawei Technologies Co., Ltd.
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu
210012, China

EMail: frank.xialiang@huawei.com

Jinwei Xia
Huawei Technologies Co., Ltd.
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu
210012, China

EMail: xiajinwei@huawei.com

Dacheng Zhang
Beijing
China

EMail: dacheng.zdc@alibaba-inc.com

Luyuan Fang
Microsoft
15590 NE 31st St
Redmond, WA 98052

EMail: lufang@microsoft.com

Christopher Gray
Comcast, Inc.
United States

EMail: Christopher_Gray3@cable.comcast.com

Rich Compton
Charter Communications, Inc.
14810 Grasslands Dr
Englewood, CO 80112

EMail: Rich.Compton@charter.com

DOTS
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2017

T. Reddy
D. Wing
P. Patil
M. Geller
Cisco
M. Boucadair
Orange
R. Moskowitz
HTT Consulting
July 6, 2016

Co-operative DDoS Mitigation
draft-reddy-dots-transport-05

Abstract

This document specifies a mechanism that a DOTS client can use to signal that a network is under a Distributed Denial-of-Service (DDoS) attack to an upstream DOTS server so that appropriate mitigation actions are undertaken (including, blackhole, drop, rate-limit, or add to watch list) on the suspect traffic. The document specifies both DOTS signal and data channels. Happy Eyeballs considerations for the DOTS signal channel are also elaborated.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 7, 2017.

Copyright Notice

Table of Contents

1. Introduction

A distributed denial-of-service (DDoS) attack is an attempt to make machines or network resources unavailable to their intended users. In most cases, sufficient scale can be achieved by compromising enough end-hosts and using those infected hosts to perpetrate and amplify the attack. The victim in this attack can be an application server, a client, a router, a firewall, or an entire network, etc.

In a lot of cases, it may not be possible for an enterprise to determine the cause for an attack, but instead just realize that certain resources seem to be under attack. The document proposes that, in such cases, the DOTS client just inform the DOTS server that the enterprise is under a potential attack and that the Mitigator monitor traffic to the enterprise to mitigate any possible attack. This document also describes a means for an enterprise, which act as DOTS clients, to dynamically inform its DOTS server of the IP addresses or prefixes that are causing DDoS. A Mitigator can use this information to discard flows from such IP addresses reaching the customer network.

The proposed mechanism can also be used between applications from various vendors that are deployed within the same network, some of them are responsible for monitoring and detecting attacks while others are responsible for enforcing policies on appropriate network elements. This cooperations contributes to a ensure a highly automated network that is also robust, reliable and secure. The advantage of this mechanism is that the DOTS server can provide protection to the DOTS client from bandwidth-saturating DDoS traffic.

How a Mitigator determines which network elements should be modified to install appropriate filtering rules is out of scope. A variety of mechanisms and protocols (including NETCONF) may be considered to exchange information through a communication interface between the server and these underlying elements; the selection of appropriate mechanisms and protocols to be invoked for that interfaces is deployment-specific.

Terminology and protocol requirements for co-operative DDoS mitigation are obtained from [I-D.ietf-dots-requirements].

2. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Solution Overview

Network applications have finite resources like CPU cycles, number of processes or threads they can create and use, maximum number of simultaneous connections it can handle, limited resources of the control plane, etc. When processing network traffic, such an application uses these resources to offer its intended task in the most efficient fashion. However, an attacker may be able to prevent the application from performing its intended task by causing the application to exhaust the finite supply of a specific resource.

TCP DDoS SYN-flood, for example, is a memory-exhaustion attack on the victim and ACK-flood is a CPU exhaustion attack on the victim ([RFC4987]). Attacks on the link are carried out by sending enough traffic such that the link becomes excessively congested, and legitimate traffic suffers high packet loss. Stateful firewalls can also be attacked by sending traffic that causes the firewall to hold excessive state and the firewall runs out of memory, and can no longer instantiate the state required to pass legitimate flows. Other possible DDoS attacks are discussed in [RFC4732].

In each of the cases described above, some of the possible arrangements to mitigate the attack are:

- If a DOTS client determines it is under an attack, the DOTS client can notify the DOTS server using the DOTS signal that it is under a potential attack and request that the DOTS server take precautionary measures to mitigate the attack. The DOTS server can enable mitigation on behalf of the DOTS client by communicating the DOTS client's request to the mitigator and relaying any mitigator feedback to the requesting DOTS client.

- If a DOTS client determines it is under an attack, the DOTS client can notify its servicing router (DOTS gateway) using the DOTS signal that it is under a potential attack and request that the DOTS gateway take precautionary measures to mitigate the attack. The DOTS gateway propagates the DOTS signal to a DOTS server.

  The DOTS server can enable mitigation on behalf of the DOTS gateway by communicating the DOTS gateway's request to the mitigator and relaying any mitigator feedback to the DOTS gateway which in turn propagates the feedback to the requesting DOTS client.

  The DOTS client must authenticate itself to the DOTS gateway, which in turn authenticates itself to a DOTS server, creating a two-link chain of transitive authentication between the DOTS client and the DOTS server (Section 8).

- If a network resource detects a potential DDoS attack from a set of IP addresses, the network resource (DOTS client) informs its servicing router (DOTS gateway) of all suspect IP addresses that need to be blocked or black-listed for further investigation.

  The DOTS client could also specify a list of protocols and ports in the black-list rule. That DOTS gateway in-turn propagates the black-listed IP addresses to the DOTS server and the DOTS server blocks traffic from these IP addresses to the DOTS client thus reducing the effectiveness of the attack.

The DOTS client periodically queries the DOTS server to check the counters mitigating the attack. If the DOTS client receives a response that the counters have not incremented then it can instruct the black-list rules to be removed. If a blacklisted IPv4 address is shared by multiple subscribers, then the side effect of applying the black-list rule will be that traffic from non-attackers will also be blocked by the access network [RFC6269].

An example of network diagram showing a deployment of these elements is shown in Figure 1. In this example, the DOTS server operating on the access network.

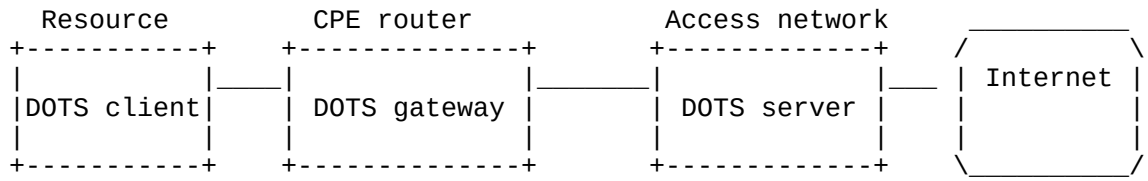


Figure 1

The DOTS server can also be running on the Internet, as depicted in Figure 2.

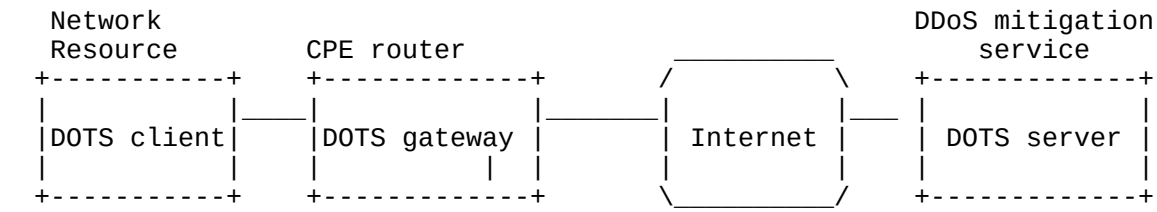


Figure 2

In typical deployments, the DOTS client belongs to a different administrative domain than the DOTS server. For example, the DOTS client is a web server serving content owned and operated by an domain, while the DOTS server is owned and operated by a different domain providing DDoS mitigation services. That domain providing DDoS mitigation service might, or might not, also provide Internet access service to the website operator.

The DOTS server may (not) be co-located with the DOTS mitigator. In typical deployments, the DOTS server belongs to the same administrative domain as the mitigator.

The DOTS client can communicate directly with the DOTS server or indirectly with the DOTS server via a DOTS gateway.

4. Happy Eyeballs for DOTS Signal Channel

DOTS signaling can happen with DTLS over UDP and TLS over TCP. A DOTS client can use DNS to determine the IP address(es) of a DOTS server or a DOTS client may be provided with the list of DOTS server IP addresses. The DOTS client must know a DOTS server's domain name; hard-coding the domain name of the DOTS server into software is NOT RECOMMENDED in case the domain name is not valid or needs to change for legal or other reasons. The DOTS client performs A and/or AAAA record lookup of the domain name and the result will be a list of IP addresses, each of which can be used to contact the DOTS server using UDP and TCP.

If an IPv4 path to reach a DOTS server is found, but the DOTS server's IPv6 path is not working, a dual-stack DOTS client can experience a significant connection delay compared to an IPv4-only DOTS client. The other problem is that if a middlebox between the DOTS client and DOTS server is configured to block UDP, the DOTS client will fail to establish a DTLS session [RFC6347] with the DOTS server and will, then, have to fall back to TLS over TCP [RFC5246] incurring significant connection delays.
[I-D.ietf-dots-requirements] discusses that DOTS client and server will have to support both connectionless and connection-oriented protocols.

To overcome these connection setup problems, the DOTS client can try connecting to the DOTS server using both IPv6 and IPv4, and try both DTLS over UDP and TLS over TCP in a fashion similar to the Happy Eyeballs mechanism [RFC6555]. These connection attempts are performed by the DOTS client when its initializes, and the client uses that information for its subsequent alert to the DOTS server. In order of preference (most preferred first), it is UDP over IPv6, UDP over IPv4, TCP over IPv6, and finally TCP over IPv4, which adheres to address preference order [RFC6724] and the DOTS preference that UDP be used over TCP (to avoid TCP's head of line blocking).

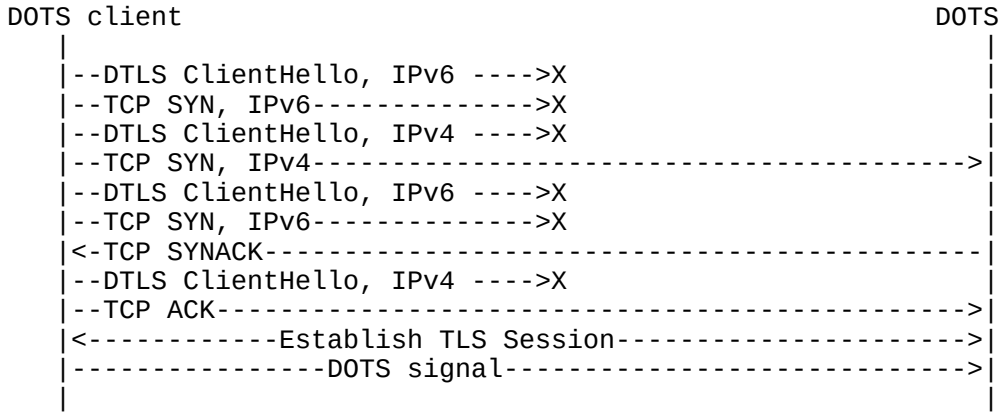


Figure 3: Happy Eyeballs

In reference to Figure 3, the DOTS client sends two TCP SYNs and two DTLS ClientHello messages at the same time over IPv6 and IPv4. In this example, it is assumed that the IPv6 path is broken and UDP is dropped by a middle box but has little impact to the DOTS client because there is no long delay before using IPv4 and TCP. The IPv6 path and UDP over IPv6 and IPv4 is retried until the DOTS client gives up.

5. DOTS Signal Channel

5.1. Overview

Constrained Application Protocol (CoAP) [RFC7252] is used for DOTS signal channel. COAP was designed according to the REST architecture, and thus exhibits functionality similar to that of HTTP, it is quite straightforward to map from CoAP to HTTP and from HTTP to CoAP. CoAP has been defined to make use of both DTLS over UDP and TLS over TCP. The advantages of COAP are: (1) Like HTTP, CoAP is based on the successful REST model, (2) CoAP is designed to use minimal resources, (3) CoAP integrates with JSON, CBOR or any other data format, (4) asynchronous message exchanges, etc.

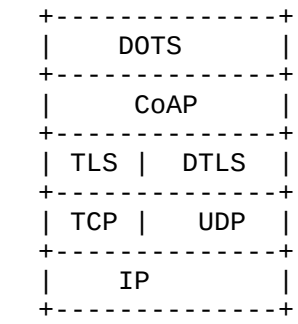


Figure 4: Abstract Layering of DOTS signal channel over CoAP over (D)TLS

JSON [RFC7159] payloads is used to convey signal channel specific payload messages that convey request parameters and response information such as errors.

TBD: Do we want to use CBOR [RFC7049] instead of JSON?

5.2. Mitigation Service Requests

The following APIs define the means to convey a DOTS signal from a DOTS client to a DOTS server:

POST requests: are used to convey the DOTS signal from a DOTS client to a DOTS server over the signal channel, possibly traversing a DOTS gateway, indicating the DOTS client's need for mitigation, as well as the scope of any requested mitigation (Section 5.2.1). DOTS gateway act as a CoAP-to-CoAP Proxy (explained in [RFC7252]).

DELETE requests: are used by the DOTS client to withdraw the request for mitigation from the DOTS server (Section 5.2.2).

GET requests: are used by the DOTS client to retrieve the DOTS signal(s) it had conveyed to the DOTS server (Section 5.2.3).

PUT requests: are used by the DOTS client to convey mitigation efficacy updates to the DOTS server (Section 5.2.4).

Reliability is provided to the POST, DELETE, GET, and PUT requests by marking them as Confirmable (CON) messages. As explained in Section 2.1 of [RFC7252], a Confirmable message is retransmitted using a default timeout and exponential back-off between retransmissions, until the DOTS server sends an Acknowledgement message (ACK) with the same Message ID conveyed from the DOTS client.

TBD: Do we want any of the above requests to be Non-confirmable ?

5.2.1. Convey DOTS Signals

A POST request is used to convey a DOTS signal to the DOTS server (Figure 5).

```
POST {scheme}://{host}:{port}/.well-known/{version}/{URI suffix for DOTS signa
l}
Accept: application/json
Content-Format: application/json
{
   "policy-id": "number",
   "target-ip": "string",
   "target-port": "string",
   "target-protocol": "string",
   "lifetime": "number"
}
```

Figure 5: POST to convey DOTS signals

The header fields are described below.

policy-id: Identifier of the policy represented using a number. This identifier MUST be unique for each policy bound to the DOTS client, i.e. ,the policy-id needs to be unique relative to the active policies with the DOTS server. This identifier must be generated by the DOTS client. This document does not make any assumption about how this identifier is generated. This is a mandatory attribute.

target-ip: A list of IP addresses or prefixes under attack. This is an optional attribute.

target-port: A list of ports under attack. This is an optional attribute.

target-protocol: A list of protocols under attack. Valid protocol values include tcp, udp, sctp, and dccp. This is an optional attribute.

lifetime: Lifetime of the mitigation request policy in seconds. Upon the expiry of this lifetime, and if the request is not refreshed, the mitigation request is removed. The request can be refreshed by sending the same request again. The default lifetime of the policy is 60 minutes -- this value was chosen to be long enough so that refreshing is not typically a burden on the DOTS client, while expiring the policy where the client has unexpectedly quit in a timely manner. A lifetime of zero

indicates indefinite lifetime for the mitigation request. The server MUST always indicate the actual lifetime in the response. This is an optional attribute in the request.

The relative order of two rules is determined by comparing their respective policy identifiers. The rule with lower numeric policy identifier value has higher precedence (and thus will match before) than the rule with higher numeric policy identifier value.

If the DOTS server is not able to respond immediately to a POST request carried in a Confirmable message, it simply responds with an Empty Acknowledgement message so that the DOTS client can stop retransmitting the request. When the response is ready, the server sends it in a new Confirmable message which then in turn needs to be acknowledged by the DOTS client (see Sections 5.2.1 and Sections 5.2.2 in [RFC7252]).

To avoid DOTS signal message fragmentation and the consequently decreased probability of message delivery, DOTS agents MUST ensure that the DTLS record MUST fit within a single datagram. If the Path MTU is not known to the DOTS server, an IP MTU of 1280 bytes SHOULD be assumed. The length of the URL MUST NOT exceed 256 bytes. If UDP is used to convey the DOTS signal and the request size exceeds the Path MTU then the DOTS client MUST split the DOTS signal into separate messages, for example the list of addresses in the 'target-ip' field could be split into multiple lists and each list conveyed in a new POST request.

Implementation Note: DOTS choice of message size parameters works well with IPv6 and with most of today's IPv4 paths. However, with IPv4, it is harder to absolutely ensure that there is no IP fragmentation. If IPv4 support on unusual networks is a consideration and path MTU is unknown, implementations may want to limit themselves to more conservative IPv4 datagram sizes such as 576 bytes, as per [RFC0791] IP packets up to 576 bytes should never need to be fragmented, thus sending a maximum of 500 bytes of DOTS signal over a UDP datagram will generally avoid IP fragmentation.

Figure 6 shows a POST request to signal that ports 80, 8080, and 443 on the servers 2002:db8:6401::1 and 2002:db8:6401::2 are being attacked.

```
POST coaps://www.example.com/.well-known/v1/DOTS signal
Accept: application/json
Content-Format: application/json
{
  "policy-id":123321333242,
  "target-ip":[
      "2002:db8:6401::1",
      "2002:db8:6401::2"
  ],
  "target-port":[
      "80",
      "8080",
      "443"
  ],
  "target-protocol":"tcp"
}
```

Figure 6: POST for DOTS signal

The DOTS server indicates the result of processing the POST request using CoAP response codes. CoAP 2xx codes are success, CoAP 4xx codes are some sort of invalid request and 5xx codes are returned if the DOTS server has erred or is incapable of performing the mitigation. Response code 2.01 (Created) will be returned in the response if the DOTS server has accepted the mitigation request and will try to mitigate the attack. If the request is missing one or more mandatory attributes then 4.00 (Bad Request) will be returned in the response or if the request contains invalid or unknown parameters then 4.02 (Invalid query) will be returned in the response. The CoAP response will include the JSON body received in the request.

5.2.2. Withdraw a DOTS Signal

A DELETE request is used to withdraw a DOTS signal from a DOTS server (Figure 7).

```
DELETE {scheme}://{host}:{port}/.well-known/{URI suffix for DOTS signal}
Accept: application/json
Content-Format: application/json
{
  "policy-id": "number"
}
```

Figure 7: Withdraw DOTS signal

If the DOTS server does not find the policy number conveyed in the DELETE request in its policy state data, then it responds with a 4.04 (Not Found) error response code. The DOTS server successfully

acknowledges a DOTS client's request to withdraw the DOTS signal using 2.02 (Deleted) response code, and ceases mitigation activity as quickly as possible.

5.2.3. Retrieving a DOTS Signal

A GET request is used to retrieve information and status of a DOTS signal from a DOTS server (Figure 8). If the DOTS server does not find the policy number conveyed in the GET request in its policy state data, then it responds with a 4.04 (Not Found) error response code.

1) To retrieve all DOTS signals signaled by the DOTS client.

```
GET {scheme}://{host}:{port}/.well-known/{URI suffix for DOTS signal}/list
Observe : 0
```

2) To retrieve a specific DOTS signal signaled by the DOTS client. The policy information in the response will be formatted in the same order it was processed at the DOTS server.

```
GET {scheme}://{host}:{port}/.well-known/{URI suffix for DOTS signal}/<policy-id
 number>
Observe : 0
```

Figure 8: GET to retrieve the rules

Figure 9 shows the response of all the active policies on the DOTS server.

```
{
  "policy-data":[
    {
      "policy-id":123321333242,
      "target-prtoocol":"tcp",
      "lifetime":3600,
      "status":"mitigation in progress"
    },
    {
      "policy-id":123321333244,
      "target-protocol":"udp",
      "lifetime":1800,
      "status":"mitigation complete"
    },
    {
      "policy-id":123321333245,
      "target-protocol":"tcp",
      "lifetime":1800,
      "status":"attack stopped"
    }
  ]
}
```

Figure 9: Response body

The various possible values of status field are explained below:

mitigation in progress: Attack mitigation is in progress (e.g., changing the network path to re-route the inbound traffic to DOTS mitigator).

mitigation complete: Attack is successfully mitigated (e.g., attack traffic is dropped).

attack stopped: Attack has stopped and the DOTS client can withdraw the mitigation request.

The observe option defined in [RFC7641] extends the CoAP core protocol with a mechanism for a CoAP client to "observe" a resource on a CoAP server: the client retrieves a representation of the resource and requests this representation be updated by the server as long as the client is interested in the resource. A DOTS client conveys the observe option set to 0 in the GET request to receive unsolicited notifications of attack mitigation status from the DOTS server. Unidirectional notifications within the bidirectional signal channel allows unsolicited message delivery, enabling asynchronous notifications between the agents.

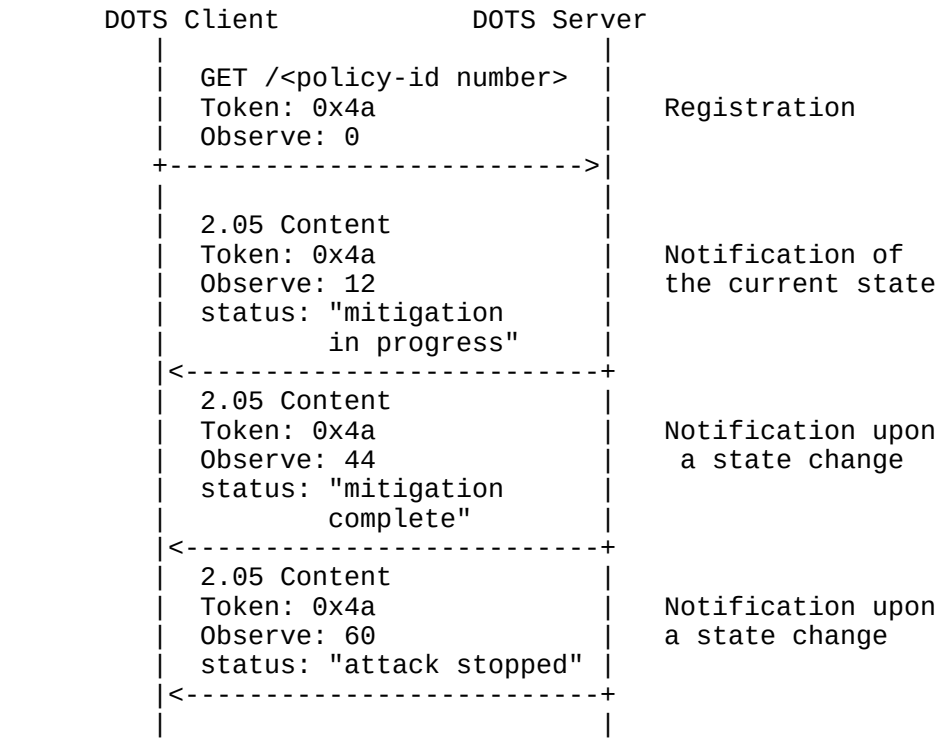


Figure 10: Notifications of attack mitigation status

5.2.3.1. Mitigation Status

A DOTS client retrieves the information about a DOTS signal at frequent intervals to determine the status of an attack. If the DOTS server has been able to mitigate the attack and the attack has stopped, the DOTS server indicates as such in the status, and the DOTS client recalls the mitigation request.

A DOTS client should react to the status of the attack from the DOTS server and not the fact that it has recognized, using its own means, that the attack has been mitigated. This ensures that the DOTS client does not recall a mitigation request in a premature fashion because it is possible that the DOTS client does not sense the DDOS attack on its resources but the DOTS server could be actively mitigating the attack and the attack is not completely averted.

5.2.4. Efficacy Update from DOTS Client

While DDoS mitigation is active, a DOTS client MAY frequently transmit DOTS mitigation efficacy updates to the relevant DOTS server. An PUT request (Figure 11) is used to convey the mitigation efficacy update to the DOTS server. The PUT request MUST include all the header fields used in the POST request to convey the DOTS signal (Section 5.2.1). If the DOTS server does not find the policy number conveyed in the PUT request in its policy state data, it responds with a 4.04 (Not Found) error response code.

```
PUT {scheme}://{host}:{port}/.well-known/{URI suffix for DOTS signal}/<policy-
id number>
Accept: application/json
Content-Format: application/json
{
   "target-ip": "string",
   "target-port": "string",
   "target-protocol": "string",
   "lifetime": "number",
   "attack-status": "string"
}
```

Figure 11: Efficacy Update

The 'attack-status' field is a mandatory attribute. The various possible values contained in the 'attack-status' field are explained below:

in-progress: DOTS client determines that it is still under attack.

terminated: Attack is successfully mitigated (e.g., attack traffic is dropped).

6. DOTS Data Channel

The data channel is intended to be used for bulk data exchanges and requires a reliable transport, CoAP over TLS over TCP is used for data channel.

JSON payloads is used to convey both filtering rules as well as data channel specific payload messages that convey request parameters and response information such as errors. All data channel URIs defined in this document, and in subsequent documents, MUST NOT have a URI containing "/DOTS signal".

One of the possible arrangements for DOTS client to signal filtering rules to a DOTS server via the DOTS gateway is discussed below:

The DOTS conveys the black-list rules to the DOTS gateway. The DOTS gateway validates if the DOTS client is authorized to signal the black-list rules and if the client is authorized propagates the rules to the DOTS server. Likewise, the DOTS server validates if the DOTS gateway is authorized to signal the black-list rules. To create or purge filters, the DOTS client sends CoAP requests to the DOTS gateway. The DOTS gateway acts as a proxy, validates the rules and proxies the requests containing the black-listed IP addresses to a DOTS server. When the DOTS gateway receives the associated CoAP response from the DOTS server, it propagates the response back to the DOTS client. If an attack is detected by the DOTS gateway then it can act as a DOTS client and signal the black-list rules to the DOTS server. The DOTS gateway plays the role of both client and server.

6.1. Filtering Rules

The following APIs define means for a DOTS client to configure filtering rules on a DOTS server.

6.1.1. Install Filtering Rules

An POST request is used to push filtering rules to a DOTS server (Figure 12).

```
POST {scheme}://{host}:{port}/.well-known/{version}/{URI suffix for filtering}
Accept: application/json
Content-Format: application/json
{
   "policy-id": "number",
   "traffic-protocol": "string",
   "source-protocol-port": "string",
   "destination-protocol-port": "string",
   "destination-ip": "string",
   "source-ip": "string",
   "lifetime": "number",
   "traffic-rate" : "number"
 }
```

Figure 12: POST to install filtering rules

The header fields are described below:

policy-id: Identifier of the policy represented using a number. This identifier MUST be unique for each policy bound to the DOTS client, i.e., the policy-id needs to be unique relative to the active policies with the DOTS server. This identifier must be generated by the client. This document does not make any

assumption about how this identifier is generated. This is an mandatory attribute.

traffic-protocol: Valid protocol values include tcp, udp, sctp, and dccp. This is an mandatory attribute.

source-protocol-port: The source port number, port number range (using "-"). For TCP, UDP, SCTP, or DCCP: the source range of ports (e.g., 1024-65535). This is an optional attribute.

destination-protocol-port: The destination port number, port number range (using "-"). For TCP, UDP, SCTP, or DCCP: the destination range of ports (e.g., 443-443). This information is useful to avoid disturbing a group of customers when address sharing is in use [RFC6269]. This is an optional attribute.

destination-ip: The destination IP address, IP addresses separated by commas, or prefixes using "/" notation. This is an optional attribute.

source-ip: The source IP addresses, IP addresses separated by commas, or prefixes using "/" notation. This is an optional attribute.

lifetime: Lifetime of the rule in seconds. Upon the expiry of this lifetime, and if the request is not refreshed, this particular rule is removed. The rule can be refreshed by sending the same message again. The default lifetime of the rule is 60 minutes -- this value was chosen to be long enough so that refreshing is not typically a burden on the DOTS client, while expiring the rule where the client has unexpectedly quit in a timely manner. A lifetime of zero indicates indefinite lifetime for the rule. The server MUST always indicate the actual lifetime in the response. This is an optional attribute in the request.

traffic-rate: This is the allowed traffic rate in bytes per second indicated in IEEE floating point [IEEE.754.1985] format. The value 0 indicates all traffic for the particular flow to be discarded. This is a mandatory attribute.

The relative order of two rules is determined by comparing their respective policy identifiers. The rule with lower numeric policy identifier value has higher precedence (and thus will match before) than the rule with higher numeric policy identifier value.

Figure 13 shows a POST request to block traffic from attacker IPv6 prefix 2001:db8:abcd:3f01::/64 to network resource using IPv6 address 2002:db8:6401::1 to operate a server on TCP port 443.

```
POST coaps://www.example.com/.well-known/v1/filter
Accept: application/json
Content-Format: application/json
 {
   "policy-id": 123321333242,
   "traffic-protocol": "tcp",
   "source-protocol-port": "0-65535",
   "destination-protocol-port": "443",
   "destination-ip": "2001:db8:abcd:3f01::/64",
   "source-ip": "2002:db8:6401::1",
   "lifetime": 1800,
   "traffic-rate": 0
 }
```

Figure 13: POST to Install Black-list Rules

6.1.2. Remove Filtering Rules

A DELETE request is used to delete filtering rules from a DOTS server (Figure 14).

```
DELETE {scheme}://{host}:{port}/.well-known/{URI suffix for filtering}
Accept: application/json
Content-Format: application/json
 {
   "policy-id": "number"
 }
```

Figure 14: DELETE to remove the rules

6.1.3. Retrieving Installed Filtering Rules

A GET request is used to retrieve filtering rules from a DOTS server.

Figure 15 shows an example to retrieve all the black-lists rules programmed by the DOTS client while Figure 16 shows an example to retrieve specific black-list rules programmed by the DOTS client.

```
GET {scheme}://{host}:{port}/.well-known/{URI suffix for filtering}
```

Figure 15: GET to retrieve the rules (1)

```
GET {scheme}://{host}:{port}/.well-known/{URI suffix for filtering}
Accept: application/json
Content-Format: application/json
{
   "policy-id": "number"
}
```

Figure 16: GET to retrieve the rules (2)

TODO: show response

7. (D)TLS Protocol Profile and Performance considerations

This section defines the (D)TLS protocol profile of DOTS signal channel over (D)TLS and DOTS data channel over TLS.

There are known attacks on (D)TLS, such as machine-in-the-middle and protocol downgrade. These are general attacks on (D)TLS and not specific to DOTS over (D)TLS; please refer to the (D)TLS RFCs for discussion of these security issues. DOTS agents MUST adhere to the (D)TLS implementation recommendations and security considerations of [RFC7525] except with respect to (D)TLS version. Since encryption of DOTS using (D)TLS is virtually a green-field deployment DOTS agents MUST implement only (D)TLS 1.2 or later.

Implementations compliant with this profile MUST implement all of the following items:

- DOTS client can use (D)TLS session resumption without server-side state [RFC5077] to resume session and convey the DOTS signal.

- While the communication to the DOTS server is quiescent, the DOTS client may want to probe the server to ensure it has maintained cryptographic state. Such probes can also keep alive firewall or NAT bindings. This probing reduces the frequency of needing a new handshake when a DOTS signal needs to be conveyed to the DOTS server.

  * A (D)TLS heartbeat [RFC6520] verifies the DOTS server still has DTLS state by returning a DTLS message. If the server has lost state, it returns a DTLS Alert. Upon receipt of an unauthenticated DTLS Alert, the DTLS client validates the Alert is within the replay window (Section 4.1.2.6 of [RFC6347]). It is difficult for the DTLS client to validate the DTLS Alert was generated by the DTLS server in response to a request or was generated by an on- or off-path attacker. Thus, upon receipt of an in-window DTLS Alert, the client SHOULD continue re-transmitting the DTLS packet (in the event the Alert was

spoofed), and at the same time it SHOULD initiate DTLS session resumption.

* TLS runs over TCP, so a simple probe is a 0-length TCP packet (a "window probe"). This verifies the TCP connection is still working, which is also sufficient to prove the server has retained TLS state, because if the server loses TLS state it abandons the TCP connection. If the server has lost state, a TCP RST is returned immediately.

* Raw public keys [RFC7250] which reduce the size of the ServerHello, and can be used by servers that cannot obtain certificates (e.g., DOTS gateways on private networks).

Implementations compliant with this profile SHOULD implement all of the following items to reduce the delay required to deliver a DOTS signal:

- TLS False Start [I-D.ietf-tls-falsestart] which reduces round-trips by allowing the TLS second flight of messages (ChangeCipherSpec) to also contain the DOTS signal.

- Cached Information Extension [I-D.ietf-tls-cached-info] which avoids transmitting the server’s certificate and certificate chain if the client has cached that information from a previous TLS handshake.

- TCP Fast Open [RFC7413] can reduce the number of round-trips to convey DOTS signal.

8. Mutual Authentication of DOTS Agents & Authorization of DOTS Clients

(D)TLS based on client certificate can be used for mutual authentication between DOTS agents. If a DOTS gateway is involved, DOTS clients and DOTS gateway MUST perform mutual authentication; only authorized DOTS clients are allowed to send DOTS signals to a DOTS server.

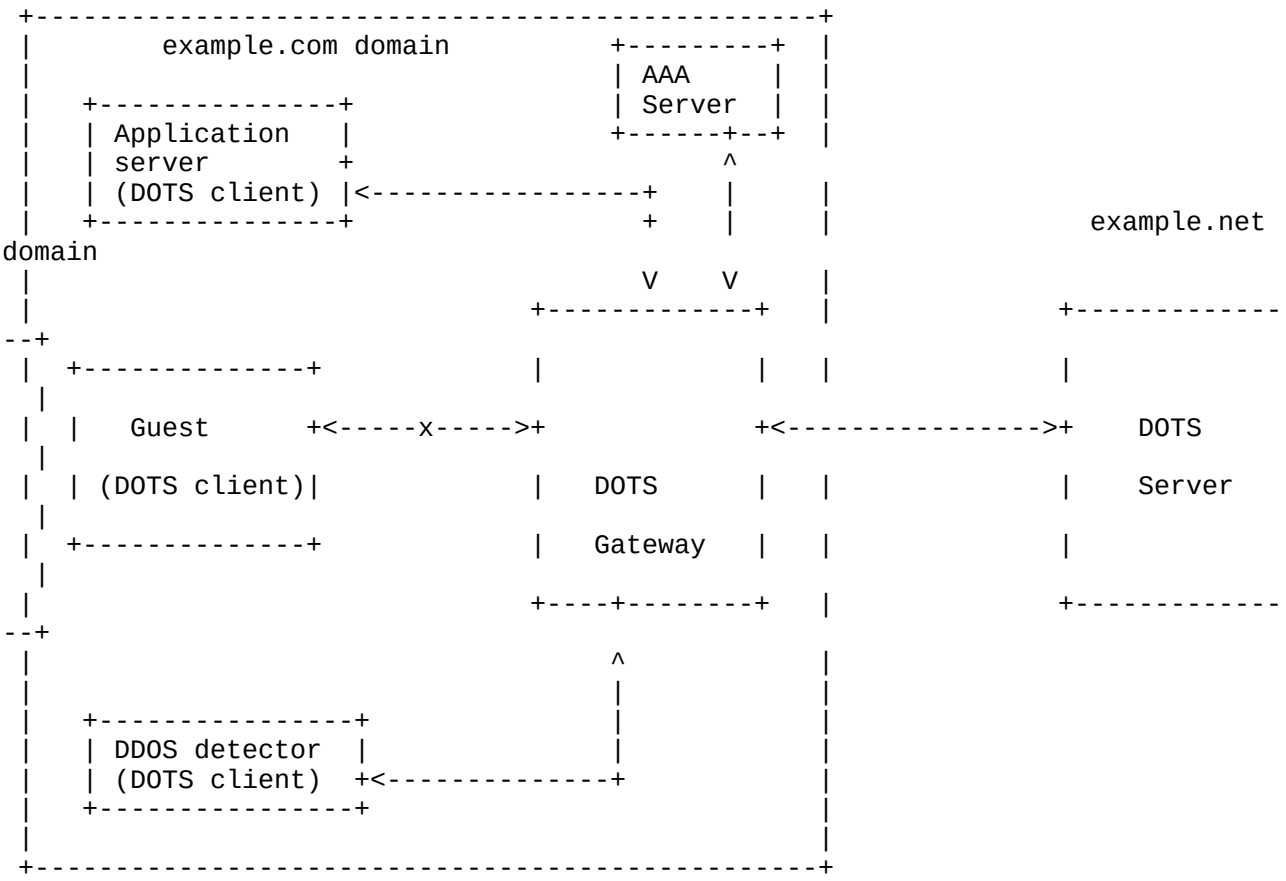


Figure 17: Example of Authentication and Authorization of DOTS Agents

In the example depicted in Figure 17, the DOTS gateway and DOTS clients within the 'example.com' domain mutually authenticate with each other. After the DOTS gateway validates the identity of a DOTS client, it communicates with the AAA server in the 'example.com' domain to determine if the DOTS client is authorized to request DDOS mitigation. If the DOTS client is not authorized, a 4.01 (Unauthorized) is returned in the response to the DOTS client. In this example, the DOTS gateway only allows the application server and DDOS detector to request DDOS mitigation, but does not permit the user of type 'guest' to request DDOS mitigation.

Also, DOTS gateway and DOTS server MUST perform mutual authentication using certificates. A DOTS server will only allow a DOTS gateway with a certificate for a particular domain to request mitigation for that domain. In reference to Figure 17, the DOTS server only allows the DOTS gateway to request mitigation for 'example.com' domain and not for other domains.

9. IANA Considerations

TODO

10. Security Considerations

Authenticated encryption MUST be used for data confidentiality and message integrity. (D)TLS based on client certificate MUST be used for mutual authentication. The interaction between the DOTS agents requires Datagram Transport Layer Security (DTLS) and Transport Layer Security (TLS) with a ciphersuite offering confidentiality protection and the guidance given in [RFC7525] MUST be followed to avoid attacks on (D)TLS.

If TCP is used between DOTS agents, attacker may be able to inject RST packets, bogus application segments, etc., regardless of whether TLS authentication is used. Because the application data is TLS protected, this will not result in the application receiving bogus data, but it will constitute a DoS on the connection. This attack can be countered by using TCP-AO [RFC5925]. If TCP-AO is used, then any bogus packets injected by an attacker will be rejected by the TCP-AO integrity check and therefore will never reach the TLS layer.

Special care should be taken in order to ensure that the activation of the proposed mechanism won't have an impact on the stability of the network (including connectivity and services delivered over that network).

Involved functional elements in the cooperation system must establish exchange instructions and notification over a secure and authenticated channel. Adequate filters can be enforced to avoid that nodes outside a trusted domain can inject request such as deleting filtering rules. Nevertheless, attacks can be initiated from within the trusted domain if an entity has been corrupted. Adequate means to monitor trusted nodes should also be enabled.

11. Acknowledgements

Thanks to Christian Jacquenet, Roland Dobbins, Andrew Mortensen, Roman D. Danyliw, and Gilbert Clark for the discussion and comments.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <http://www.rfc-editor.org/info/rfc5925>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <http://www.rfc-editor.org/info/rfc7641>.

12.2. Informative References

[I-D.ietf-dots-requirements] Mortensen, A., Moskowitz, R., and T. Reddy, "Distributed Denial of Service (DDoS) Open Threat Signaling Requirements", draft-ietf-dots-requirements-01 (work in progress), March 2016.

[I-D.ietf-tls-cached-info]
Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", draft-ietf-tls-cached-info-23 (work in progress), May 2016.

[I-D.ietf-tls-falsestart]
Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", draft-ietf-tls-falsestart-02 (work in progress), May 2016.

[IEEE.754.1985]
Institute of Electrical and Electronics Engineers, "Standard for Binary Floating-Point Arithmetic", August 1985.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[RFC4732] Handley, M., Ed., Rescorla, E., Ed., and IAB, "Internet Denial-of-Service Considerations", RFC 4732, DOI 10.17487/RFC4732, December 2006, <http://www.rfc-editor.org/info/rfc4732>.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <http://www.rfc-editor.org/info/rfc4987>.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, DOI 10.17487/RFC5077, January 2008, <http://www.rfc-editor.org/info/rfc5077>.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <http://www.rfc-editor.org/info/rfc5575>.

[RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <http://www.rfc-editor.org/info/rfc6269>.

[RFC6520] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", RFC 6520, DOI 10.17487/RFC6520, February 2012, <http://www.rfc-editor.org/info/rfc6520>.

[RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, DOI 10.17487/RFC6555, April 2012, <http://www.rfc-editor.org/info/rfc6555>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <http://www.rfc-editor.org/info/rfc7413>.

Appendix A. BGP

BGP defines a mechanism as described in [RFC5575] that can be used to automate inter-domain coordination of traffic filtering, such as what is required in order to mitigate DDoS attacks. However, support for BGP in an access network does not guarantee that traffic filtering will always be honored. Since a DOTS client will not receive an acknowledgment for the filtering request, the DOTS client should monitor and apply similar rules in its own network in cases where the DOTS server is unable to enforce the filtering rules. In addition, enforcement of filtering rules of BGP on Internet routers are usually governed by the maximum number of data elements the routers can hold as well as the number of events they are able to process in a given unit of time.

Authors' Addresses

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tireddy@cisco.com

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

Prashanth Patil
Cisco Systems, Inc.

Email: praspati@cisco.com

Mike Geller
Cisco Systems, Inc.
3250
Florida 33309
USA

Email: mgeller@cisco.com

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Robert Moskowitz
HTT Consulting
Oak Park, MI 42837
United States

Email: rgm@htt-consult.com