IPFIX Information Elements for inspecting network security issues
draft-fu-ipfix-network-security-01

Abstract

IPFIX protocol has been used to carry Information Elements, which are defined to measure the traffic information and information related to the traffic observation point, traffic metering process and the exporting process. Network or device status are checked through analysing necessary observed information. Although most of the existing Information Elements are useful for network security inspection, they are still not sufficient to determine the reasons behind observed events such as for DDOS attack, ICMP attack, and fragment attack. To allow administrators making effective and quick response to the attacks, this document extends the standard Information Elements and describes the formats for inspecting network security.

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 October 30, 2015.
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.

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

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

The IPFIX requirement [RFC3917] points out that one of the target applications of IPFIX is attack and intrusion detection. The IPFIX Protocol [RFC7011] defines a generic exchange mechanism for flow information and events. It supports source-triggered exporting of information due to the push model approach other than exporting upon flow-end or fixed time intervals. The IPFIX Information Model [RFC5102] defines a list of standard Information Elements (IEs) which can be carried by the IPFIX protocol. Even though the existing standard IEs are useful to check the status/events of the traffic, they are not sufficient to help network administrators identify categories of the attacks. The scanty information will result in an inaccurate analysis and slowing down the effective response towards network attacks.

For instance, CC (Challenge Collapsar) attack is a typical application layer DDoS attack, which mainly attacks the dynamic pages of web server. It makes the web server’s resources exhausted and paralyzed, so the server will be denial of service. Because CC attacker imitates normal users’ behavior pretty well by using different real IP addresses with relatively competitive access process (even with low speed), it makes the attack concealed well compared with traditional network layer DDoS (e.g. SYN-Flood, etc). In addition, the attacker often manipulates the attack behind the scenes by non-direct communicate with target server, so the attack is not easy to be tracked and discovered. It would be useful to collect application status information for application layer attacks. In this case, CC attack is likely to happen if a large number of non 2XX HTTP status code replied from the server are observed.

Fragment attack employs unexpected formats of fragmentation, which will result in errors such as fragmentation buffer full, fragment overlapped, fragment incomplete. Existing IPFIX fragmentation metrics includes fragmentOffset, fragmentIdentification, fragmentFlags, which only indicate the attributes of a single fragment, and are not suitable for attack detection. Integrated measurements are needed to provide an holistic review of the session. Furthermore 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). The current ICMP information elements of IPFIX contains the ICMP type and code for both IPv4 and IPv6, however they are 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 [RFC 5477]. Similar problems occur in TCP, UDP, SNMP and DNS attack, it would be useful to derive the number of the upstream and downstream packets separately and over time in order to detect the anomalies of the network.

Upon the above discussions and per IPFIX applicability [RFC 5472], derived metrics are useful to provide sufficient evidence about security incident. A wisely chosen sets of derived metrics will allow direct exporting with minimal resource consumption. This document extends the IPFIX Information model and defines Information Elements (IEs) that SHOULD be used to identify different attack categories, the standardization of those IEs will improve the network security and will support the offline analysis of data from different operators in the future.

3. Information Elements and use cases

This section presents the information elements that are useful for attack detection, the IPFIX templates could contain a subset of the Information Elements (IEs) shown in Table 1 depending upon the attack under concern of the network administrator. For example a session creation template contains

\{(sourceIPv4Address, destinationIPv4Address, sourceTransportPort, destinationTransportPort, protocolIdentifier, pktUpstreamCount, pktDownstreamCount, selectorAlgorithm, samplingPacketInterval, samplingPacketSpace)\}

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

\{(192.168.0.201, 192.168.0.1, 51132, 80, 7, 67, 87, 3, 100, 1000)\}

3.1. Information Elements

The following is the table of all the IEs that a device would need to export for attack statistic analysis. The formats of the IEs and the IPFIX IDs are listed below. Most of the IEs are defined in [IPFIX-IANA], while some of the IPFIX IE's ID are not assigned yet, and hence the detailed explanation of these fields are presented in the following sections. The recommended registrations to IANA is described the IANA considerations section.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Size (bits)</th>
<th>IANA IPFIX ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceIPv4Address</td>
<td>32</td>
<td>8</td>
<td>Source IPv4 Address</td>
</tr>
<tr>
<td>destinationIPv4Address</td>
<td>32</td>
<td>12</td>
<td>Destination IPv4 Address</td>
</tr>
<tr>
<td>sourceTransportPort</td>
<td>16</td>
<td>7</td>
<td>Source Port</td>
</tr>
<tr>
<td>destinationTransportPort</td>
<td>16</td>
<td>11</td>
<td>Destination Port</td>
</tr>
<tr>
<td>protocolIdentifier</td>
<td>8</td>
<td>4</td>
<td>Transport protocol</td>
</tr>
<tr>
<td>packetDeltaCount</td>
<td>64</td>
<td>2</td>
<td>The number of incoming packets since the previous report (if any) for this</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flow at the Observation Point</td>
</tr>
<tr>
<td>pktUpstreamCount</td>
<td>64</td>
<td>TBD</td>
<td>Upstream packet counter</td>
</tr>
<tr>
<td>pktDownstreamCount</td>
<td>64</td>
<td>TBD</td>
<td>Downstream packet counter</td>
</tr>
<tr>
<td>octetUpstreamCount</td>
<td>64</td>
<td>TBD</td>
<td>Upstream octet counter</td>
</tr>
<tr>
<td>octetDownstreamCount</td>
<td>64</td>
<td>TBD</td>
<td>Downstream octet counter</td>
</tr>
<tr>
<td>tcpSynTotalCount</td>
<td>64</td>
<td>218</td>
<td>The total number of packets of this Flow with TCP &quot;Synchronize sequence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>numbers&quot; (SYN) flag set</td>
</tr>
<tr>
<td>tcpFinTotalCount</td>
<td>64</td>
<td>219</td>
<td>The total number of packets of this Flow with TCP &quot;No more data from</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sender&quot; (FIN) flag set</td>
</tr>
<tr>
<td>IE Name</td>
<td>Size</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>tcpRstTotalCount</td>
<td>64</td>
<td>220</td>
<td>The total number of packets of this Flow with TCP &quot;Reset the connection&quot; (RST) flag set.</td>
</tr>
<tr>
<td>tcpPshTotalCount</td>
<td>64</td>
<td>221</td>
<td>The total number of packets of this Flow with TCP &quot;Push Function&quot; (PSH) flag set.</td>
</tr>
<tr>
<td>tcpAckTotalCount</td>
<td>64</td>
<td>222</td>
<td>The total number of packets of this Flow with TCP &quot;Acknowledgment field significant&quot; (ACK) flag set.</td>
</tr>
<tr>
<td>tcpUrgTotalCount</td>
<td>64</td>
<td>223</td>
<td>The total number of packets of this Flow with TCP &quot;Urgent Pointer field significant&quot; (URG) flag set.</td>
</tr>
<tr>
<td>tcpControlBits</td>
<td>8</td>
<td>6</td>
<td>TCP control bits observed for packets of this Flow</td>
</tr>
<tr>
<td>flowEndReason</td>
<td>8</td>
<td>136</td>
<td>The reason for Flow termination</td>
</tr>
<tr>
<td>minimumIpTotalLength</td>
<td>64</td>
<td>25</td>
<td>Length of the smallest packet observed for this Flow</td>
</tr>
<tr>
<td>maximumIpTotalLength</td>
<td>64</td>
<td>26</td>
<td>Length of the largest packet</td>
</tr>
<tr>
<td>IE Name</td>
<td>Type</td>
<td>Length</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>flowStartSeconds</td>
<td>dateTimeSeconds</td>
<td>150</td>
<td>The absolute timestamp of the first packet of this Flow</td>
</tr>
<tr>
<td>flowEndSeconds</td>
<td>dateTimeSeconds</td>
<td>151</td>
<td>The absolute timestamp of the last packet of this Flow</td>
</tr>
<tr>
<td>flowStartMilliseconds</td>
<td>dateTimeMilliseconds</td>
<td>152</td>
<td>The absolute timestamp of the first packet of this Flow</td>
</tr>
<tr>
<td>flowEndMilliseconds</td>
<td>dateTimeMilliseconds</td>
<td>153</td>
<td>The absolute timestamp of the last packet of this Flow</td>
</tr>
<tr>
<td>flowStartMicroseconds</td>
<td>dateTimeMicroseconds</td>
<td>154</td>
<td>The absolute timestamp of the first packet of this Flow</td>
</tr>
<tr>
<td>flowEndMicroseconds</td>
<td>dateTimeMicroseconds</td>
<td>155</td>
<td>The absolute timestamp of the last packet of this Flow</td>
</tr>
<tr>
<td>applicationErrorCodeCount</td>
<td>32</td>
<td>TBD</td>
<td>Number of packets with application error code detected</td>
</tr>
<tr>
<td>fragmentFlags</td>
<td>8</td>
<td>197</td>
<td>Fragmentation properties indicated by flags in the IPv4 packet header or the IPv6 Fragment header, respectively</td>
</tr>
<tr>
<td>fragmentIncompleteCount</td>
<td>32</td>
<td>TBD</td>
<td>Counter of incomplete</td>
</tr>
<tr>
<td>IE Name</td>
<td>Size</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fragmentFirstTooShortCount</td>
<td>32</td>
<td>TBD</td>
<td>Number of packets with first fragment too short</td>
</tr>
<tr>
<td>fragmentOffsetErrorCount</td>
<td>32</td>
<td>TBD</td>
<td>Number of fragments with offset error</td>
</tr>
<tr>
<td>fragmentFlagErrorCount</td>
<td>32</td>
<td>TBD</td>
<td>Number of fragments with flag error</td>
</tr>
<tr>
<td>icmpTypeIPv4</td>
<td>8</td>
<td>176</td>
<td>Type of the IPv4 ICMP message</td>
</tr>
<tr>
<td>icmpCodeIPv4</td>
<td>8</td>
<td>177</td>
<td>Code of the IPv4 ICMP message</td>
</tr>
<tr>
<td>icmpTypeIPv6</td>
<td>8</td>
<td>178</td>
<td>Type of the IPv6 ICMP message</td>
</tr>
<tr>
<td>icmpCodeIPv6</td>
<td>8</td>
<td>179</td>
<td>Code of the IPv6 ICMP message</td>
</tr>
<tr>
<td>icmpEchoCount</td>
<td>32</td>
<td>TBD</td>
<td>The number of ICMP echo.</td>
</tr>
<tr>
<td>icmpEchoReplyCount</td>
<td>32</td>
<td>TBD</td>
<td>The number of ICMP echo reply.</td>
</tr>
<tr>
<td>selectorAlgorithm</td>
<td>16</td>
<td>304</td>
<td>This Information Element identifies the packet selection methods (e.g.,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Filtering, Sampling) that are applied by the Selection Process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The number of packets that are consecutively sampled.</td>
</tr>
<tr>
<td>samplingPacketInterval</td>
<td>32</td>
<td>305</td>
<td>The number of packets between two &quot;s</td>
</tr>
<tr>
<td>samplingPacketSpace</td>
<td>32</td>
<td>306</td>
<td></td>
</tr>
</tbody>
</table>
3.2. Packet upstream/downstream counters

A sudden increase of Flow from different sources to one destination may be caused by an attack on a specific host or network node using spoofed addresses. However it may be caused by legitimate users who seek access to a recently published web content. Only reporting the total packet number is not sufficient to indicate whether attacks occur, as it lacks details to separate good packets from abnormal packets. As a result, upstream and downstream packets should be monitored separately so that upstream to downstream packet number ratio can be used to detect successful connections. \texttt{pktUpstreamCount} and \texttt{pktDownstreamCount} are added to IPFIX to represent the cumulated upstream and downstream packet number respectively.

3.3. ICMP echo/echo reply counters

An unusual ratio of ICMP echo to ICMP echo reply packets can refer to ICMP attack. However the existing set of IPFIX IEs provides the type and code of ICMP packet, continuously export the information will result in serious resource consumption at the exporter, the collector and the bandwidth. The number of echo and echo reply packets in a Flow can be derived for the Observation Domain in a specific time interval or once the ratio exceeds threshold. The basic metrics \texttt{icmpEchoCount} and \texttt{icmpEchoReplyCount} are defined as new IPFIX Information Elements.

3.4. Fragment statistic

Typical fragment attack includes fragmentation buffer full, fragment overlapped, fragment incomplete. Existing IPFIX fragmentation metrics includes fragmentIdentification, fragmentOffset, fragmentFlags, which are not sufficient to identify errors, and are not suitable for early attack detection. Integrated measurements are needed to provide an holistic review of the flow. \texttt{fragmentIncompleteCount} checks the number of incomplete fragmentation, \texttt{fragmentFirstTooShortCount} verifies the number of fragments with first fragment too short, \texttt{fragmentOffsetErrorCount} checks the number of fragments with offset error, and \texttt{fragmentFlagErrorCount} detect early whether the fragmentation is caused by a malicious attack.
3.5. Application error code

The application layer attack requires IPFIX protocol capture packet payload. An initial consideration of the application error code comes from the HTTP status code except 2XX successful code. Other application layer protocol error code are also supported. The error code list can be expanded in the future as necessary. The data record will have the corresponding error code value to identify the error that is being logged.

3.6. Extended value of FlowEndReason

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.

4. Encoding

4.1. IPFIX

This document uses IPFIX as the encoding mechanism to monitor security events. However, the information that is logged SHOULD be the same irrespective of what kind of encoding scheme is used. IPFIX is chosen, because it is an IETF standard that meets all the needs for a reliable logging mechanism and one of its targets are for security applications. IPFIX provides the flexibility to the logging device to define the data sets that it is logging. The IEs specified for logging MUST be the same irrespective of the encoding mechanism used.
5. IANA Considerations

The following information elements are requested from IANA IPFIX registry.

Name: pktUpstreamCount

Description: The number of the upstream packets for this Flow at the Observation Point since the Metering Process (re-)initialization for this Observation Point.

Abstract Data Type: unsigned64

Data Type Semantics: TBD

Name: pktDownstreamCount

Description: The number of the downstream packets for this Flow at the Observation Point since the Metering Process (re-)initialization for this Observation Point.

Abstract Data Type: unsigned64

Data Type Semantics: TBD

Name: octetUpstreamCount

Description: The total number of octets in upstream packets for this Flow at the Observation Point since the Metering Process (re-)initialization for this Observation Point. The number of octets includes IP header(s) and IP payload.

Abstract Data Type: unsigned64

Data Type Semantics: TBD

Name: octetDownstreamCount

Description: The total number of octets in downstream packets for this Flow at the Observation Point since the Metering Process (re-)initialization for this Observation Point. The number of octets includes IP header(s) and IP payload.

Abstract Data Type: unsigned64

Data Type Semantics: TBD

Name: applicationErrorCodeCount
Description: This Information Element identifies the number of packets with application layer error code detected.

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: fragmentIncompleteCount

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

Abstract Data Type: unsigned32

Data Type Semantics: TBD

Name: fragmentFirstTooShortCount

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

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: fragmentFlagErrorCount

Description: This Information Element specifies number of fragments with offset 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

A new value is added to FlowEndReason:

0x06: protocol exception timeout
The flow was terminated due to protocol state machine anomaly and unexpected timeout.

6. Security Considerations

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

7. References

7.1. Normative References


7.2. Informative References


Authors’ Addresses

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

Email: futianfu@huawei.com

Dacheng Zhang

Email: dacheng.zhang@gmail.com

Danping He
Huawei
Q14, Huanbao Yuan, 156 Beiqing Road, Haidian District
Beijing 100095
China

Email: ana.hedanping@huawei.com

Liang Xia (Frank)
Huawei
No.101, Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: frank.xialiang@huawei.com
Abstract

The document details use cases to mitigate DDoS attacks. These use cases are expected to illustrate involved communications to detect and mitigate DDoS attacks. It is expected that these communications will be in the future handled by the DDoS Open Threat Signaling (DOTS). These scenarios are intended to be useful to derive requirements for the design of DDoS Open threat Signaling.

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 October 22, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of
1. Introduction

DDoS is a major threat that affects any organization of any size. In addition, these attacks have become more and more frequent, complex and sophisticated which makes DDoS attacks harder to be detected at a single point.

More specifically, traditional SYN TCP or ICMP flood attacks were relatively easy to detect at the border of the network by an on-premise device. Although such DDoS attacks remain, DDoS attacks become more and more applications specific. This results in more specialized DDoS attacks, that require a fine grained monitoring to detect suspicious traffic.

For example, DNS can be used as a channel to establish a communication channel between a bot and its Command and Control (CC) channel. A generic DNS flow traffic monitoring is not sufficient to detect such attacks. Instead it may require monitoring FQDNs with NXDOMAIN associated to behavioral traffic analysis. DNS(SEC) or NTP are used to perform DDoS reflection attacks. Detection of these attacks may involve monitoring how the source IP address may be unusually associated to heavy traffic. That said, more specific traffic monitoring and analysis is not sufficient when DDoS attacks target a specific application. In the case of slowloris flows DDoS attacks for example, the attacker initiates regular conversations with the servers, except that it maintains these conversations open. The use of TLS/DTLS makes on path monitoring impossible.

Table of Contents

1. Introduction .......................... 2
2. Terminology and Acronyms ................ 3
3. On-premise use case .................... 4
   3.1. Symmetric .......................... 5
   3.2. Asymmetric ......................... 8
4. Cloud Use Case ........................ 10
5. Hybrid Cloud Use Case .................. 11
6. Security Considerations ................ 12
7. Privacy Considerations ................ 12
8. IANA Considerations .................... 12
9. Acknowledgments ....................... 12
10. Normative References .................. 12

Author’s Address ........................ 12
The complexity and the multitude of potential targets results in making DDoS detection a distributed system over a network. Flood attacks can be detected at the entrance of the network, SYN flood may be detected by firewalls associated to behavioral analysis. TLS and HTTP floods or low and slow and application based DDoS attacks are expected to be detected on the server side.

The multitude of DDoS monitoring appliance requires coordination. Coordination is necessary in order to manage the DDoS appliances as well as to collect the various information provided by each appliance and correlate these piece of information. Such correlation is expected to provide early detection, as well as more accurate alarms. Once a DDoS attack has been detected, the mitigation should proceed. Mitigation could be handled locally or outsourced.

The document details use cases to mitigate DDoS attacks. These use cases are expected to illustrates involved communications to detect and mitigate DDoS attacks. It is expected that these communications will be in the future handled by the DDoS Open Threat Signaling (DOTS). These scenarios are intended to be useful to derive requirements for the design of DDoS Open threat Signaling.

The document illustrates how DOTS makes possible DDoS to go beyond the scope of an isolated appliance and:

- A) Make possible a global and cross layered DDoS Monitoring, to make DDoS detection more accurate and earlier.

- B) Make possible a global and cross layer DDoS Mitigation, to mitigate in an coherent and efficient way.

- C) Make possible to share monitored information between multiple parties.

- D) Make possible to share and delegate DDoS monitoring and mitigation to third party.

2. Terminology and Acronyms

- Deny of Service (DoS): is an attack that makes resource of a service unavailable for its intended users. The resource may be computing or networking resource.

- Distributed Deny of Service (DDoS): is a DoS attack where the resources used by the attacker to perform the attack are distributed.
- DDoS Monitoring: designates the ability to inspect and monitor the traffic. This may include, exporting flow information to a Flow Repository or generating an alarm to the DDoS Controller when some threshold have been reached. In this document, DDoS Monitoring represents indifferently either a specific and dedicated DDoS Appliance, a virtual DDoS Appliance or a module.

- DDoS Mitigation: designates the ability to mitigate the DDoS attack. This may include providing filtering rules for example. In this document, DDoS Mitigation represents indifferently either a specific and dedicated DDoS Appliance, a virtual DDoS Appliance or a module.

- DDoS Controller: designates the entity that centralized monitoring, the alarms received and provides the mitigation actions. As DDoS attacks become more and more complex, a single DDoS monitoring device become dedicated to limited aspect of DDoS. As a result, these devices have only a fractional view of the ongoing activity. On the other hand, the DDoS Controller can aggregate and correlate this information have as such has a global view of the attacks. As result the DDoS Controller is more likely to take the appropriated decision to mitigate the attack.

- DDoS Appliance: designates an appliance that embeds DDoS Monitoring and/or DDoS Mitigation function. In this document, DDoS Appliance can be indifferently a hardware or virtual virtual DDoS Appliance.

- Flow Repository: designates the entity that centralized all the flow information. The Repository, may be shared between various entities and third parties. In fact, it is expected that information could be shared between independent actors, in order to mitigate DDoS Internet wild.

- Service: designates the destination of the traffic and the service that is under attack.

3. On-premise use case

The on-premise uses cases describe scenarios where DDoS is detected and mitigated on site. Section 3.1 describes the symmetric on-premise scenario, where the DDoS Appliance is place on path both the inbound and outbound traffic to the Service. Section 3.2, on the other hand presents the case where only a sub traffic is dynamically directed to the DDoS Appliance.
3.1. Symmetric

As depicted in Figure 1 the DDoS Appliance is on path of the inbound and outbound traffic to the Service. In other words, traffic coming from the Service to the end users goes also through the DDoS Appliance.

Such scenario may be associated to Small Office Home Office (SOHO) networks. In this case, the network, most likely, has a single DDoS Appliance. On the other hand, this scenario may also apply to large data center where, for example, each VM could be associated to a virtual DDoS Appliance.

The typical use case includes the following steps:

1. The DDoS Controller requests the DDoS Monitoring and DDoS Mitigation capabilities of the DDoS Appliance. Such request provides flexibility for both the DDoS Controller and the DDoS Appliances. First the DDoS Controller does not need to be tied to the DDoS Appliance, and so a single DDoS Controller may be used for various heterogeneous DDoS Appliances. Heterogeneity can be in term of vendors and/or in term of proposed capabilities. Similarly, this provides flexibility for the DDoS Appliances, as a DDoS Appliance may implement a subset of capabilities. In our example, the DDoS Controller, discovers both the DDoS Monitoring and DDoS Mitigating capabilities. DDoS Monitoring capabilities are necessary for monitoring the traffic and latter setting the alarms (see 2.). DDoS Mitigation capabilities are not mandatory to be requested here, as they are only expected to be used when the network is under attack. The reason the DDoS Controller requests those at this stage is to be able to plan its strategy for DDoS mitigation in advance instead of doing so while being under attack.

2. The DDoS Controller, then configures the appropriated capabilities on the DDoS Appliance. The configuration can typically be setting the thresholds upon which an alarm is raised by the DDoS Appliance to the DDoS Controller. Another type of setting may also be related to monitoring. DDoS Appliance may be configured to provide flow or resource (like CPU usage) information. These information may be exported to the Flow Repository in an appropriated format that enabled processing and correlation analysis by the DDoS Controller.

3. The DDoS Appliance sends the monitoring information to the Flow Repository. Note that the Flow Repository must be provided some means to authenticated the received packets as well as to
check the received information corresponds to the one requested by the DDoS Controller.

4. The DDoS Appliance raises an alarm that some suspicious traffic has been detected. This alarm corresponds to the settings performed by the DDoS Controller in step 2. As mentioned in Section 1 it may be difficult for the DDoS Appliance to determine from a local observation that a DDoS attack is ongoing or not. This is the reason the alarm is raised for suspicious traffic.

5. The DDoS Controller analyzes and correlates the received alarm for suspicious traffic and confirm or not that a DDoS attack is ongoing. Confirmation may require the DDoS Controller to perform some traffic analysis and correlates the alarm with some additional data. To do so, the DDoS Controller may consult the Flow Repository.

6. The DDoS Controller concludes that the network is under attack, and so proceeds to DDoS mitigation. In this example, the DDoS Controller is aware of the DDoS Mitigation capabilities of the DDoS Appliance as it has proceeded to the discovery mechanism in step 1. If that is not the case, the DDoS Controller should discover the DDoS mitigation capacities now. DDoS mitigations performed by the DDoS Controller are related to DDoS service. This may include for example setting some filtering rules or activation rate limitation. If traffic redirection should be performed, it is not expected to be performed by the DDoS Controller. In fact redirection implies a network reconfiguration and is considered outside the scope of the DDoS Controller. In addition to mitigate the DDoS attack, the DDoS Controller may also adjust its DDoS Monitoring settings. Motivations for doing so, may be for example to reduce the traffic on the network, or reversely, to provide a more accurate monitoring.

6bis. Eventually, the DDoS Controller may conclude that the network is not under attack. In this case the alarm is ignored or acknowledged to avoid the alert is re-sent and eventually load the network or the DDoS Controller. Similarly to step 6, the DDoS may also decide to adjust the monitoring settings to reduce false positive alarms. Note that the latest should be used cautiously as, such mechanism may be used as a vector of attack.
Figure 1: On-premise Symmetric Use Case

Figure 1 shows the DDoS Controller as distinct from the DDoS Appliance. In fact nothing prevents the DDoS Controller to be located on the DDoS Appliance. In this case the communications between the DDoS Controller and the DDoS Monitoring or DDoS Mitigation functions would be implementation dependent and thus outside of the scope of DOTS. The DDoS Appliance may embed a basic and limited DDoS Controller for basic configuration of the device. This is one reason why a DDoS Appliance may be configured by multiple DDoS Controllers.

Similarly, there is no requirements that the DDoS Controller belongs to the same network as the DDoS Appliances. The DDoS Controller could be placed inside the on-premise DDoS Appliances’ network or remotely see Section 5 for more details.

How the DDoS Controller handles alarms and determines a suspicious traffic corresponds or not to a DDoS attack is out of scope of DOTS. Similarly, the mitigating strategies are also out of scope of DOTS.
3.2. Asymmetric

The asymmetric on-premise scenario optimize resources compared to the symmetric on-premise scenario. More specifically, in the symmetric on-premise scenario, the traffic going from the Service to the end users also goes through the DDoS Appliance. Such deployment may lead to unnecessary load on the DDoS Appliance. In fact, the outbound traffic may not need to be either monitored or mitigated, and as such may reduce the packet rate or bit rate upper bound limit for inbound traffic. This may be one motivation for splitting the DDoS Monitoring module and the DDoS Mitigation modules in two different DDoS Appliances. In addition, for large networks, having a dedicated DDoS Appliance for DDoS mitigation may rationalize the cost and use of DDoS Mitigation Appliances. In fact, DDoS Mitigation Appliances may be shared by multiple Services or instances of VM of a given Service. As a result, the DDoS Mitigation Appliance do not need to scale the service traffic but instead the traffic of DDoS attacks -- which is most likely expected to remain smaller. This may not be the case for the DDoS Monitor Appliance as there is a need to always monitor the whole service traffic.

In the use case depicted by Figure 2 and Figure 3 the DDoS Mitigation Appliance only handles DDoS traffic.

The typical use case includes the following steps:

1. corresponds to the capabilities discovery phase. It is similar as the one exposed in Section 3.1. The main difference remains that DDoS Monitoring capabilities and DDoS Mitigating capabilities are discovered on two distinct DDoS Appliances.

2., 3., 4. and 5. corresponds to the monitoring and alarms settings. Monitoring may result in exporting data to the Flow Repository. This is similar as the steps described in Section 3.1.

6. If the DDoS Controller determines the network is under a DDoS attack, mitigation is performed in two steps. They may be ordered differently depending on criteria that are beyond the scope of this use case. First, the DDoS Mitigation is configured as described in Section 3.1 as a result of an analysis performed by the DDoS Controller. Then, traffic redirection is performed. In our case, the redirected traffic corresponds only to the inbound traffic from the end users. The traffic from the service to the end users is not redirected. This operation is not directly handled by the DDoS Controller. It can be performed manually, or upon a request from the DDoS Controller. This request is then treated by a
network management function in order to perform the appropriated network configurations.

6bis. In the case, the DDoS Controller determines the network is not under a DDoS attack, this step similar to the one described in Section 3.1.

Figure 2: On-premise Asymmetric Use Case Monitoring Phase
4. Cloud Use Case

Figure 4 illustrates the Cloud use case. In this scenario, the entire DDoS monitoring and mitigation service is outsourced to a third party designated as Cloud Based DDoS Cleaning Service or Cloud for short. In order to do so, the traffic associated to the Service goes through the Cloud Based DDoS Cleaning Service as detailed in Figure 4. On the other hand, this scenario makes DDoS mitigation transparent to the Service provider, which then benefits from a "clean pipe".

Figure 4 presents the case where the Cloud is on path of both inbound and outbound traffic, a similar scenario may also consider that only the inbound traffic, that is the traffic destined to the service is directed to the cloud whereas the outbound traffic destined to the users does not.

Internal organization of the Cloud Based DDoS Cleaning Service is transparent to the Service provider. A combination of the on-premises scenarios may be used.
5. Hybrid Cloud Use Case

The inconvenient the cloud use case scenario described in Section 4 is that redirecting the traffic to the cloud is likely to introduce additional latency. This is inconvenient as it adds a constant service degradation and cost to the Service provider. In order to address this, this section details the Hybrid Cloud scenario that combines the on-premise scenarios detailed in Section 3 and the cloud scenario detailed in Section 4.

The main driver for combining the cloud and on-premise scenarios is to be able to outsource the DDoS attack mitigation to a third party only when the Service provider is under attack, or when it is not able to handle the ongoing DDoS attack. In the general case, the determination on how the service provider is able to cope and detect a DDoS attack is up to the Service provider. A continuum of scenarios can be considered and this section details only a few of them.

A specific case may consider that DDoS mitigation is outsourced by outsourcing the DDoS Controller to a third party. This DDoS Controller, drives the DDoS Monitor functions on the premise. When an alert is raised, the DDoS Controller may take the decision to mitigate internally with the DDoS attack only using on-premise facilities. This case correspond to the scenarios detailed in Section 3, except that the DDoS Controller is either located remotely, or at least accessed remotely by the third party.
other hand, the DDoS Controller may also decide that the DDoS attack
cannot be mitigated on premise, and that mitigation should be
outsourced to a cloud service as described in Section 4. In this
case, the DDoS Controller is expected to redirect at least the
inbound traffic of the Service provider to the cloud infrastructure.
This case corresponds to the on premise asymmetric scenario detailed
in Section 3.2. The difference is that redirection does not occur
inside the Service provider, but involves sites redirection -- most
likely using BGP signaling.

Another scenario may provide more independence to the Service
provider. In this scenario, the Service provider, may have the
complete control on the DDoS Monitor and DDoS Mitigation Appliances,
and only uses the Cloud as a backup solution when it is not likely to
deal with the DDoS attack. In this case, the DDoS Controller sends
an alert to the DDoS Controller of the third party. The third party
first analyzes the attack, which may require to grant access to the
third party to the Flow Repository. If DDoS mitigation action are
performed by the third party DDoS Controller, means should be
provided to transmit information from the third party DDoS Controller
to the DDoS Appliances. This could be done for example by providing
access to the DDoS Appliances, or by DDoS Controller that acts as a
proxy for the third party DDoS Controller.

6. Security Considerations
7. Privacy Considerations
8. IANA Considerations

This document makes no request of IANA.

9. Acknowledgments
10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Author’s Address
Abstract

This document discusses the requirements for a protocol sufficient for the goals of the DDoS Open Threat Signaling (DOTS) Working Group.

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 8, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
# 1. Introduction

## 1.1. Overview

As DDoS attack scale and frequency continue to grow, a number of cloud mitigation providers have emerged to offer on-demand traffic scrubbing services. Each service offers its own ad hoc interfaces for subscribers to request threat handling, leaving subscribers tied
to proprietary implementations that are not portable from service to service. These ad hoc implementations also severely limit the subset of network elements capable of participating in any coordinated attack response.

The current lack of a common method to make inter-domain threat handling requests and share realtime attack telemetry hampers response coordination. The DOTS Working Group has assigned itself the task of standardizing a protocol or protocols to address that lack.

The requirements for these protocols are unusually stringent. The data link between signaling elements may be saturated with attack traffic—likely inbound, but outbound congestion must also be considered—and the signaling elements cannot rely on the availability of an out-of-band channel to report the attack and request threat handling. High packet loss rates are to be expected, rendering every round trip uncertain.

As such, the protocol which DOTS develops or adapts must have certain characteristics tending to increase the probability of signal delivery between endpoints. At the same time, the protocol must be rich enough to support not only simple calls for aid and limited attack telemetry, but also extensibility such that DOTS is adaptable to future needs.

1.2. 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].

2. Terminology

The following terms are meant to help define relationships between elements as well as the data they exchange:

2.1. Attack Telemetry

Attack Telemetry is a catch-all term for collected network traffic characteristics defining the nature of a DDoS attack, and which contributes to the detection, classification, traceback, and mitigation of the attack.

In addition to the properties defining IP Traffic Flow as described by [RFC3917], the Attack Telemetry may include information like:
o traffic rates from attacker sources in packets and bytes per second,
o detected attack class (e.g., reflection/amplification, resource exhaustion, etc.),
o attack duration

as well as any other information deemed valuable for attack response by the Working Group.

2.2. Configuration Channel

The Configuration Channel is a RESTful [REST] interface to establish a common understanding of signal and threat handling between the Signaler and Signal Handler. The RESTful interface enables local operator control over DOTS elements.

2.3. Signal Channel

The Signal Channel refers to the bidirectional communication layer established between the Signaler and the Signal Handler, over which Signals and Signal Responses are transmitted.

2.4. Signal

A DOTS Signal is a message sent from a Signaler to a Signal Handler. The Signal carries information necessary to identify the Signaler and communicate Signaler intent, attack insight to the Signal Handler, and indicators useful for measuring Signal Channel Health.

A Signal permits a Signaler to request threat handling.

2.5. Signal Response

A DOTS Signal Response is a message sent from Signal Handler to a Signaler. A Signal Response is a variation of a Signal, in that it includes data identifying the originating Signal Handler and indicators of Signal Channel Health. The Signal Response will also include information describing the status of any ongoing threat handling undertaken at a Supplicant’s request.

Note that Signal Responses are sent without solicitation by a Signaler. That is, a Signal Handler sends Signal Responses to an established Signaler regardless of whether the Signal Handler has received a Signal message. (See Signal Channel Health below.)
2.6. Signaler

The DOTS endpoint transmitting a Signal to a Signal Handler in order to communicate Attack Telemetry and request or withdraw a request for threat handling. When a Signaler requests threat handling from the Signal Handler, the Signaler is called a Supplicant.

A Signaler MAY establish Signal Channels with multiple Signal Handlers.

2.7. Supplicant

A DOTS Supplicant is a Signaler requesting threat handling from the Signal Handler. The Supplicant is often downstream of the attack from the Signal Handler, so the Supplicant will often be requesting attack response closer to the sources of attack.

2.8. Signal Handler

The DOTS endpoint responsible for processing and responding to Signals received from a Signaler. A Signal Handler may or may not be in the same domain as the Signaler. When a Supplicant requests threat handling, the Signal Handler is responsible for communicating that request to the entities tasked with the attack response. The attack response itself is out of scope for DOTS, but the Signal Handler should transmit Signal Responses with threat handling feedback to the Supplicant.

Note that Signal Handler and Threat Handler are often but not always synonymous.

2.9. Signal Relay

A DOTS node acting as a Signal Handler and a Signaler. In the role of a Signal Handler, a Signal Relay receives Signals from a downstream Signaler, and then acts as a Signaler when relaying the Signal to an upstream Signal Handler. A Signal Relay also relays any responses from upstream to the originating Signaler.

2.10. Threat Handler

The Threat Handler is the entity or collection of entities tasked with handling an attack at the request of a Supplicant. The Threat Handler and Signal Handler may be one and the same, but are not required to be.
3. Protocol

This section examines requirements for successful threat signaling.

The Working Group has thus far focused attention on adapting IPFIX as a possible vehicle for the DOTS protocol. The expectation as described in [I-D.teague-open-threat-signaling] is that IPFIX’s templating system will provide DOTS the necessary flexibility and extensibility, while the wide availability of IPFIX will lower the bar for adoption among vendors. The IANA registry of IPFIX Informational Entities [IANA-IPFIX-IE] similarly increases the appeal of IPFIX by eliminating the need to define a variety of field types.

However, the ultimate selection of IPFIX as the foundation of the DOTS protocol is by no means certain at this stage. It is our hope that by reaching a common understanding of protocol requirements the Working Group will be able to make rapid progress defining the protocol itself.

3.1. Operation

One of the unusual aspects of DOTS is that it depends not so much on protocol reliability but on protocol resiliency. Signal lossiness, to a greater or lesser degree, is to be expected, and the protocol must continue to operate regardless.

DOTS should be able to absorb the loss of multiple consecutive Signals or Signal Responses and still operate nominally, relying on measures like redundant message transmission to increase the likelihood of successful delivery. By the same token, the protocol demands the DOTS nodes share a common understanding of a failed signal channel.

This section discusses the protocol characteristics required to achieve the necessary resiliency, while also retaining the signal effectiveness sought by the Working Group.

3.1.1. Endpoint Communication

A synchronous message-oriented protocol is ill-suited for the conditions under which DOTS is expected to operate. Such a protocol would require a level of reliable message delivery in either direction that we cannot depend on for DOTS.

In contrast, an asynchronous message-oriented protocol fits DOTS requirements, offering resiliency even when dealing with a high level of signal lossiness. As long as the protocol includes indicators showing the time or sequence of the last message received by the
peer, each endpoint can continue signaling, and incorporate the most recent data from its peer when messages arrive.

In practice, the Signaler sends messages to the Signal Handler, regardless of responses from Signal Handler, and the Signal Handler does the same in the opposite direction. Until an endpoint detects fail health continue to arrive at each endpoint, DOTS is operational.

3.1.1.1. Signal Channel Health

Monitoring signal delivery success rates is vital to normal DOTS operations. The protocol SHOULD include a way for each endpoint to detect when their respective peers last received a message. This could be achieved through inclusion of timestamps or sequence numbers in the signal messages.

With this method for detecting signal lossiness in place, any received DOTS message acts as a signal heartbeat, meaning no additional keep-alive messages are needed.

Should too much time elapse since an endpoint last received a message from its peer, the endpoint SHOULD consider the peer unresponsive, and in some way alert the operator to the loss of signal. Similarly, if messages continue to arrive from the peer, but the timestamp or sequence number do not update in spite of repeated message transmissions to the peer, the signal MUST be considered degraded, and an appropriate alert should be delivered to the operator.

The method of alerting is out of scope. The endpoints may agree upon the signal failure time-to-live using the configuration channel.

3.1.2. Message Frequency

TODO

3.1.3. Redundant Signal Channels

A Signaler may wish to establish Signal Channels with multiple Signal Handlers in the same domain to increase the likelihood that a Supplicant request for threat handling will be honored.

3.1.4. Redundant Transmission

The likelihood of packet loss due to congestion caused by, for example, a volumetric attack diminishes the resiliency of the protocol. A low-cost method to increase the probability of successful message delivery is through redundant message transmission at send time.
3.2. Transport

As noted above, the DOTS signal protocol does not require reliable, in-order delivery to be effective. The protocol may indeed become less reliable in the attempt to ensure all signal messages are delivered in the order sent, as pathological network conditions lead to missed delivery acknowledgments from the peer. In the worst case, none of the transport acknowledgements reach the signaler, resulting in spurious dead peer detection and subsequent connection teardown.

As such, it is RECOMMENDED that the DOTS protocol use connectionless transports like the User Datagram Protocol (UDP) [RFC0768]. While UDP imposes some additional work on the protocol, the minimal overhead for transmission aligns with DOTS requirements for protocol resiliency.

3.2.1. Congestion Control Considerations

A DOTS signal channel will not contribute to link congestion, as the protocol’s transmission rate will be negligible regardless of network conditions.

3.2.2. Alternative Transport Considerations

Where additional constraints imposed by middlebox limitations, overly aggressive filtering, or network policy disqualify UDP, TCP MAY be used for the Signal Channel. However, TCP remains a poor choice for inter-domain signaling over a saturated link for the reasons described above, and consideration should be given to using a Signal Relay between the Signaler and the remote domain’s Signal Handler.

3.2.3. Message Size

DOTS protocol messages MUST be smaller than the path maximum transmission unit (MTU) to avoid fragmentation. In the lossy network conditions under which DOTS is expected to operate, fragmentation unnecessarily increases the likelihood of message delivery failure, as a single lost fragment will cause the entire message to be discarded.

3.2.4. Transport Security

The DOTS Working Group charter describes the need to ensure "appropriate regard for authentication, authorization, integrity, and authenticity" in any developed or adapted protocols.
3.2.4.1. DTLS

On the surface, Datagram Transport Layer Security [RFC6347] would seem to be the obvious choice to meet those requirements. However, the conventional three-way TLS handshake using public-key infrastructure incurs significant overhead. The elevated likelihood of handshake failure due to saturated links or otherwise hostile network conditions may be unacceptable for DOTS.

Some of this overhead may be eliminated using preshared keys (e.g., [RFC5487] and [RFC5489]), but the round-trip overhead of the three-way handshake is less easily overcome. The current drafts for TLS 1.3 [I-D.ietf-tls-tls13] make some headway in this regard, introducing a 1-RTT TLS handshake. This is a vast improvement for DOTS operations, but the timeline for standardization and vendor implementation is uncertain.

Regardless of TLS handshake innovation, DTLS by itself lacks a way to detect dead peers. The DTLS Heartbeat Extension [RFC6520] resolves this, but represents another messaging layer likely to be affected by network lossiness. In addition, the DTLS Heartbeat extension requires immediate responses to heartbeat requests, with the requester retransmitting up to the limit defined in [RFC6347]. The DTLS Heartbeat Extension indicates a DTLS session SHOULD be terminated if the peer does not respond after the retransmission limit is reached. Given the unpredictability of message delivery in the typical DOTS scenario, this rigidity only adds to concerns about the aptness of DLTS for DOTS transport security.

3.2.4.2. Continued Evaluation

The DOTS Working Group will need to evaluate available options for meeting the goal of providing protocol confidentiality, integrity, and authenticity. Guidance should be sought from the TLS Working Group as appropriate.

Guidance and insight may also be found in the DTLS in Constrained Environments [DICE] Working Group. The DICE WG is currently evaluating and developing techniques for transport security in network conditions that may be similar to those in which DOTS will need to work.

3.3. Message Data

As we note above, the Working Group has thus far focused on the suitability of IPFIX as the DOTS signaling protocol. This section makes no judgment in that regard.
3.3.1. Signal

In addition to the requirements laid out in the Protocol Operation section, the Signal MUST be able to:

- provide such attack telemetry as is available to the signaler,
- permit the signaler to request or withdraw a request for intervention from the signal receiver,
- permit the signaler to request refinement or expansion of the scope of threat handling performed by the signal receiver,
- allow customization to the extent required to adapt to emerging requirements or local needs.

3.3.2. Signal Response

In addition to the requirements laid out in the Protocol Operation section, the Signal Response SHOULD deliver feedback to the signaler from the entity or entities handling a threat on the signaler’s behalf.

Feedback would include threat handling status, threat handling scope, blocked packet and byte counters, and so on.

4. Configuration Channel

The Configuration Channel would permit local operator control over threat handling by a Signal Handler.

Configurable features might include:

- Signaler address space protection preferences
- Static Black/White lists to apply during threat handling
- UUID assigned to the Signaler by the Signal Handler, which the Signaler must include in subsequent Signals
- Other information not well-suited to transmission under attack conditions

4.1. Configuration Protocol

An obvious choice for the configuration protocol is a RESTful interface over a secure HTTP [RFC2616] channel. Such interfaces are well-understood and easily adopted. With configuration as a concern,
5. IANA Considerations

TODO

6. Security Considerations

The DOTS Working Group was formed to standardize methods for realtime inter-domain threat signaling. Any protocols must therefore be capable of transmitting information over public networks, with consequent requirements for message integrity, confidentiality, and authenticity.

Transport security and message authenticity are addressed above. In the event either is compromised, regardless of the method involved, the security risks exposed include:

- attack telemetry forgery
- threat handling request forgery
- Denial of Service (DoS) attacks

In scenarios in which DOTS endpoints are communicating across public networks, the endpoints are themselves subject to attack. Endpoint operators SHOULD take steps to restrict access as much as possible to known valid peers through application of network policy and peer authentication.

TODO

7. Acknowledgments

Roland Dobbins deserves full credit for suggesting the Signal Relay, as well as thanks for his insight and his generosity with his time as we discussed the topics that led to this draft.

Thanks to Larry Huston, Sean O’Hara, and David Watson for discussions and support.

8. References
8.1. Normative References


8.2. Informative References


Author’s Address

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

Email: amortensen@arbor.net
Abstract

This document discusses the need and the mechanisms to dynamically update configuration of network monitoring devices to help identify distributed denial-of-service (DDoS) attacks in a network. Once an attack is signalled by a client or detected locally, provisioning cycles are triggered to program a set of network elements to undertake appropriate actions (including, blackhole, drop, rate-limit, or add to watch list) on the suspect traffic.

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 31, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
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 or router, a firewall, or an entire network, etc. Typically, enterprises configure Network Elements and Monitoring Devices (appliances) to export traffic flow information for further processing by applications hosted on other devices, such as DDoS monitoring applications.

DDoS monitoring applications analyze and correlate flow records to baseline proper behaviour and measure deviation from that expected norm ("Observed" vs. "Expected"). Analytics is applied to deliver a baseline of the network in normal operation conditions and then to highlight when an anomalous event occurs. As DDoS attacks get more complex and more sophisticated, DDoS monitoring applications may need more or different fields in the flow records, change the frequency of flow record collection, increase the granularity of flow record collection for traffic to a network resource, tweak the sampling logic, enable or disable packet sampling, modify the packet selection technique for sampling, etc., to adjust their decision-making process for a better detection efficiency.
This document explains mechanisms to dynamically change the configuration of IPFIX-compliant Monitoring Devices ([RFC7011]) and PSAMP-compliant Monitoring Devices ([RFC5476]) using the Network Configuration Protocol (NETCONF) ([RFC6241]) to identify attacks on the network and once an attack is detected, use NETCONF to carry instructions meant to dynamically enforce appropriate filtering rules on a set of network devices. In addition to the required intelligence to decide which actions are needed, a decision-making process to decide "where" (i.e., which network elements) these filtering actions are to be performed.

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. Terminology

This document makes use of the following terms:

- Network Element: refers to a node that is involved in the delivery of connectivity services. A Network Element can be a router, a switch, a service function (e.g., firewall), etc.

- DOTS Client: Refers to the entity that is responsible for signalling an attack. The entity could be a network resource (e.g., Network application) subjected to attack or flow collector, firewall, CPE etc. detecting attack on the network.

- DOTS Controller: Refers to the entity that is responsible for undertaking appropriate actions to satisfy the requests from a DOTS Client.

- Flow Collector: Refers to the functional entity that is responsible for instructing the Network Elements about the monitoring strategy. It is also responsible for collecting monitoring information from the network. One or multiple Flow Collectors may be enabled. Considerations about internal communications between multiple Flow Collectors are out of scope. A Flow Collector may be collocated with a DOTS Client.

- Configuration Manager: Refers to an entity that is responsible for the provisioning of a set of Network Elements.

- Monitoring Devices: These are devices in the network that are provisioned to monitor network flows, collect information and export them to a "Flow Collector."
4. Solution Overview

Flow collector (or DDoS monitoring application) needs to program IPFIX- and PSAMP-compliant Monitoring Devices using vendor-independent configuration data model. A vendor-independent configuration data models helps to store and manage the configuration data of Monitoring Devices in a consistent format. The data model could be specified using YANG [RFC6020] to dynamically configure Monitoring Devices. The configuration data models for IPFIX and PSAMP are discussed in [RFC6728].

In order to offer more automation and dynamicity in changing the configuration of network monitoring, this document proposes an architecture that is composed of two parts:

1. Flow Collector communicates the configuration of network monitoring to the DOTS Controller. This assumes the Flow Controller has been provisioned with the locator(s) of DOTS Controller(s) to contact. For multi-homed networks, the Flow Controller should contact the DOTS Controller attached to the network from which the suspect traffic is received from.

2. The DOTS Controller is responsible for configuring the Monitoring Devices. This assumes the DOTS Controller has access to the underlying network topology (including the interconnection map and the set of advanced service functions).
1. Initial DDoS monitoring provisioning cycle

2. Configure monitoring devices using NETCONF

5. DDoS monitoring re-provisioning cycle

DOTS Controller

Config manager

Switch (..) Middle box

Router

4. Update/Modify monitoring config

Flow collector

Analyzer

3. Export IPFIX messages, packet samples to flow collector

Figure 1: Configuration Cycle for IPFIX

Figure 1 provides a high level overview of the solution. The proposed solution is to build a dynamic configuration model in DDoS Monitoring using a feedback system where a Flow Collector can influence monitoring configurations on the devices to gather information about a potential DDoS event.

The sequences marked (1)-(5) in Figure 1 refer to the work flow of the proposed solution, these flows can be broadly categorized into three phases:
1. Initial Provisioning Cycle: Represents the initial state of the monitoring configuration where an administrator updates the Controller with a default or preliminary monitoring configuration delivered to Monitoring Devices. For example, the initial configuration on the Monitoring Devices is to collect information elements such as IP addresses/prefixes, application type, transport ports, flow timestamps, interfaces and so on.

2. Flow Monitoring: Refers to the activity of Monitoring Devices to inspect and watch network flows. Based on the monitoring configuration, the Monitoring Device is instructed to collect specific flow information and export them to a "Flow Collector".

3. Flow Collection and Analysis: A "Flow Collector" device collects and (possibly) aggregates flow information from one or more Monitoring Devices. As the Collector continues to gather more and more data, it can potentially correlate and analyze flow information to "guess" or determine if a DDoS event is in progress. If so, the Flow Collector may consider gathering additional data from the Monitoring Devices and signals this intent to a "Controller".

4. Re-provisioning Cycle: The Controller receives from the "Flow Collector", the intent to re-provision Monitoring Devices to produce additional flow information elements. The Controller, then delivers the new or updated configuration to the appropriate Monitoring Devices.

The other provisioning interface is the one between the DOTS Controller and Network Elements. Concretely, when the Flow Collector identifies an active attack, it signals to the DOTS Controller the set of traffic identification information (including all suspect IP addresses) together with a suggested action (e.g., rate-limit, drop, monitor). Then, the DOTS Controller propagates the filtering rules to the Network Elements (including routers, middleboxes). The Flow Collector, after certain duration, requests the rules to block traffic from these IP addresses be removed once the attack has stopped. Means to detect an attack is not valid anymore may be static (an administrative decision) or dynamic (based on an analysis of the traffic).

Note, [RFC6088] provides typical information that can be included in the traffic identification information set.
a. Configure network devices using NETCONF
b. Configuration ACK/NACK

Figure 2: Configuration Cycle for Attack Mitigation

As shown in Figure 3, two distinct interfaces are defined: the one used by a Flow collector to signal appropriate filtering rules to a DOTS Controller (for example, [I-D.reddy-dots-transport] can be used for this interface) and the one to enforce policies in the appropriate nodes (for example, NETCONF can be used).
DOTS Client and DOTS controller could be located in different administrative domains. Local decisions (e.g., install filters) can be made locally by the DOTS Controller. A notification is then sent to the DOTS Clients using the signaling interface. Concretely, the decision-making process of the DOTS Controller can be based on events that are reported by other DOTS Clients, local monitoring tools, etc. Appropriate notifications and feedback objects should be carried over the signaling interface.

The signaling interface can also be used by a DOTS Controller to request a confirmation from a DOTS Client about the enforcement of a filter. For example, this can occur when the DOTS Controller detects that some traffic is likely to be a DoS, before undertaking actions on Network Elements, the DOTS Controller contacts first the DOTS Client to double check whether that traffic is really a DoS. Upon confirmation from the DOTS Client, the DOTS Controller initiates a configuration cycle accordingly.
5. Security Considerations

The authentication mechanism between the Flow Collector and DOTS Controller should be immune to pervasive monitoring [RFC7258]. An attacker can intercept traffic by installing rules that would lead to redirect all or part of the traffic to an illegitimate Flow Collector. Means to protect against attacks that would lead to install, remove, or modify rules must be supported.

In order to protect against denial of service that would be caused by a misbehaving trusted Flow Collector, DOTS Controller should rate limit the configuration changes received from a Flow Collector.

6. Acknowledgements

Thanks to C. Jacquenet for the review.

7. References

7.1. Normative References


7.2. Informative References


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

Prashanth Patil
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore
India

Email: praspati@cisco.com

Mike Geller
Cisco Systems, Inc.
3250
Florida  33309
USA

Email: mgeller@cisco.com

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California  95134
USA

Email: dwing@cisco.com
Sandeep Rao  
Cisco Systems, Inc.  
Cessna Business Park, Varthur Hobli  
Sarjapur Marathalli Outer Ring Road  
Bangalore, Karnataka  560103  
India  

Email: rsandeep@cisco.com  

Mohamed Boucadair  
France Telecom  
Rennes  35000  
France  

Email: mohamed.boucadair@orange.com
Abstract

This document discusses mechanisms that a downstream Autonomous System (AS) can use, when it detects a potential Distributed Denial-of-Service (DDoS) attack, to request an upstream AS to perform inbound filtering in its ingress routers for traffic that the downstream AS wishes to drop. The upstream AS can then undertake appropriate actions (including, blackhole, drop, rate-limit, or add to watch list) on the suspect traffic to the downstream AS thus reducing the effectiveness of the attack.

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 31, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
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. The reader may refer, for example, to [REPORT] that reports the following:

- Very large DDoS attacks above the 100 Gbps threshold are experienced.
- DDoS attacks against customers remain the number one operational threat for service providers, with DDoS attacks against infrastructures being the top concern for 2014.
Over 60% of service providers are seeing increased demand for DDoS detection and mitigation services from their customers (2014), with just over one-third seeing the same demand as in 2013.

Enterprises typically deploy DDoS monitoring appliances that are capable of inspecting and monitoring traffic to detect potential DDoS threats and generate alarms when some thresholds have been reached. Most of these tools are offline; further steps are required to introduce online tools that would have immediate effects on traffic associated with an ongoing attack. Thanks to the activation of dynamic cooperative means, countermeasure actions can be enforced in early stages of an attack, which can optimize any service degradation that can be perceived by end users.

This document describes a means for such enterprises to dynamically inform its access network of the IP addresses that are causing DDoS. The access network 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 cooperation contributes to ensure a highly automated network that is also robust, reliable and secure.

The advantage of the proposed mechanism is that the upstream AS can provide protection to the downstream AS from bandwidth-saturating DDoS traffic. The proposed mechanism can also be coupled with policies to trigger how requests are issued. Nevertheless, it is out of scope of this document to elaborate on an exhaustive list of such policies.

How a server 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.

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.

The complexity and the multitude of potential targets result in making DDoS detection a distributed system over a network. Flood attacks can be detected at the entrance of the network, SYN floods may be detected by firewalls associated to behavioral analysis. 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. Other possible DDoS attacks are discussed in [RFC4732].

In each of the cases described above, if a network resource detects a potential DDoS attack from a set of IP addresses, the network resource informs its servicing router of all suspect IP addresses that need to be blocked or black-listed for further investigation. That router in-turn propagates the black-listed IP addresses to the access network and the access network blocks traffic from these IP addresses to the customer network thus reducing the effectiveness of the attack. The network resource, after certain duration, requests the rules to block traffic from these IP addresses 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.

4. Protocol Requirements

The protocol requirements for co-operative DDoS mitigation are the following:

- Acknowledgement for the processing of a filtering request and the enforcement of associated countermeasures.
- Mechanism to delete a configured rule.
- Mechanism to convey lifetime of a rule.
- Mechanism to extend the validity of a rule.
- Mechanism to retrieve a list of filtering rules.
- Protocol needs to support "forward compatibility" where the network resource can tell the network entity what version it supports and vice-versa. Any protocol describing attack
mitigations needs forwards compatibility so that new attacks can be described while still allowing older peers (who do not yet understand the new attack) to provide some mitigation.

- The mechanism should support the ability to send a request to multiple destinations (e.g., multi-homing cases).
- Because multiple clients may be allowed to send requests on behalf of a downstream node, the mechanism should allow to signal conflicting requests.
- The request to install a filter may indicate an action (e.g., block, add to a watch list, etc.).
- The mechanism must be transported over a reliable transport.

The security requirements for co-operative DDoS mitigation are the following:

- There must be a mechanism for mutual authentication between the network resource that is signaling black-list rules and the network entity that uses the rules either to propagate the rules upstream or enforces the rules locally to block traffic from attackers.
- Integrity protection is necessary to ensure that a man-in-the-middle (MITM) device does not alter the rules.
- Replay protection is required to ensure that passive attacker does not replay old rules.

5. Protocols for Consideration

An access network can advertise support for filtering rules based on REST APIs. A CPE router should use RESTful APIs discussed in this section to inform the access network of any desired IP filtering rules. If the access network does not advertise support for REST, BGP can be used. The means by which an access network can make this advertisement is outside the scope of this document.

5.1. REST

A network resource could use HTTP to provision and manage filters on the access network. The network resource authenticates itself to the CPE router, which in turn authenticates itself to a server in the access network, creating a two-link chain of transitive authentication between the network resource and the access network. The CPE router validates if the network resource is authorized to signal the black-list rules. Likewise, the server in the access network validates if the CPE router is authorized to signal the black-list rules. To create or purge filters, the network resource sends HTTP requests to the CPE router. The CPE router acts as HTTP proxy, validates the rules and proxies the HTTP requests containing the black-listed IP addresses to the HTTP server in the access network.
When the HTTP proxy receives the associated HTTP response from the HTTP server, it propagates the response back to the network resource.

If an attack is detected by the CPE router then it can act as a HTTP client and signal the black-list rules to the access network. Thus the CPE router plays the role of both HTTP client and HTTP proxy.

![Network Diagram](image)

**Figure 1**

 JSON [RFC7159] payloads can be used to convey both filtering rules as well as protocol-specific payload messages that convey request parameters and response information such as errors.

5.1.1. Install black-list rules

An HTTP POST request will be used to push black-list rules to the access network.

```plaintext
POST {scheme}://{host}:{port}/.well-known/{version}/{URI suffix}
Accept: application/json
Content-type: 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 2: POST to install black-list 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 same downstream network. This identifier must be generated by the
client and used as an opaque value by the server. This document does not make any assumption about how this identifier is generated.

traffic-protocol: Valid protocol values include tcp and udp.

source-protocol-port: For TCP or UDP: the source range of ports (e.g., 1024-65535).

destination-protocol-port: For TCP or UDP: 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].

destination-ip: The destination IP addresses or prefixes.

source-ip: The source IP addresses or prefixes.

lifetime: Lifetime of the policy in seconds. Indicates the validity of a rule. Upon the expiry of this lifetime, and if the request is not reiterated, the rule will be withdrawn at the upstream network. A null value is not allowed.

traffic-rate: This field carries the rate information in IEEE floating point [IEEE.754.1985] format, units being bytes per second. A traffic-rate of ’0’ should result on all traffic for the particular flow to be discarded.

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.

Note: administrative-related clauses may be included as part of the request (such a contract Identifier or a customer identifier). Those clauses are out of scope of this document.

The following example shows 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 provide HTTPS web service.
POST https://www.example.com/.well-known/v1/acl
Accept: application/json
Content-type: application/json

{
    "policy-id": 123321333242,
    "traffic-protocol": "tcp",
    "source-protocol-port": "1-65535",
    "destination-protocol-port": "443",
    "destination-ip": "2001:db8:abcd:3f01::/64",
    "source-ip": "2002:db8:6401::1",
    "lifetime": 1800,
    "traffic-rate": 0,
}

Figure 3: POST to install black-list rules

5.1.2. Remove black-list rules

An HTTP DELETE request will be used to delete the black-list rules programmed on the access network.

DELETE {scheme}:://{host}:{port}/.well-known/{URI suffix}
Accept: application/json
Content-type: application/json

{
    "policy-id": number
}

Figure 4: DELETE to remove the rules

5.1.3. Retrieving the black-list rules installed

An HTTP GET request will be used to retrieve the black-list rules programmed on the access network.
1) To retrieve all the black-lists rules programmed by the CPE router.

GET {scheme}://{host}:{port}/.well-known/{URI suffix}

2) To retrieve specific black-list rules programmed by the CPE router.

GET {scheme}://{host}:{port}/.well-known/{URI suffix}
Accept: application/json
Content-type: application/json
{
   "policy-id": number
}

Figure 5: GET to retrieve the rules

5.1.4. TBD

TBD

1. A CPE router can optionally convey metadata describing the attack type and characteristics of the attack to the access network. In some cases, especially with new forms of attack that don’t fit existing mitigation mechanisms or exceed network or mitigation capacity, the attack can’t be slowed or stopped. The access network might be able to signal its inability to stop the attack (if it is aware) or might be unaware that the attack continues to flow. In such cases where the attack continues, even after filters are requested and installed, the CPE may still need to obtain DDoS mitigation from an external service, outside the scope of this document.

2. The network resource periodically queries the CPE router to check the counters mitigating the attack and the query is recursively propagated upstream till it reaches the access network that has blocked the attack. If the network resource receives response that the counters have not incremented then it can instruct the black-list rules to be removed.

5.2. 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 CPE router will not receive an acknowledgment for the filtering request, the CPE router should monitor and apply similar rules in its own network in cases where the upstream network is unable to enforce the filtering rules.
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.

6. IANA Considerations

TODO

7. Security Considerations

If REST is used then HTTPS must be used for data integrity and replay protection. TLS based on client certificate or HTTP authentication must be used to authenticate the network resource signaling the black-list rules.

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.

8. Acknowledgements

Thanks to C. Jacquenet for the discussion and comments.

9. References

9.1. Normative References


9.2. Informative References


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.
Bangalore
India

Email: praspati@cisco.com

Mike Geller
Cisco Systems, Inc.
3250 Florida  33309
USA

Email: mgeller@cisco.com

Mohamed Boucadair
France Telecom
Rennes  35000
France

Email: mohamed.boucadair@orange.com
Abstract

This document defines a method by which a device or application may signal information relating to current threat handling to other devices/applications that may reside locally or at the service provider. The initial focus is ddos mitigation; however, the method may be extended to communicate any threat type. This will allow for a vendor or provider agnostic approach to threat mitigation utilising multiple layers of protection as the operator sees fit.

The dissemination of threat information will occur utilising JSON RPC API over HTTPS communications between devices/applications and will be augmented by IPFIX and UDP or SCTP for signaling telemetry information relating to attacks and protected object data.

An open standards based approach to communication between on-premise DDoS mitigation devices and service provider based DDoS protection services allows for enterprises to have a wider range of options to better secure their environments without the limitations of vendor lock-in.

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, 2016.
1 Introduction

There are many devices and applications dealing with threat handling that may be a discrete part of a larger strategy. These elements may be required from time to time to signal to an upstream component or provider that the capabilities of the device are exceeded and that an offramping of attack traffic to a more capable element or infrastructure is desired or required. Signaling the need to off-ramp is not the only necessary feature; however, it is also desirable to communicate the form that the threat takes in order to accelerate the next layer mitigation process.

Although many vendors and providers implement their own variation or invest in integrating disparate APIs, we are proposing the adoption of a standard method for elements to signal allowing greater integration among any on-premise device or service provider. In addition to the goal of interoperability, the intent is to present a robust method capable of continued signaling in the event of congested ingress paths to the originator. Stateful transport exchanges between components may leverage recognized JSON API channels in order to pass white & black lists, export collector information, protected object attributes, signature updates, mitigation details etc. These exchanges can occur at regular intervals during times of relative inactivity and could continue during attacks up to the point where a signaling component or path becomes overwhelmed. In parallel, a UDP or SCTP IPFIX channel will export data pertaining to protected objects as well as current and ongoing incidents. The receiver for export will be delivered to the signaling component via the JSON API channel, allowing for the upstream element to set the destination dynamically. The UDP or SCTP export will focus more specifically on communicating the current state of the threat and the component dealing with it. Should the signaling component risk becoming degraded, the telemetry data passed from this node will communicate this risk while also ensuring an upstream device or provider has the required information to take over traffic handling without the need to relearn and re-detect. Should an upstream element take over mitigation of an incident, this element will signal the ongoing status of the threat handling to the originating entity to ensure increased awareness to support effective decision making.

2 Data Dictionary

The data dictionary refers to a set of attributes common across implementations. The dictionary is not exhaustive and expansion is encouraged. The object definitions, as presented in this draft, are intended to communicate an event. An event will include a number of attributes which will identify an attack profile, the targeted
resource, any existing mitigation actions being undertaken, sla information and scope. In certain circumstances, such as initial registration and discovery, it may be desirable to export information regarding protected objects currently managed by the signaling component outside the scope of any threat or action. These may be identified as informational when the record is accompanied by an event key of 0. For the sake of initial scoping the attributes are expressed in descriptive terms. Reuse of existing IPFIX field names, where appropriate, is to be encouraged.

The event attributes will appear:

- Access Token
- Key
- Time
- Type (category and subtype)
- Description
- Scope
- SOS
- Thresholds

* Access Token - authentication token (e.g. pre-shared nonce)
* Key - the signaling component specific event identifier.
* Time - the time the event was triggered. The timestamp of the record may be used to determine the resulting duration.
* Type - determined from the attack definitions
* Description - textual notes
* Scope - refers to the status of started, ended or ongoing
* SOS - this allows for a signaling component to simply communicate that further filtering by additional infrastructure, or provider is necessary. This negates the need to perform additional analytics on traffic characteristics and load. This field should be ignored where the scope identifies an attack as having ended. The SOS field is expected to communicate whenever the signaling component is overwhelmed but in certain circumstances this may need to be set for any or all events, it should therefore not be exclusively tied to signaling components health.
* Thresholds - load_factor1 % of max, load_factor2 % of max

The above event attributes will be augmented by additional data relating to the resource being attacked and the current handling.
The protected object attributes will appear:

- Access Token
- Key
- Label
- IPv/Prefix
  - version
  - address/prefix
  - protocol
  - port(s)
- SLA/QoS
- Mitigation status
- B/W threshold
- Counters
  - CurrentPps
  - CurrentBps
  - PeakPps
  - PeakBps
  - TypicalPps
  - TypicalBps

The Access Token and Key elements correspond to those found in the event notifier. Timestamp may be derived from the export-timestamp in the IPFIX header.

* Label - textual label
* IPv/Prefix - identifies the protected object under attack including ip version, address, protocol, port
* SLA - expressed as the first three bits of the dscp field value. This will map to (lowest to highest) BE, CS1, CS2, CS3, CS4, and CS5. The purpose is for the upstream element or provider to be able to classify and handle attack traffic accordingly.
* Mitigation status - simple true or false to denote whether an active mitigation is occurring
* B/W threshold - event bandwidth as a % of overall capacity
* Rate/frequency - exports counters based upon current, peak and average bps/pps

The attack type identifier will be constructed from a category and a sub element. The category will be one of the high level types below with the sub element providing greater granularity into the event. This specific set of identifiers may be further expanded and a mechanism to update the attack dictionary across the JSON API channel or alternately for the elements to negotiate a standard set of definitions or an expanded set should be considered for a future iteration.
2.1 Upstream Feedback

In the event that attack traffic is offramped to an upstream element it is desirable that attack status information is relayed back to the originating device. In intra-domain applications this is essential in order for the operator to fully understand the extent of the ongoing mitigation. This may be done in the form of a return data set comprising the event Key and the CurrentPps and CurrentBps of the attack. This may potentially be extended to include additional information such as onramp clean traffic.

The feedback attributes will appear:

- Access Token
- Key
- CurrentBps
- CurrentPps

The Access Token and Key elements correspond to those found in the event notifier. As with other data sets the timestamp may derived from the export-timestamp contained in the IPFIX header.

* CurrentBps - the current level of offramped traffic in Bps
* CurrentPps - the current level of offramped traffic in Pps

3 Attack/threat categories and sub elements

- Bandwidth - b/w exceeds available capacity or threshold
- Packet Rate - pps exceeds capacity or threshold
- Ipv4 Object - may be one or a combination of the following:
  - addr
  - protocol
  - src port
  - dscp
  - length
  - flags
  - ttl
  - martian
- Ipv6 Object - may be on or a combination of the following:
  - addr
  - protocol/next-header
  - src port
  - length
  - traffic class
  - hop limit
  - flow label
  - martian
- Packet Sanity - packets that fail basic sanity checks:
  - UDP packets with invalid UDP length
- TCP packets with corrupt header
- UDP/TCP with src/dst port 0
- invalid version
- invalid option
- runt/giant/ping of death
- land
- fragments
- TCP - attacks against TCP:
  - syn abuse
  - ack abuse
  - fin abuse
  - rst abuse
  - psh abuse
  - urg abuse
  - window abuse
  - invalid TCP flags (null, xmas)
  - fragment abuse
  - invalid option
  - sockstress
- UDP - attacks against UDP:
  - flood abuse
  - fragment abuse
  - 0 payload
- ICMP - attacks against icmp:
  - flood
- Application - higher layer attacks:
  - hash collision
  - http
    - get flood
    - post flood
    - random/invalid url
    - slowloris
    - slow read
    - r-u-dead-yet (rudy)
    - url regex
    - malformed request
    - xss
  - https
    - ssl session exhaustion
- dns
  - request spoofing
  - query flood
  - nxdomain flood
  - any flood
  - query regex
  - malformed query
  - response flood
  - dnssec abuse
- sip
  - malformed request
- sql
  - injection
- smtp
  - backscatter
  - abuse
- Amplification - amplified/amplifier attacks
  - dns
  - ntp
  - snmp
  - netbios
  - sadp
  - chargen
  - qotd
  - bittorrent
  - kad
  - smurf
  - quake
  - steam
- Intrusion - potential intrusion or nuisance
  - port scan
  - buffer overflow
  - well know threat identifiers (CERT, emerging threats etc.)
- Custom - used for arbitrary definitions
  - custom1
  - custom2
  - etc.

An event will be triggered based on the attack profile. E.g. application:http-slowloris and icmp:flood would be considered 2x separate events. The ability to roll individual events into a parent event id is also permissible. In these instances the ability to identify a parent event would be necessary. A device may use a threat data field in the export to communicate a sample payload for scrutiny by an upstream system or provider and on which a signature based filter may be based.
4 Threat Enumeration

Threats will be identified using a 16 bit format split into 2x octets, the 1st octet will identify the category where the 2nd octet will relate to a specific sub type.

<table>
<thead>
<tr>
<th>Category</th>
<th>ID</th>
<th>SubType</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Packet Rate</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPv4 Object</td>
<td>11</td>
<td>addr</td>
<td>0b1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>protocol</td>
<td>0b10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>port</td>
<td>0b11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>src port</td>
<td>0b100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dscp</td>
<td>0b101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length</td>
<td>0b110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags</td>
<td>0b111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ttl</td>
<td>0b1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>martian</td>
<td>0b1001</td>
</tr>
<tr>
<td>IPv6 Object</td>
<td>100</td>
<td>addr</td>
<td>0b1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>protocol/nh</td>
<td>0b11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>src port</td>
<td>0b100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length</td>
<td>0b101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>traffic class</td>
<td>0b110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hop limit</td>
<td>0b111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flow label</td>
<td>0b1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>martian</td>
<td>0b1001</td>
</tr>
<tr>
<td>Packet Sanity</td>
<td>101</td>
<td>UDP length</td>
<td>0b1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP corrupt header</td>
<td>0b10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UDP/TCP src port 0</td>
<td>0b11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid version</td>
<td>0b100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid option</td>
<td>0b101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>runt/giant/ping of death</td>
<td>0b110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>land</td>
<td>0b111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fragments</td>
<td>0b1000</td>
</tr>
<tr>
<td>TCP</td>
<td>110</td>
<td>syn abuse</td>
<td>0b1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ack abuse</td>
<td>0b10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fin abuse</td>
<td>0b11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rst abuse</td>
<td>0b100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>psh abuse</td>
<td>0b101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>urg abuse</td>
<td>0b111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>window abuse</td>
<td>0b1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid TCP flags</td>
<td>0b1001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fragment abuse</td>
<td>0b1010</td>
</tr>
<tr>
<td>Layer</td>
<td>Option</td>
<td>Bit Representation</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>--------------------</td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>111</td>
<td>invalid option</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sockstress</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>flood abuse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fragment abuse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 payload</td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>1000</td>
<td>flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fragment abuse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 payload</td>
<td></td>
</tr>
<tr>
<td>Application</td>
<td>1001</td>
<td>hash collision</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - get flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - post flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - random/invalid url</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - slowloris</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - slow read</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - r-u-dead-yet (rudy)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - malformed request</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>http - xss</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>https - ssl session exhaustion</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - request spoofing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - query flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - nxdomain flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - query regex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - malformed query</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - response flood</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dns - dnssec abuse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sip - malformed request</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sql - injection</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>smtp - backscatter</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>smtp - abuse</td>
<td></td>
</tr>
<tr>
<td>Amplification</td>
<td>1010</td>
<td>dns</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ntp</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>snmp</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>netbios</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ssdp</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>chargen</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>qotd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bittorrent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>kad</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>smurf</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>quake</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>steam</td>
<td></td>
</tr>
<tr>
<td>Intrusion</td>
<td>1011</td>
<td>port scan</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>buffer overflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>well known - emerging threats</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>well known - us-cert</td>
<td></td>
</tr>
</tbody>
</table>
The JSON API channel is expected to be opened at regular intervals for the exchange of command and control data. The signaling component will authenticate using a standard user/role:password or api-key and request URL:{scheme}://{host}:{port}/dots/api/info using a POST method with a request body of:

```json
{"device_ip":"<device ip>", "device_load_config":{
  "load_factor1": ["<alias>", "% of max"],
  "load_factor2": ["<alias>", "% of max"]
}}
```

The upstream element will use the access token plus ip address to verify the originators credentials as valid signaling component. The upstream element may then pass to the requesting component the IPFIX ID token, the IPFIX destination address, white lists and mitigation information.

METHOD:POST - URL:{scheme}://{host}:{port}/dots/api/info

Request Body:

```json
{
  "device_ip": "<device ip>",
  "device_load_config": {
    "load_factor1": [
      "<alias>",
      "% of max"
    ],
    "load_factor2": [
      "<alias>",
      "% of max"
    ]
  }
}
```

Teague                  Expires January 6, 2016                
[Page 11]
Response Body:

```json
{
    "access_token":"<Access-Token>",
    "export_host":"<ip>",
    "whitelist_ips": [
        "<ip1>",
        "<ip2>",
        ...
    ],
    "mitigation": {
        "status": "<status(Inactive,Monitoring,Mitigating)>",
        "swing_flag": "<true or false>",
        "blacklistaddrs": [
            "<ip1>",
            "<ip2>",
            ...
        ]
    },
    "custom": "arbitrary data"
}
```

Request Body:

The `device_ip` attribute simply details the signaling components source address.

The `device_load_config` is an extensible object that allows the device to communicate aliases and thresholds associated to load factors of interest e.g. CPU, memory, state table etc. The threshold may be used to communicate a level at which the element may be expected to trigger an SOS=true event if exceeded.

Response Body:

The access_token will be used for basic authentication of IPFIX exports to the upstream collector.

The export_host will communicate the ipv4/ipv6 addr of the upstream IPFIX collector.

The whitelist_ips attribute will allow for a provider instance to white list certain ip addresses from which all traffic should be accepted to ensure that any proxied traffic where the original address is obscured is not mistaken for a new attack signature.

The mitigation object is also extensible and will communicate the current status, offramp/restoration status and any relevant black
list information. The status attribute of the mitigation_info object as 3x states:

* Inactive - no IPFIX messages have been received in the last health-refresh-timeout period.
* Monitoring - IPFIX messages are being received
* Mitigating - the upstream is actively mitigating a threat

The swing attribute of the mitigation_info object is set either true or false:

* True - the attack has abated or reduced (if volumetric) to a level deemed within the capacity of the original signaling component. *
* False - the attack mitigation should continue to be handled by the upstream element.

The blacklistaddrs attribute is a simple set of ipv4 or ipv6 addresses and allows an upstream element to communicate known bad actors or compromised hosts to the signaling component.

The custom field may be used for the upstream element to communicate an arbitrary object. This could include a service provider portal url or some other yet to be standardised data.

5.1 JSON API example interaction

```
+-----+                +-----+
| D   | HTTPS            | C   |
+-----+                +-----+
|      |                  |     |
+-----+                +-----+
|      | IPFIX            | I   |
+-----+

D = DDoS mitigation device 192.0.2.1
C = Service provider 198.51.100.1
I = IPFIX receiver 203.0.113.1

D initiates an https connection to C:
URL: https://user:password@198.51.100.1:443/dots/api/info
D posts:
{
  "device_ip":"192.0.2.1",
  "device_load_config": {
    "load_factor1": [
      "cpu",
      "85"
    ],
    "load_factor2": [
      "mem",
      "85"
    ],
    "load_factor3": [
      "bandwidth",
      "75"
    ]
  }
}

C responds:
{
  "access_token":"abc123",
  "export_host":"203.0.113.1",
  "whitelist_ips": [
    "203.0.113.254",
    "203.0.113.253"
  ],
  "mitigation": {
    "status":"Inactive",
    "swing_flag":"True",
    "blacklistaddrs":[]
  },
  "custom": {"portal_url":"https://portal.ddoss.net/Mitige?=192.0.2.1"}
}

Upon receipt of the response body the device 192.0.2.1 would now send event exports to the IPFIX collector at 203.0.113.1 and would authenticate using the ID "abc123". Periodically a new token may be exchanged or an alternate IPFIX destination (export_host) set. In these instances the signaling component should start using the new credentials or destination immediately. The component will whitelist the ip addresses of 203.0.113.254 and 203.0.113.253. The SOS flag will be set to true should the component cpu=85% or mem=85% or bandwidth=75%.
6 IPFIX export

IPFIX was selected for this channel due to its push nature, extensible templates and its existing availability on ddos and security platforms. Leveraging an existing protocol will result in minimal retooling and hopefully lower any barrier to adoption.

An attack will trigger the creation of an incident record on the component which in turn will trigger IPFIX export to an upstream device or provider with details of the attack parameters. Due to the unreliable nature of UDP event data sets will repeat at regular intervals for the duration of the attack.

An attack may generate different data exports which will communicate various facets of the threat, the target and the overall incident. The event data set will define the base key and this will be used to link other records such as protected objects and threat profile data sets. Corresponding data sets referencing the same key will be considered part of the same event when combined with the component id.

An IPFIX event data export may be used as a heartbeat between elements. It is recommended that the signaling component periodically send heartbeats upstream to verify its status during periods of relative inactivity, failure by the upstream to receive these heartbeats may then trigger an alert or further investigation into why they never reached their destination.

6.1 Event Template

The template for events will contain 8x fields as detailed:

```
+----------------+-------------------+
| Set ID = 2     | Length            |
+----------------+-------------------+
| Template ID n  | Field Count = 8   |
+----------------+-------------------+
| Access Token   | Field Length = n  |
+----------------+-------------------+
| Key            | Field Length = n  |
+----------------+-------------------+
| Time           | Field Length = n  |
+----------------+-------------------+
| Type           | Field Length = n  |
+----------------+-------------------+
| Description    | Field Length = n  |
+----------------+-------------------+
| Scope          | Field Length = n  |
+----------------+-------------------+
```

Teague                  Expires January 6, 2016                
[Page 15]
6.2 Protected Object Template

The template for protected object will contain 16x fields as detailed:

| 1 | Set ID = 2 | Length |
| 1 | Template ID n | Field Count = 16 |
| 1 | Access Token | Field Length = n |
| 1 | Key | Field Length = n |
| 1 | Label | Field Length = n |
| 1 | IP version | Field Length = n |
| 1 | Address/Prefix | Field Length = n |
| 1 | Protocol | Field Length = n |
| 1 | Port | Field Length = n |
| 1 | SLA Code Point | Field Length = n |
| 1 | Mitigation Status | Field Length = n |
| 1 | B/W Threshold | Field Length = n |
| 1 | Current Pps | Field Length = n |
| 1 | Current Bps | Field Length = n |
| 1 | Peak Pps | Field Length = n |
| 1 | Peak Bps | Field Length = n |
| 1 | Typical Pps | Field Length = n |
| 1 | Typical Bps | Field Length = n |
6.3 Attack and Threat Identification Template

The template for attack and threat identification will contain 4x fields as detailed:

<table>
<thead>
<tr>
<th>Set ID = 2</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template ID n</td>
<td>Field Count = 4</td>
</tr>
<tr>
<td>1</td>
<td>Access Token</td>
</tr>
<tr>
<td>1</td>
<td>Key</td>
</tr>
<tr>
<td>1</td>
<td>Threat Identifier</td>
</tr>
<tr>
<td>1</td>
<td>Threat Data</td>
</tr>
</tbody>
</table>

Note - where no threat data is required to aid in mitigation (ie the identifier is enough) the Threat Data field may be set to null.

6.4 Feedback Template

The template for feedback will contain 4x fields as detailed:

<table>
<thead>
<tr>
<th>Set ID = 2</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template ID n</td>
<td>Field Count = 4</td>
</tr>
<tr>
<td>1</td>
<td>Access Token</td>
</tr>
<tr>
<td>1</td>
<td>Key</td>
</tr>
<tr>
<td>1</td>
<td>CurrentBps</td>
</tr>
<tr>
<td>1</td>
<td>CurrentPps</td>
</tr>
</tbody>
</table>
7 Security Considerations

The protocol described here serves as a security mitigation tool. Potential vulnerabilities of this system are addressed by the use of encrypted channels for communication between the elements and the use of low overhead control signals in case there is denial of service or congestion affecting the paths between the elements. The security considerations of [RFC7011], [RFC5405] and [RFC4960] apply to the IPFIX, UDP and SCTP based channels respectively. Additional security considerations will be added to subsequent drafts.

8 IANA Considerations

There may be requests to IANA in order to update the registry of IPFIX entities.

9 Contributors

The initial version of this document represented a collaborative effort by engineers at Verisign and Juniper to create a candidate for an open standards effort supporting communication between on-premise DDoS mitigation devices and provider based DDoS mitigation services. A standards based approach allows businesses to have a wider range of options to better secure their complex environments without the limitation of vendor lock-in. The companies published a draft specification through the Internet Engineering Task Force (IETF) to encourage community participation and further development of these proposals toward becoming an open standard.

10 Acknowledgements

The following people are acknowledged for their technical contributions in the development of this document: Aziz Mohaisen, Jon Shallow, Suresh Bhogavilli, Jeshmi Raman, Malathy Poruran, Roman Danyliw, Andrew Mortensen, Rich Groves, Scott Barvick, Franck Martin, Brian Trammel.
11 References

11.1 Normative References


12 Changelog

Changes between 00 and 01:
* Cleaned up language to remove 'cloud' and 'cpe' terminology for clarity
* Added feedback element and described same
* Called out SCTP as possible transport for IPFIX
* Adjusted thresholds to be informed by the originating device
* Tidied JSON and other representations
* Added smtp ddos to the data dictionary

Authors’ Addresses

Nik Teague
Verisign Inc.
12061 Bluemont Way
Reston, VA 20190
US

EMail: nteague@verisign.com
The Extended DDoS Open Threat Signaling Use Cases
draft-xia-dots-extended-use-cases-00.txt

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), 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/ietf/1id-abstracts.txt

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

This Internet-Draft will expire on December 27, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract
This draft proposes two extended use cases which illustrate more scenarios and multiple ways of implementation within the existing DOTS work scope. One is the data mining and SDN based centralized Anti-DDoS use case, the other is the NFV based distributed DDoS mitigation use case.

Table of Contents

1. Introduction ................................................ 2
   1.1. Background ............................................. 2
2. Conventions used in this document ........................... 4
3. Data Mining and SDN Based Centralized DDoS Protection ...... 5
4. NFV Based Distributed DDoS Mitigation Use Case .............. 7
5. Security Considerations ..................................... 9
6. IANA Considerations ......................................... 9
7. References .................................................. 9
   7.1. Normative References ................................... 9
   7.2. Informative References ................................. 9
8. Acknowledgments ............................................. 9

1. Introduction

DDoS attacks are one of the largest threats to the Internet, and are evolving very quickly whatever its volume size or complexity. The DDoS attack victims include ISPs, enterprises, and websites. To defend their network resource or services against DDoS attack, Anti-DDoS solutions are needed. According to specific scenarios or requirements, as well as the emerging new technologies such as cloud, NFV and big data, various Anti-DDoS solutions exist in current industry.

This document will present two use cases for a distributed Anti-DDoS solution based on standard inter-system communications between the components. These standards will permit a mix of "best of breed" deployment.

1.1. Background

Current Anti-DDoS solution is to deploy a proprietary Anti-DDoS system close to the protected site, or in the network, close to the protected site. Anti-DDoS systems can be either one physical box or a distributed system. The former application means that the detection and mitigation modules are all located in the same box. In comparison, the latter is a distributed system which includes distributed devices responsible for detection (i.e., DPI), mitigation (i.e., scrubbing) and central control respectively. The
latter application is better in overall performance and deployment flexibility. To meet the various requirements, the Anti-DDoS system is deployed in various locations in a network. For example, it is deployed near the protected sites for easily detecting application-layer attacks, or near to the attack source to mitigate attacking traffic as soon as possible and prevent them flooding into the network.

Due to the challenges of high volume and complexity brought by today’s DDoS attacks, the cloud-based Anti-DDoS service is becoming attractive and adopted by more and more customers. By this way, all of the customer’s traffic is monitored and scrubbed by the Anti-DDoS service provider in real time, and the customer can manage its own Anti-DDoS service and get the related information through the web-based customer portal. This type of service has the benefits of high performance and scalability.

On the other hand, Network Function Virtualization (NFV) is considered as a promising technology used by network operators for its great benefits such as saving cost and speeding up new service’s provision. Specifically, for the Anti-DDoS service provided by network operators, they can dynamically create the Anti-DDoS Virtual Network Functions (VNFs) and deploy them to the appropriate locations in the network (i.e., near to the attack source or destination, or both) as needed, because they have the information and control of the whole network. The network operators have the inherent advantage comparing with the third-party Anti-DDoS service providers in this aspect.

Furthermore, in addition to the detection by specific devices (e.g., Deep Packet Inspection (DPI)), normal network forwarding devices (e.g., router or switch) can also be involved in the DDoS attack detection by collecting the L3/L4 flow information and sending them to the centralized platform for analysis or data mining. It can be a complimentary way to current DDoS detection mechanism, or an independent detection method by itself.

During the last few years, the above technologies are in the process of integration, aiming to develop a comprehensive distributed and collaborative Anti-DDoS solution. One example is the hybrid solution by combining the specified on-premise Anti-DDoS devices with cloud-based Anti-DDoS service. The on-premise devices monitor all the traffic of customer and effectively mitigate the application-layer attacks. When attack size reaches customer-established thresholds, mitigation can be moved to the cloud platform. The ultimate goal of the integration is forming a full spectrum of Layer 3-7 defenses both on-premise and in the cloud. For all the distributed and
collaborative Anti-DDoS solutions, the coordination among all the member elements is necessary for managing them, as well as collecting and correlating various information from them so as to form a holistic network security view.

[I-D.draft-mglt-dots-use-cases] describes several DDoS Open Threat Signaling (DOTS) use cases for communication across distributed Anti-DDoS devices or between on-premise device and cloud platform. Additionally, it also illustrates the benefits the DOTS work can bring.

This draft proposes two new use cases which illustrate more scenarios and multiple ways of implementation within the existing DOTS work scope:

- Collect and correlate security related flow information from network forwarding devices and proactively detect the DDoS attack by centralized analysis or data mining;

- Dynamic and distributed Anti-DDoS solution by creating VNFs and deploying them to the edge network on demand.

2. Conventions used in this document

- DDoS - Distributed Denial of Service
- DOTS - DDos Open Threat Signaling
- SDN - Software Defined Network
- NFV - Network Function Virtualization
- DPI - Deep Packet Inspection
- CAPEX - Capital Expenditure
- IPFIX - IP Flow Information Export
- ACL - Access Control List
- PoP - Point of Presence
3. Data Mining and SDN Based Centralized DDoS Protection

With the development of big data and SDN/NFV technologies, new ways of thinking of DDoS protection come along as well. A centralized data mining and SDN-like control platform plays a key role for DDoS protection in this use case.

The centralized platform collects L3/L4 flow information from normal network forwarding devices (e.g., router or switch) in the whole network, and then analyzes them with data mining technology to get the holistic view of network DDoS threats leading to an easy DDoS attack detection. Compared with traditional signature based solution, data mining analysis focuses more on the behaviors and patterns of the data flows other than the content of the packets. Multi-dimension to ultra-high dimension models can be built to accurately profile the data flows on-line, which allows detecting and even predicting DDoS attacks in real-time. By this way, operators can greatly reduce the Capital Expenditure (CAPEX), as complicated and expensive detecting devices with Deep Packet Inspection (DPI) functions will be no longer essential. Furthermore, in contrast to dedicated Anti-DDoS devices, the data mining platform is highly scalable without obvious performance limit (the data mining functions can be executed on the elastic computing environment). And it has self-adapting capability to proactively detect new mutations of DDoS attacks.

This Anti-DDoS solution involves a large number of elements, i.e., routers, switches, data mining platform, dedicated Anti-DDoS devices, and etc, as well as frequent information exchange between them to fulfill its essential functions, i.e., packet/flow sampling, traffic diversion, sending security policies, and etc. All these elements and related control processes can be integrated into the SDN-like control architecture to improve the automation level so as to reduce operational involvement in DDoS attack management.
Figure 1. Data Mining and SDN Based Centralized Anti-DDoS Use Case

As illustrated in Figure 1, a data mining and SDN based centralized Anti-DDoS solution forms a closed-loop control system which includes the following steps:

1. Data mining platform monitors network traffics by big data analysis algorithms based on received IP Flow Information Export (IPFIX) packet sampling records, and it probably needs some extensions to current IPFIX specification for security requirements [I-D.draft-fu-ipfix-network-security].

2. Data mining platform sends the monitoring report to the SDN controller, which provides the inputs for SDN controller to take next step actions. The report contains the information about the detected DDoS attacks based on the data mining models taken by the platform, the information could be the abnormal flows, the suspicious DDoS attack sources or destinations.

3. Based on the monitoring reports input, the SDN controller can control the network forwarding devices to perform various operations, e.g., adjusting the IPFIX flow sampling policies, or configuring device security policies such as rate-limiting or Access Control List (ACL), or traffic redirection to specified mitigation devices or tracking the attack sources and etc.
4. The suspicious traffic is identified and redirected to specified Anti-DDoS devices for further inspection and cleaning, and then clean traffic is transmit back to the network;

5. At last, the DPI and scrubbing statistics information created by the specified Anti-DDoS devices are reported to the data mining platform, which are used to help it to improve and derive further security intelligence by self-learning mechanism.

4. NFV Based Distributed DDoS Mitigation Use Case

Previously, due to the deployment limit of physical DDoS mitigation devices and the third-party Anti-DDoS service provider does not have the control of the network infrastructure, the centralized deployment of DDoS mitigation devices is more suitable than the distributed deployment. The centralized way is not optimized in saving network bandwidth, and is possible to make DDoS mitigation devices to be the bottleneck.

Now, the distributed deployment of DDoS mitigation appliances to the network edge is becoming feasible as NFV technologies grows quickly and are widely adopted by network operators for managing network infrastructure. By the way of dynamic deployment, the virtual DDoS mitigation appliances (i.e., virtual FW, scrubbing center, etc) are distributed at the network edges to relieve the performance and network bandwidth consuming problems.

Generally, for the distributed Anti-DDoS solution, the DDoS monitoring appliances should be closer to the attacked destination for easy detection, and the DDoS mitigation appliances should be closer to the attacking sources for saving network bandwidth. So, the source tracking mechanism is an important part of the whole solution.
Figure 2. NFV Based Distributed DDoS Mitigation Use Case

Figure 2 illustrates the use case including the following steps:

1. DDoS monitoring appliance sends the monitoring report to the Anti-DDoS controller, providing the inputs for next step actions;

2. Anti-DDoS controller performs the attacking source tracing mechanism to locate the network edges (i.e., PoPs) needed to deploy the virtual DDoS mitigation appliances;

3. ISP’s NFV orchestration center dynamically deploys the virtual DDoS mitigation appliances on the network edge to filter/clean the attacking traffic.
5. Security Considerations

This specification talks about the use cases for anti-DDoS solutions, which does not introduce any new security threats to the network. However, if the anti-DDoS system could be hacked by attackers, then it could be used for malicious purposes, such as protecting the attacks, or generating new attacks.

6. IANA Considerations

There is no IANA consideration for this specification.

7. References

7.1. Normative References


7.2. Informative References

[I-D.draft-mglt-dots-use-cases] Migault, D., "DDos Open Threat Signaling use cases", work in progress, April 2015.


8. Acknowledgments

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

Liang Xia (Frank)
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: Frank.xialiang@huawei.com

Haibin Song
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: haibin.song@huawei.com