Transparent Packet Architecture
Future Infrastructure Group
September 2019

CableLabs®

Transparent Security Architecture

Prepared by
Randy Levensalor, Lead Architect | r.levensalor@cablelabs.com
Dan Schrimpsher Ph.D., Lead Engineer | d.schrimpsher@cablelabs.com

Executive Summary

Distributed denial of service (DDoS) attacks and other Internet-based attacks cost members billions of dollars, and the number and scale of the various threats continue to grow yearly. Internet of things devices, a variety of new devices subject to attack, and readily available upstream bandwidth make it easier to create increasingly impactful attacks. Transparent Security uses programmable data plane capabilities to enable real-time packet processing, high-resolution packet inspection, and in-band telemetry. In-band telemetry allows MSOs to identify the compromised devices in a fraction of a second instead of several minutes. This technology is enabled by new, currently available programmable chips that can process packets in real time and be deployed at any point in the network, from the core to the head-end to residential and business customer premises. It has numerous opportunities for future innovations as it is flexible, scalable, and compatible with cable infrastructure while also being inexpensive to deploy.

Contents

1 Introduction

Distributed denial of service (DDoS) attacks and other cyberattacks cost operators billions of dollars, and the impact of these attacks continues to grow in size and scale, with some exceeding 1 Tbps. The number of Internet of things (IoT) devices continues to grow rapidly, many have poor security, and upstream bandwidth is ever increasing—this perfect storm has led to exponential increases in IoT attacks, by over 600% between 2016 and 2017 alone.[1] With an estimated increase in the number of IoT devices from 5 billion in 2016 to over 20 billion in 2020,[2] we can expect the number of attacks to continue this upward trend.

Detecting attacks is difficult, but mitigating them is even harder, and a number of solutions have been proposed with varying degrees of success. Typically, these solutions focus on the target of the attack rather than the source. However, even if the target is completely protected, an operator's access networks can still be seriously affected, resulting in connectivity loss and quality of service (QoS) issues for customers.

Enhanced device visibility and packet processing can be used to identify the sources of such attacks. Leveraging programmable ASICs and the P4 applications provides support for these enhancements by making the behavior of the data plane expressible in software and customizable without affecting performance. Specifically, this project will pair a DOCSIS modem with a series of P4-enabled devices connecting back to the operator headend via the P4 Runtime. This architecture allows for visibility throughout the access network.

Transparent Security leverages a machine-learning controller that has been trained with patterns to identify conditions and to perform dynamic operations by deploying new packet processing behaviors in the network (e.g., DDoS mitigation, virtual firewall, QoS detection/enforcement, and DOCSIS data plane functions). All operations will be performed at line rate while leveraging P4 in-band network telemetry (INT), which allows data to be collected for reporting and analysis without control plane intervention. By inserting telemetry into the packet header, telemetry can be added to all packets rather than simply a sampling, which significantly reduces the time required to identify and mitigate the attack.

2 Background

Most proposed DDoS solutions fall into one of two camps, detection or mitigation. This section discusses additional possible solutions in those categories and examines their advantages and disadvantages.

2.1 Packet Collection Methods

2.1.1 Sampling

The sampling technique cannot offer an operator a view of the entire network. Sampling is a statistics-based method for measuring network traffic and collecting, storing, and analyzing a sample of traffic data. This method has the advantage of allowing many interfaces to be monitored without significantly affecting network traffic. Packet sampling, inspecting a small percentage of traffic, can provide sample data with a relatively low overhead. Frequently, only 1 in 5,000 packets are inspected when attempting to detect a DDoS attack. Each sample is sent out-of-band as a duplicate packet across the network from every network device that provides the sampling data, which generates additional traffic.

SFlow[3] and NetFlow[4] are two protocols used for sampling data. SFlow is a sampling protocol that does not perform any operations of the sample data before sending it to the collector. However, it records only the original packet and the routing table at the time of the sample, making it difficult to directly correlate device state with packet information. NetFlow performs some manipulation on the packets

[1] "Internet Security Threat Report, Volume 23," March 2018, Symantec
[2] "Leading the IoT," 2017, Gartner—M. Hung, ed.
[3] "Traffic Monitoring Using sFlow," 2003, sFlow.org
[4] "NetFlow Generation: The Security Value Proposition," November 2015, Gigamon

before sending them. As a result, NetFlow is very taxing on the system, which means that frequent sampling can have an adverse impact on latency and network throughput.

Although the sampling method can effectively identify larger trends, it can miss anomalies that occur in a small portion of the traffic. That is, it can detect a DDoS attack at the target, but it can miss or take a long time to detect a DDoS attack at the source. With source-based DDoS attacks, the system is trying to identify smaller anomalies, which are unlikely to be captured in the sample.[4]

2.1.2 In-Band Network Telemetry

In-band network telemetry (INT) is a method for adding telemetry data to every packet at multiple points on the network. This approach can help determine things such as the path of a packet or the source device emitting packets. With INT, the packet only needs to be inspected at the edge of the network, which reduces the overhead and generates less traffic compared with sampling at multiple points in the network.

In an attempt to standardize INT, the P4.INT specification[5] suggests a set of predefined fields:

- switch ID,
- control plane version,
- ingress/egress port,
- timestamp,
- RX packet count, and
- congestion status.

By leveraging P4 to implement INT, the data can be customized to meet the needs of a service provider. With this data, one can identify the source of a packet behind a firewall by including the originating MAC and IP address in the INT header. With these data, one can obtain the specific source of a packet behind a firewall. Because customer premises and wireless networks are dynamic and multiple devices share the same ingress point on the gateway, knowing the ingress port is not sufficient when trying to identify the packet source. These extensions to the standard P4 INT data structure provide the packet source's MAC and IP addresses, which are unique and are required to identify specific devices for customer premises and wireless networks.

One potential issue with INT is that it increases the size of the packet header. If this increase exceeds the frame size, it will cause packet fragmentation, which has a negative impact on network performance. This issue should not arise with Transparent Security, however, because the INT data are added at the gateway. With access networks, such as DOCSIS, the frame size is larger between the gateway and the core network than at the customer premises.

2.2 DDoS Attack Mitigation Methods

There is a wide variety of methods for mitigating DDoS attacks. A few of the common methods and the methods used by Transparent Security are highlighted below. This list is not exhaustive.

Also examined in this section are the points on the network where the mitigation is performed.

2.2.1 Types of Mitigation

2.2.1.1 Scrubber

As the name suggests, a scrubber cleans traffic to a specific host that is under attack. Whether the host under attack is an appliance or service, when the attack is detected, the traffic to that host is analyzed—malicious packets are removed, and clean packets are forwarded. Different types of scrubbers have issues with either high-volume attacks or low-volume attacks.[6] See Section 2.2.2.1, "Out-

[5] "In-band Network Telemetry (INT)," June 2016, P4 Language Consortium (C. Kim, P. Bhide, E. Doe, H. Holbrook, A. Ghanwani, D. Daly, M. Hira, and B. Davie)
[6] "Preparing to Withstand a DDoS Attack," October 2015, SANS Institute—G. Pandya

of-Band," for explanations of the weaknesses of out-of-band packet scrubbing. Scrubbers also require additional hardware in the network, which adds capital and operational costs for the service provider.

2.2.1.2 Switch with OpenFlow

OpenFlow 1.4 supports a number of classifiers that could potentially be used to detect and mitigate network events.[7] However, this approach has three main problems:

- OpenFlow focuses on a single switch at a time.
- Many of the classifiers are optional and require switch support to be used.
- A limited number of classifiers are available.

Each OpenFlow SDN Controller would have to convert the DDoS mitigation pattern to a best-effort match based on the attributes available for pattern matching on the switch. This best-effort match could result in false positives and the dropping of traffic that is not related to a DDoS attack.

2.2.1.3 Switch or Gateway with P4

The features available with networking hardware supporting the P4 language allow for the development of flexible, open, and consistent DDoS mitigation solutions. Using the information gleaned from the INT data, it is possible not only to detect an attack very quickly but also to identify the compromised device. Once identified, the switch or gateway can be notified to reroute or drop the problematic packets with a simple match rule performed at line speed when deployed with hardware acceleration in software on a gateway device containing a P4 chiplet. Additional information on P4 can be found at https://p4.org/.

2.2.2 Network Locations for Mitigation

2.2.2.1 Out of Band

Out-of-band DDoS mitigations come in two flavors, appliances and scrubbing services. When an attack is detected against a host in the network, the traffic is routed through the out-of-band device, where it can remove or re-route the malicious traffic.[6] An appliance routes traffic to the target to itself then removes the malicious traffic and provides a clean flow to the target. A service functions similarly except it routes target traffic to a mitigation center that cleans the traffic and forwards it to the destination.[6]

However, both methods have downsides. The appliance is generally costly, deployment can be complex, and it often fails against high-volume persistent attacks. The service has difficulty defending low-bandwidth slow attacks, and every interface must be protected or it can fail to stop some attacks. Both methods can add latency to network traffic during an attack, and they generally rely on other methods for attack detection.[6]

2.2.2.2 In Band

In-band DDoS mitigations work in the network path, comparable to a firewall, allowing the method to see all network traffic and react accordingly. As a result, in-band mitigation can provide detection as well as mitigation. It does, however, add to network complexity and introduces another point of failure in the network.[6] Transparent Security uses the in-band DDoS mitigation method, along with source-based mitigation.

2.2.2.3 At the Source

Network operators can provide DDoS detection and mitigation at the source (typically the edge servers). This method protects target organizations with little to no effort on their part.[6] It can also provide insight into hacked devices on the customer premises if packet visibility is detailed enough. This information can go a long way into preventing future attacks. Mitigation at this location has the added

[7] "OpenFlow Switch Specification," version 1.4.0, October 2013, Open Networking Foundation, p. 51–60

benefit of limiting attack traffic on an operator's network by blocking malicious packets before they enter the core network. Transparent Security mitigates primarily at the source, but the method is compatible with additional in-band target-based mitigation methods.

2.2.2.4 At the Target

Most DDoS mitigation solutions focus on the target. Generally, services or devices are provided at the edge of the target organization's network, which has an advantage of blocking traffic from anywhere and not depending on network providers for "clean pipes."[6] However, it does nothing to solve the problem of hacked devices that are generating the attack and therefore cannot prevent future attacks, which could eventually succeed.

3 Transparent Security Architecture

Transparent Security uses programmable data plane capabilities to enable real-time packet processing, high-resolution packet inspection, and in-band network telemetry (INT). INT allows MSOs to identify the compromised devices in milliseconds rather than minutes. This technology is enabled by new, currently available programmable chips that can process packets at line speed and be deployed at any point in the network, from the core network to residential and business customer premises.

Rather than being limited to an out-of-band sampling of packets, Transparent Security takes advantage of the P4 programing language's ability to inspect every packet and add additional INT data to the packet header for further processing upstream. Transparent Security is focused on inspecting, finding, and blocking malicious packets as close to the source as possible by adding details about the packet's source device, exact route through the network, and travel duration. INT can then be used by upstream processes to identify traffic patterns and then act on that information.

3.1 Data Plane Architecture

Figure 1 shows a typical data plane architecture of customer premises connected to the access network into an operator's core network. Any combination of the customer gateway, aggregate switch, or core switch can be P4 enabled or not. By enabling P4, the available use cases increase substantially, but this architecture can be implemented in stages.

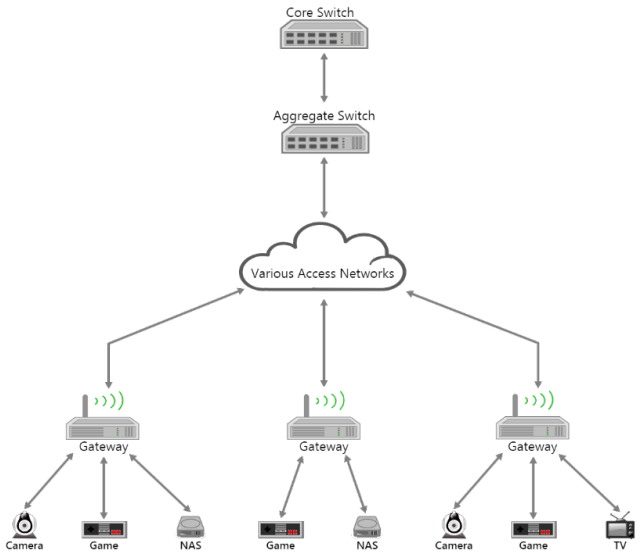

Figure 1 - General Architecture for the Data Plane in the Transparent Security Model

3.2 Control Plane Architecture

Figure 2 depicts an example control plane architecture in which the analytics engine receives INT data from the core in-band as packets flow through the network. When malicious patterns are detected, the SDN controller is notified and updates the P4-enabled devices to handle the packets based on the pattern signature. The management interface between the controller and the P4-enabled devices can use a variety of protocols, including GRPC, Thrift, HTTP, or RPC. The protocol between the SDN controller and switches can vary, depending on the protocols supported by the switches and gateways. Telemetry data and alert notifications can optionally be sent to a dashboard or NOC server for integration with other analytics.

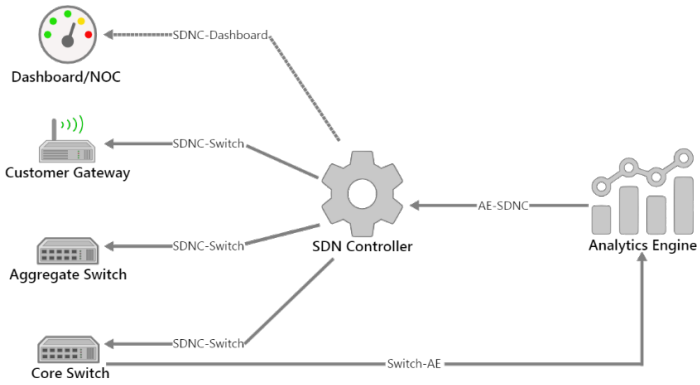

Figure 2 - General Architecture for the Control Plane in the Transparent Security Model

3.3 Demonstration Architecture

Finally, Figure 3 depicts the architecture used in our proof of concept. Three customer premises, each containing two or more devices, are connected to an operator-supplied P4-capable gateway. The gateways, in turn, connect to the access network through a P4-enabled aggregate switch. The aggregate switch then connects to a P4-enabled core switch. The core switch sends traffic to the Internet and to the analytics engine for pattern recognition. The controller initializes each P4-enabled device and updates them when needed to block malicious traffic.

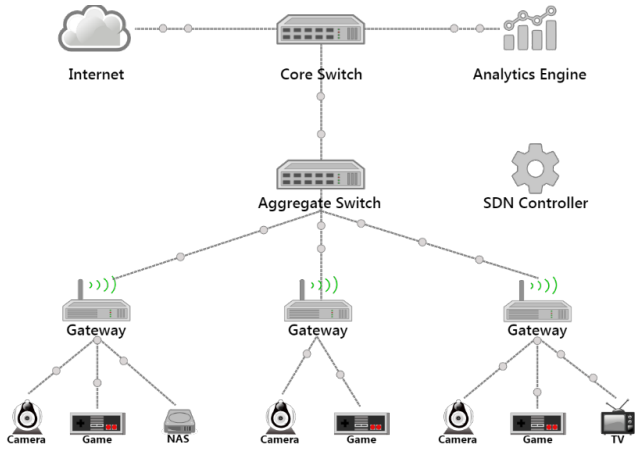

Figure 3 - Architecture Used in the Initial Demonstration of the Transparent Security Concept

4 Components

4.1 Analytics Engine

The analytics engine (AE) serves as the intelligent core of the programmable data plane. Its purpose is to analyze a stream of packets containing P4.INT data and make inferences relative to generally defined patterns. These patterns represent the goals of the system, such as DDoS mitigation, QoS, and proactive network maintenance (PNM). When a pattern is matched, the AE informs the SDN controller, which is responsible for implementing network changes through the control plane.

The AE's functions include the following:

- look at header and INT data,
- determine when there is an attack,
- share the attack signature with the controller to mitigate the attack, and
- notify the controller when the attack has ended.

4.2 Controller

The information used by the AE is captured by the P4 devices as part of the data plane and forwarded to the AE in line with the network traffic. When the SDN controller is informed by the AE that a network goal (DDoS attack, QoS, etc.) has been detected, it updates action tables in the P4 devices through the control plane management interface (using protocols such as GRPC or Thrift). Thus, the P4 devices gain additional abilities in the data plane with a minimally intrusive controller activating them before consequences are experienced.

The controller's functions include the following:

- manage the network configuration on switches and gateways,
- push DDoS mitigation to managed devices,
- remove DDoS mitigation from managed devices, and
- track which devices are participating in an attack through counters of dropped packets based on DDoS mitigation.

4.3 P4-Enabled Data Plane

The switches and customer gateways used in Transparent Security support P4. With P4-enabled devices, the functionality of the devices can easily be changed after they are deployed. P4 simplifies these devices by enabling operators to configure them with only the functionality that is being used. With traditional switches, all of the available functionality is configured when the switch is built, and the manufacturer can make only limited changes through firmware updates.

4.3.1 Core Switch

P4-enabled switches could replace most of the routing rules functionality, with more complex functions hosted by the operator in the cloud.

The core switch's functions include the following:

- manage the traffic between the access and core networks,
- send P4.INT data and packet headers to the analytics engine,
- add P4.INT data (source port, time, queue),
- remove P4.INT headers before they leave the service provider network, and
- mitigate DDoS attacks from the core network and the aggregate network.

4.3.2 Aggregate Switch

P4-enabled switches could replace most of the routing rules functionality, with more complex functions hosted by the operator in the cloud.

The aggregate switch's functions include the following:

- manage the traffic between the gateways and the core switch,
- add P4.INT data,
- forward traffic between gateways to the core switches, and
- mitigate DDoS attacks from the core switches and gateways.

4.3.3 Gateway

Adding a P4-enabled chip to the gateway could offload most of its functionality, which is currently performed on the general-purpose processor in the gateway. More complex functions, such as the user interface, can be hosted by the operator in the cloud. However, the gateway can also be the source of infected traffic, so it should not always be trusted.

The gateway's functions include the following:

- manage the traffic between the customer premises and the access network,
- add P4.INT data, and
- mitigate DDoS attacks from individual devices.

4.4 Customer Premises Equipment

The customer end devices are the typical end units used by end customers, including IoT devices, phones, laptops, and an ever-increasing variety of devices. Many are connected over Wi-Fi or LTE.

For the proof of concept , we use the following types of devices:

- NAS—network attached storage,
- TV—streaming television,
- Camera—IP video streaming camera, and
- Game—game console.

5 Message Flows

The following sections outline high-level message flows between the possible components.

5.1 Initializing and Reporting

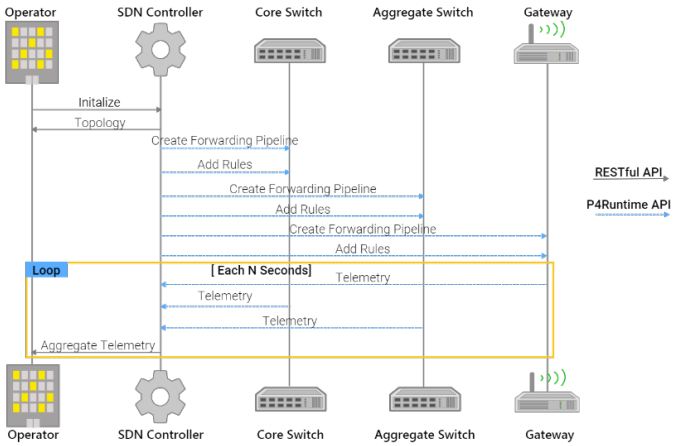

Figure 4 - High-Level Message Flow in the Control Plane at Initialization and During Normal Operation

5.1.1 Switch/Gateway Initialization

The SDN controller initializes all P4-enabled switches and gateways at startup:

- connects to the device through remote interface (through protocols such as GRPC or Thrift);
- sets itself as the master arbitrator;
- transmits the compiled P4 program, tables, actions, and header definitions; and
- transmits the routing rules for each P4-enabled component (i.e., table entries).

The physical format of the messages depends on the protocol used, such as GRPC or Thrift, and the target platform.

5.1.2 Telemetry

At initialization, the controller transmits the network topology to the network operator, including the links between all managed devices.

At a configurable interval, the controller requests counters from each P4-enabled device:

- forwarded packet/bytes count and
- blocked packet/bytes count.

At a configurable interval, the controller transmits updated telemetry data to the network operator.

5.2 Packet Flow and Alerts

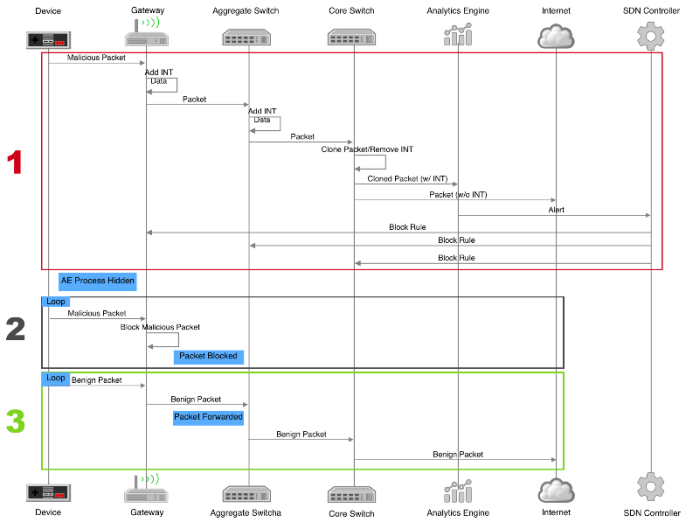

Figure 5 - High-Level Message Flow in the Control Plane and Data Plane when (1) a Malicious Packet Is Discovered, (2) a Malicious Packet Is Blocked, and (3) Benign Packets Are Allowed Through.

5.2.1 In-Band Network Telemetry Packet Usage

INT data are added to the packet header at several hops in the network; the data add packet traceability information and enable the analytics engine to determine the source of the attack even when the IP is spoofed.

INT is manipulated and used as follows:

- the customer device sends packets to the P4 gateway,
- the gateway adds INT headers to packets and forwards to the aggregate switch,
- the aggregate switch updates the INT header and forwards to the core switch,
- the core switch clones the packet,
- the clone with INT header is forwarded to the analytics engine, and
- the INT header is removed from the original packet and forwarded to the destination.

5.2.2 Malicious Pattern Alert

When the analytics engine recognizes a pattern in the packets, it transmits the signature to the controller. An example signature might include the following:

- destination IP address,
- destination port, and
- packet size.

When the controller receives an alert signature, it updates the rules on the P4-enabled devices with the following information:

- Match: signature, and
- Action: block.

5.2.3 Malicious Packet Flow

A malicious packet flow is the flow for the traffic that is participating in the DDoS attack. It contains the packets that will be removed from the network. These flows are typically generated by customer devices and routers.[1] The following actions are taken depending on the compromised element.

Compromised customer device

- Compromised customer device sends malicious packets to the P4 gateway.
- Gateway matches packet signature with block rule.
- Packet is blocked at the gateway.

Compromised gateway

- Gateway sends malicious packets to the aggregate switch.
- Aggregate switch matches packet signature with block rule.
- Packet is blocked at the aggregate switch.

Compromised aggregate switch

- Aggregate switch sends malicious packets to the core switch.
- Core switch matches packet signature with block rule.
- Packet is blocked at the core switch.

Compromised traffic from the core network

- Malicious packets generated outside of the service provider network are sent to the core switch.
- Core switch matches packet signature with block rule.
- Packet is blocked at the core switch.

5.2.4 Benign Packet Flow

A benign flow includes the normal packets that need to be sent through the network as quickly as possible. Benign packets from both compromised and secure devices should be allowed through the network. The following actions are taken for benign packets.

Benign customer device

- Customer device sends benign packets to the P4 gateway.
- Gateway does not match any packet signature with the block rule.
- Packet is forwarded to the aggregate switch.
- Aggregate switch does not match any packet signature with the block rule.

- Packet is forwarded to the core switch.
- Core switch does not match any packet signature with the block rule.
- Packet is cloned and sent to the analytics engine.
- INT data are removed from the packet.
- Packet is forwarded to the Internet.

5.3 Operator Responses

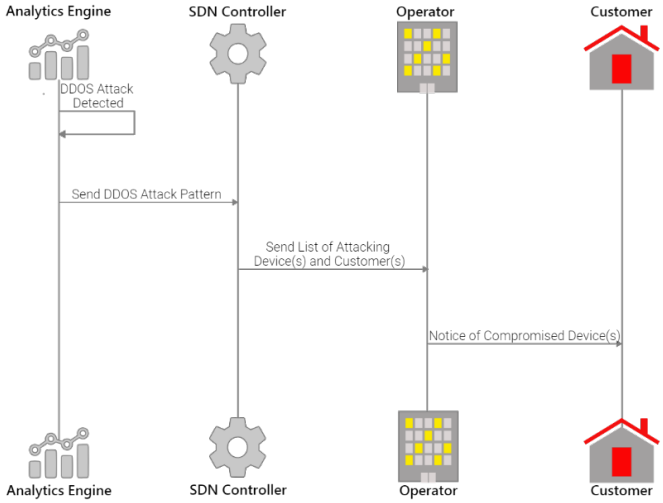

Figure 6 - High-Level Message Flow in the Control Plane when a Malicious Packet Is Discovered—The alert that the customer premises has been compromised is sent to the network operator. The network operator then could communicate with the customer in-band or out-of-band to resolve the issue.

5.3.1 Operator Alert

When the controller receives an alert signature, it alerts the operator with the following information:

- gateway attributes to enable customer identification and notification,
- customer device MAC address,
- destination, and
- type of attack.

5.3.2 Operator Response

The operator can contact the customer through predefined channels to notify them of which devices are compromised and help the customer fix the compromised device.

6 Interfaces Between Components

This section describes the interfaces between components, the data transmitted over these interfaces, and the format used for the proof of concept (PoC). The interfaces (see Figure 3) will need to be defined and updated in order to have interoperability between components. Components include the analytic engine (AE), SDN controller (SDNC), core and aggregate switches, gateway, and dashboard.

6.1 AE-SDNC

This interface defines the management between the analytics engine and the SDN controller. For the PoC, this interface is implemented as a RESTful API, but it could be changed to Protocol Buffers (protobuf) over GRPC to follow the P4 Runtime pattern. A publish/subscribe, or pub/sub, messaging model such as Kafka can be used to scale the solution and remove the burden of tracking the SDN controllers from the AE. Initial setup and deployment information is used to configure the destination for the packet header and INT data and also includes information on how to mitigate a new attack. The PoC currently applies the mitigation to all gateways and switches in the domain managed by the SDN controller. In the future, this function can be extended to support selective application of mitigation to only the impacted switches and gateways.

Potential future extensions to this interface include the following:

- selective application of mitigation to only the impacted switches and gateways,
- modification or removal of existing mitigation rules, and
- collection of out-of-band telemetry data.

6.2 SDNC-Switch/Gateway

This interface connects the SDN controller to the switches and gateways. The PoC supports the same protocol and data model for both switches and gateways. The data model consists of the following:

- signature of the traffic to be blocked,
- configuration and setup information, and
- telemetry data.

The switches support the P4 Runtime standard of protobuf over GRPC. For some of the gateways in the PoC, the device vendor supports an older Thrift interface. With an SDN controller, support is possible for vendors that have not adopted the P4 Runtime API and can leverage Thrift or OpenFlow for the functionality that is exposed. This flexibility in communicating with switches and gateways is inherent to SDN controllers and can be leveraged to lower the barrier to adoption.

6.3 Switch-AE

The data flowing from the core switch to the AE consists of the packet header, with the standard L3–L7 headers as defined by the OSI Networking Model, and the INT data collected while traversing the network.

Note: The entire packet is currently being sent over this interface for the PoC, but it is NOT the long-term solution because only the header information is being inspected by the AE.

6.4 SDNC-Dashboard

This interface is for sending an alert from the SDN controller to a dashboard or other monitoring system. The alert shows which devices are involved in an attack and the traffic associated with that attack. This interface is implemented as a RESTful API for the PoC. For production, it could be changed to a streaming interface such as GRPC, MQTT, Kafka, or similar protocol.

7 Implementation Phases

Transparent Security can be deployed across the service provider network in several phases. A phased deployment makes it easier to deploy and realize many of the benefits while postponing any changes to cable modems (CMs). The process of a phased deployment begins at the hub or head end and moves to the customer premises at later phases.

7.1 Hardware and Software Updates in the Hub

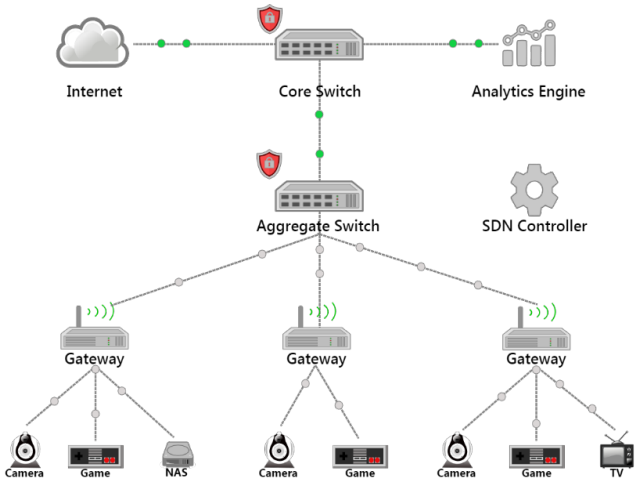

Figure 7 - Phased Deployment: Head End/Hub Updates

The first phase of deployment begins with upgrading the head end/hub by adding programmable switches with an analytics engine and an SDN controller. These switches can be included as part of distributed access architecture (DAA) upgrades. They provide the ability to mitigate a DDoS attack from a customer at the head end and identify which customer is the source of the attack.

This solution can also address ingress attacks from outside of the hub. If a local device is attacking another local device, the typical edge-based DDoS mitigation model will not work.

There are no changes to the CM for this phase. The primary limitation to this solution is that it cannot identify the device originating the attack, only the customer. The compromised device remains active and continues to participate in ongoing attacks.

7.2 Configuration Changes on Gateway Devices

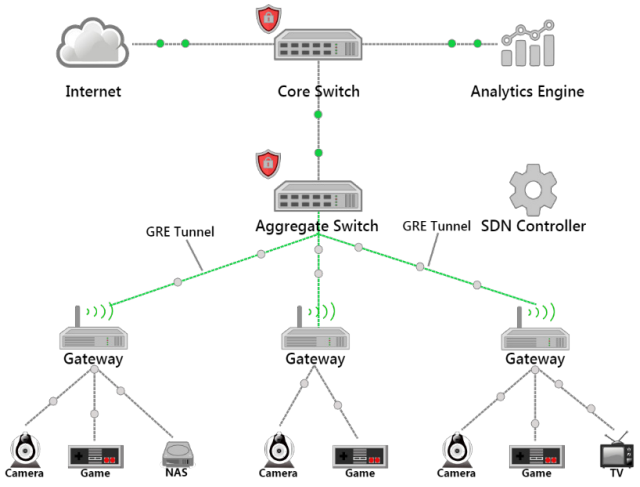

Figure 8 - Phased Deployment: Gateway Configuration

In this scenario, the operator can preserve the identity of the device originating the attack without implementing a firmware or hardware update. Instead, the IP and MAC addresses of the source device is encapsulated in tunnels between the CPE and the aggregate switch. Some CPE functionality, such as Network Address Translation (NAT) and port forwarding, is moved off the CM to the hub.

Tunnels cannot detect attacks directly, but creating, for example, a GRE or VXLAN tunnel from the customer premises to a central site can provide a profile of the customer device. This information is often missing from other analytics methods. However, because information for each packet on intermediate hops in not captured, this method is limited in its benefit to security.

Several CM operating systems include GRE support, so GRE tunnels can be deployed with only a configuration change on the modem. Deploying GRE tunnels without a firmware update reduces the cost and the time needed to deploy this option. Both RDK-B and OpenWRT support GRE tunnels.

7.3 Firmware Updates on Gateway Devices

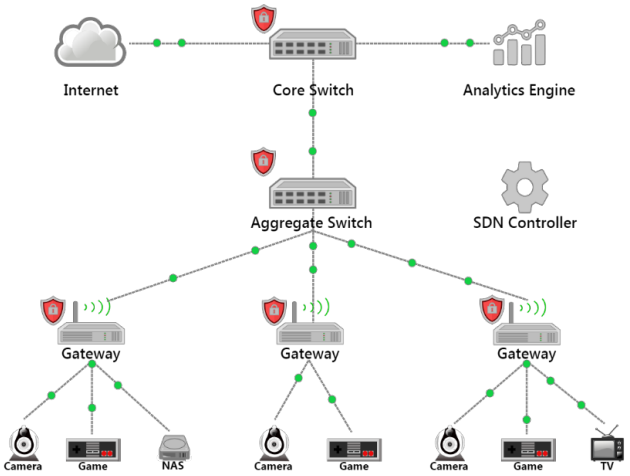

Figure 9 - Phased Deployment: Gateway Firmware Updates

As with the GRE tunnel, this implementation model enables the source device to be detected without new hardware. It also has the added benefit of distributing functionality back to the gateway, including blocking the malicious traffic before it leaves the customer premises. P4 support functionality is added to the gateway through a firmware update, exposing source IP and MAC addresses to the aggregation switch. As a result, INT data can be added to the packet and attack mitigation can occur in the gateway.

Once this model is deployed, the traffic on the access network will be scrubbed before it can impact any other customers, and customers will be able to address issues on compromised devices. The malicious traffic is dropped, and the count of blocked packets for each device is tracked. This model will not block benign traffic from the compromised device.

7.4 Hardware Updates to Gateway Devices

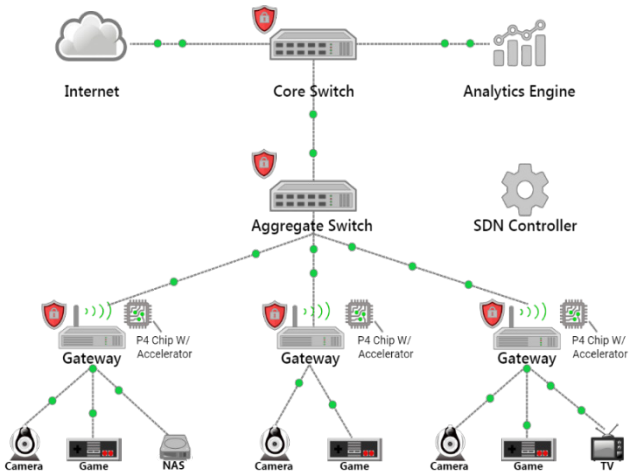

Figure 10 - Phased Deployment: Gateway Hardware Updates

Hardware updates to gateway devices in the form of a purpose-built chip can provide full line rate support by offloading P4 functionality to hardware, providing the same functionality as the firmware update but with improved network performance. Field-programmable gate arrays (FPGAs) can also be programmed to offload P4 capabilities.

The Open Compute Project (OCP) launched the Open Domain-Specific Architecture subgroup[8] to launch within the OCP Server Work Group. This project leverages a chiplet architecture, which can combine P4 acceleration cores with general-purpose CPUs. A chip composed of these chiplets can replace the current gateway CPUs used for packet processing.

8 Alternate Applications for Programmable Data Plane

The Transparent Security source-based DDoS use case is just one of many that can benefit from the programmable data plane. Once deployed, this architecture can improve many of the operations performed today. It will also open networks to new waves of innovation and allow operators to provide such things as network optimization and additional customer services.

With the programmable data plane, it is possible to change the behavior of the network after hardware has been deployed. Use of an analytics engine and controller, as is done with Transparent Security, can provide a closed-loop automation. Some of the network management capabilities available with the programmable data plane are listed below:

- packet flow tracking and optimization,
- prioritization on low-latency flows,
- active network monitoring/management, and
- traffic shaping.

Providing new services frequently requires installing new purpose-built hardware or deploying a virtual machine. With the programmable data plane, some of these services can be deployed on existing switches with very good performance. Some of the services that can be deployed with the programmable data plane include the following:

- firewalls,
- managed router as a service,
- Layer 4 load balancing, and
- SD-WAN (software-defined networking in a wide-area network).

Micronets is another CableLabs project that can leverage the programable data plane. Micronets creates additional layers of security within the customer premises and helps protect devices by using OpenFlow for the L3 traffic management. Transparent Security is focused on identifying devices participating in a DDoS attack and mitigating the attack at the source. The programable data plane, analytics engine, and controller used in Transparent Security can be leveraged by Micronets to improve packet processing performance and provide access to additional fields in the packet header. For more information on the Micronets project, visit www.cablelabs.com/micronets.

9 Conclusion

Distributed denial of service (DDoS) attacks and other cyberattacks can cost operators billions of dollars. With more and more devices coming into customers' homes and businesses, many of which with less than optimal security, this problem will only get worse. By quickly identifying attacks and blocking them before they reach the access network, Transparent Security can

- reduce the operations impact of large-scale attacks by mitigating closer to the source within seconds of an attack starting,
- eliminate the risk of revenue impact resulting from a failure to meet service-level agreements, and
- protect and enhance customer sentiments by avoiding large-scale DDoS attacks within a network.

[8] "New Open Domain-Specific Architecture Sub-Project to Launch Within the OCP Server Project," March 2019, Open Compute Project blog—Dirk Van Slyke

APPENDIX A INT Encapsulation Header Format

Inspection packets have INT metadata added to the header to be forwarded to the analytics engine (AE) component. The original IP packet is placed within an inspection header. The Ethernet frame is set to type 0×1212, a custom type that the AE in the proof of concept recognizes. The IP layer destination and port are set to the AE, but the originals are kept within the inspection header. This form will likely evolve over time a more efficient formats are developed.

A.1 Example INT Packet

```
0000 0800274cce28060b7e10771212120000
0010 00000101ac1f250c0ac57b0b04d20800
0020 45000021000100003d11d617ac1f250c
0030 ac1f2a69d78d270f000da8a648656c6c
0040 6f000000000000000000000000000xxxx
```

```
Frame 4: 78 bytes on wire (624 bits), 78 bytes captured (624 bits)
    Encapsulation type: Ethernet (1)
Ethernet II, Src: 06:0b:7e:10:77:12 (06:0b:7e:10:77:12), Dst: PcsCompu_4c:ce:28
(08:00:27:4c:ce:28)
    Destination: PcsCompu_4c:ce:28 (08:00:27:4c:ce:28)
        Address: PcsCompu_4c:ce:28 (08:00:27:4c:ce:28)
    Source: 06:0b:7e:10:77:12 (06:0b:7e:10:77:12)
        Address: 06:0b:7e:10:77:12 (06:0b:7e:10:77:12)
    Type: Unknown (0x1212)
Data (64 bytes)
    Data: 000000000101ac1f250c0ac57b0b04d20800450000210001...
    [Length: 64]
```

A.2 Inspection Header

```
000 xxxxxxxxxxxxxxxxxxxxxxxxxxxx0000
0010 00000101ac1f250c0ac57b0b04d20800
```

Field	Example Value
srcAddr	00:00:00:00:01:01
deviceAddr	ac 1f 25 0c (172.31.41.175)
dstAddr	0a c5 7b 0b (10.197.123.11)
dstPort	04 d2 (1234)
proto_id	0x800 (IP4)

A.3 UDP/IP Packet

```
0020   45 00 00 21 00 01 00 00 3d 11 d6 17 ac 1f 25 0c
0030   ac 1f 2a 69 d7 8d 27 0f 00 0d a8 a6 48 65 6c 6c
0040   6f xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```

```
Internet Protocol Version 4, Src: 172.31.41.175, Dst: 172.31.42.105
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Total Length: 33
User Datagram Protocol, Src Port: 56287, Dst Port: 1234
    Source Port: 56287
    Destination Port: 9999
    Length: 13
Data (5 bytes)
    Data: 48656c6c6f
    [Length: 5]
```

A.4 Header Structure

```
struct headers {
   ethernet_t   ethernet;
   inspection_t inspection;
   ipv4_t       ipv4;
   udp_t        udp;


}

const bit<16> TYPE_INSPECTION = 0x1212;
const bit<8> TYPE_UDP = 0x11;
const bit<16> TYPE_IPV4 = 0x800;
typedef bit<9>  egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;




struct header ethernet_t {
   macAddr_t dstAddr;
   macAddr_t srcAddr;
   bit<16>   etherType;

}

struct header inspection_t {
   macAddr_t srcAddr;
   ip4Addr_t deviceAddr;
   ip4Addr_t dstAddr;
   bit<16>   dstPort;
   bit<16> proto_id;

}

struct header ipv4_t {
   bit<4>    version;
   bit<4>    ihl;
   bit<8>    diffserv;
   bit<16>   totalLen;
   bit<16>   identification;
   bit<3>    flags;
   bit<13>   fragOffset;
   bit<8>    ttl;
   bit<8>    protocol;
   bit<16>   hdrChecksum;
   ip4Addr_t srcAddr;
   ip4Addr_t dstAddr;

}

struct header udp_t {
   bit<16> src_port;
   bit<16> dst_port;
   bit<16> len;
   bit<16> cksum;

}
```

DISCLAIMER

This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, noninfringement, or fitness for a particular purpose of this document, or any document referenced herein. Any use or reliance on the information or opinion in this document is at the risk of the user, and CableLabs and its members shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy, or utility of any information or opinion contained in the document.

CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various entities, technology advances, or changes in equipment design, manufacturing techniques, or operating procedures described, or referred to, herein.

This document is not to be construed to suggest that any company modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any of its members to purchase any product whether or not it meets the characteristics described in the document. Unless granted in a separate written agreement from CableLabs, nothing contained herein shall be construed to confer any license or right to any intellectual property. This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.