**AWTA Module: P4 Source Code**

|  |
| --- |
| **// Define headers**  **header ethernet\_t {**  **macAddr\_t dstAddr;**  **macAddr\_t srcAddr;**  **bit<16> etherType;**  **}**  **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;**  **bit<32> srcAddr;**  **bit<32> dstAddr;**  **}**  **// Define metadata to store real-time monitoring data**  **struct metadata\_t {**  **bit<32> packet\_count;**  **bit<32> flow\_rate;**  **bit<32> entropy\_score;**  **bit<32> window\_size;**  **bit<32> last\_entropy;**  **}**  **// Define headers and metadata**  **header ethernet\_t ethernet;**  **header ipv4\_t ipv4;**  **metadata\_t meta;**  **// Parse headers**  **parser MyParser(packet\_in pkt, out headers\_t hdr, inout metadata\_t meta) {**  **state start {**  **pkt.extract(ethernet);**  **transition select(ethernet.etherType) {**  **0x0800: parse\_ipv4;**  **default: accept;**  **}**  **}**  **state parse\_ipv4 {**  **pkt.extract(ipv4);**  **transition accept;**  **}**  **}**  **// State table to store monitoring data for dynamic window adjustment**  **table dynamic\_monitor\_table {**  **key = {**  **ipv4.srcAddr: exact;**  **ipv4.dstAddr: exact;**  **}**  **actions = {**  **increment\_counters;**  **adjust\_window;**  **reset\_window;**  **}**  **size = 1024;**  **default\_action = increment\_counters;**  **}**  **// Action to increment counters and capture traffic features**  **action increment\_counters() {**  **meta.packet\_count = meta.packet\_count + 1;**  **meta.flow\_rate = meta.packet\_count / meta.window\_size;**  **}**  **// Adjusts window size based on entropy score**  **action adjust\_window() {**  **if (meta.entropy\_score < meta.last\_entropy) {**  **meta.window\_size = max(meta.window\_size - 1, 5); // Decrease window size for high-intensity traffic**  **} else {**  **meta.window\_size = min(meta.window\_size + 1, 50); // Increase window size for low traffic**  **}**  **meta.last\_entropy = meta.entropy\_score;**  **}**  **// Reset window counters after processing**  **action reset\_window() {**  **meta.packet\_count = 0;**  **meta.entropy\_score = 0;**  **}**  **// Control block for ingress**  **control ingress {**  **apply(dynamic\_monitor\_table);**  **}**  **// Main control flow**  **control MyIngress(inout headers\_t hdr, inout metadata\_t meta, inout standard\_metadata\_t std\_meta) {**  **MyParser() apply(hdr, meta);**  **ingress.apply();**  **}** |

**Explanation of the P4 Code**

* **Header Definitions**: We define basic Ethernet and IPv4 headers, which are essential for handling incoming traffic in the SD-IoT environment.
* **Metadata**: The metadata structure includes fields like *packet\_count, flow\_rate, entropy\_score, window\_size*, and *last\_entropy*. These fields help track the necessary metrics for adaptive window adjustment and DDoS detection.
* **Parser**: The *MyParser* function extracts Ethernet and IPv4 headers, enabling the processing of IP traffic, which is typically the target of DDoS attacks.
* **State Table (***dynamic\_monitor\_table***)**: This state table stores source-destination pairs as keys and dynamically applies one of three actions: *increment\_counters, adjust\_window, or reset\_window*. This table allows the module to adjust window sizes dynamically based on detected traffic patterns.
* **Actions**:
* *increment\_counters*: This action counts packets and calculates the flow rate by dividing the packet count by the window size.
* *adjust\_window*: This action adjusts the window size based on entropy trends. If entropy decreases, it reduces the window size, enhancing monitoring resolution for high-intensity attacks.
* *reset\_window*: This action resets the counters at the end of each window to start the next monitoring cycle.
* **Ingress Control:** The ingress control block applies the state table and actions to incoming packets.

**Step-by-Step Deployment Instructions (Emulation with Mininet-WiFi)**

1. **Install P4 Development Environment**:
   * Install the P4 development environment on your machine. This includes the P4 compiler (*p4c*), behavioral model (*bmv2*), and Mininet-Wifi (for emulating the network).
2. **Compile the P4 Program**:
   * Compile the P4 program using the P4 compiler: *p4c --target bmv2 --arch v1model -o AWTA.json AWTA.p4*
   * This command generates a JSON file *(AWTA.json)* that describes the P4 pipeline for the switch.
3. **Set Up the Switch**:
   * Use the behavioral model (bmv2) to run a software switch with the compiled P4 program: *sudo simple\_switch --log-console -i 1@<interface> AWTA.json*
   * Replace <*interface*> with the appropriate network interface (e.g., eth0).
4. **Test the Program**:
   * Use Mininet to create a network topology with the P4 switch: *sudo mn --custom p4\_mininet\_script.py --topo p4\_topo --controller=remote*
   * Ensure that the traffic monitoring module is working correctly by generating traffic and observing the counters.
   * Emulate traffic using Mininet-WiFi to observe AWTA’s monitoring and DDoS detection capabilities.
   * Run commands on Mininet hosts to generate varying traffic and monitor the switch’s behavior.

**Step-by-Step Deployment Instructions (Real-World Deployment on P4-Enabled Switches)**

1. **Ensure P4 Hardware Support**:
   * Deploy the AWTA module on a P4-capable switch, like Barefoot Tofino, which supports real-world deployment.
2. **Compile and Load the Program**:
   * Use the hardware vendor’s compiler to compile AWTA.p4 into a compatible binary format. Load the program onto the switch following the vendor's instructions.
3. **Set Up Controller**:
   * Connect a remote controller (e.g., ONOS or ODL) to the switch to manage the network and interact with AWTA for ongoing adjustments.
4. **Traffic Simulation and Monitoring**:
   * Generate legitimate and DDoS attack traffic in the network. Use tools like *iperf* or *hping3* to simulate different traffic patterns and observe AWTA’s response in real-time.