Date: May 5, 2026

Revision	Date	Changes
1.0	May 5, 2026	Initial release
1.1	May 7, 2026	Clarified 7280R3, 7500R3 and 7800R3 exposure is limited

The CVE-ID tracking this issue: CVE-2026-7473
CVSSv3.1 Base Score: 5.8 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:N/I:L/A:N)
CVSSv4.0 Base Score: 6.8
(CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:N/VI:L/VA:N/SC:N/SI:L/SA:N)
Common Weakness Enumeration: CWE-1023: Incomplete Comparison with Missing Factors
This vulnerability is being tracked by BUG1086442 and BUG1519884

Description

On affected platforms running Arista EOS where a tunnel decapsulation configuration—such as VXLAN (Virtual Extensible LAN), decap-groups, or a GRE (Generic Routing Encapsulation) tunnel interface—is present, the switch will incorrectly decapsulate and forward other unexpected tunneled packet with a destination IP matching its configured decapsulation IP. This occurs because the switch does not verify the tunnel protocol type, potentially leading to the unexpected processing of non-configured tunnel traffic.

This issue has been reported as being exploited in the wild.

Arista would like to acknowledge and thank Scott Christiansen, Lukas Peitz, Rich Compton, and Jonathan Davis at Comcast for responsibly reporting CVE-2026-7473.

Vulnerability Assessment

Affected Software

EOS Versions

- All later releases are affected
- All releases in the 4.36.x train
- All releases in the 4.35.x train
- All releases in the 4.34.x train
- All releases in the 4.33.x train
- All releases in the 4.32.x train
- All releases in the 4.31.x train
- All releases in the 4.30.x train
- All releases in trains older than 4.30.x

Affected Platforms

The following products **are** affected by this vulnerability:

- Arista EOS-based products:
  - 7020R Series
  - 7280R/R2Series
  - 7500R/R2 Series
    Limited exposure (IP-in-IPv6 and GUEv6) on:
  - 7280R3 Series
  - 7500R3 Series
  - 7800R3 Series

The following product versions and platforms **are not** affected by this vulnerability:

- Arista EOS-based products:
  - 710 Series
  - 720D Series
  - 720XP/722XPM Series
  - 750X Series
  - 7010 Series
  - 7010X Series
  - 7020R4 Series
  - 7130 Series running EOS
  - 7150 Series
  - 7160 Series
  - 7170 Series
  - 7050X/X2/X3/X4 Series
  - 7060X/X2/X4/X5/X6 Series
  - 7250X Series
  - 7260X/X3 Series
  - 7280E/R4 Series
  - 7300X/X3 Series
  - 7320X Series
  - 7358X4 Series
  - 7368X4 Series
  - 7388X5 Series
  - 7500E Series
  - 7800R4 Series
  - 7700R4 Series
  - AWE 5000 Series
  - AWE 7200R Series
  - CloudEOS
  - cEOS-lab

    - vEOS-lab
    - CloudVision eXchange, virtual or physical appliance
- Arista Wireless Access Points
- CloudVision CUE, virtual appliance or physical appliance
- CloudVision CUE cloud service delivery
- CloudVision Portal, virtual appliance or physical appliance
- CloudVision as-a-Service
- CloudVision AGNI
- Arista 7130 Systems running MOS
- Arista Converged Cloud Fabric (formerly Big Switch BCF)
- Arista DANZ Monitoring Fabric (formerly Big Switch BMF)
- Arista Network Detection and Response (NDR) Security Platform (Formerly Awake NDR)
- Arista Edge Threat Management - Arista NG Firewall and Arista Micro Edge (Formerly Untangle)
- Arista NetVisor OS, Arista NetVisor UNUM, and Insight Analytics (Formerly Pluribus)
- VeloCloud Orchestrator (Formerly VeloCloud Orchestrator by Broadcom)
- VeloCloud Gateway (Formerly VeloCloud Gateway by Broadcom)
- VeloCloud Edge (Formerly VeloCloud Edge by Broadcom)

Required Configuration for Exploitation

In order to be vulnerable to CVE-2026-7473, the following condition must be met:

The device must be configured as a tunnel endpoint with a decapsulation IP — for example, as a VXLAN VTEP, a GRE tunnel endpoint, or with an ip decap-group.

A device configured to decapsulate one tunnel type will also incorrectly accept and decapsulate other tunnel protocols destined to the same IP address, even if those protocols were not explicitly configured. The following table summarizes which additional tunnel types a device will decapsulate based on its configured decapsulation type (note that some cases require extra protocol specific configurations for traffic to be decapsulated). Note that in all cases the inner header could be IPv4 or IPv6.

Note on Platforms:

- All scenarios below apply to 7020R Series, 7280R/R2 Series, and 7500R/R2 Series.
- Only the IP-in-IPv6 and GUE IPV6 Decap Group scenarios apply to 7280R3 Series, 7500R3 Series, and 7800R3 Series.

Configured decapsulation tunnel type	Unexpected decapsulation of tunnel type traffic to configured decap IP	Additional configurations required for exploitation
VXLAN IPv4 Tunnel Interface	GRE, IPoIP	None

	NVGRE	TNI in NVGRE packet must match a VXLAN VNI configured on switch
GRE IPv4 Tunnel Interface	VXLAN	VXLAN Tunnel Interface (VTI) and VNI mapping must be configured
	Generic UDP Encapsulation (GUE)	GUE Decap Group and relevant UDP destination port to payload mapping must be configured. Both source and destination IP must match GRE tunnel configuration.
	IPoIP	Both source and destination IP must match GRE tunnel configuration.
GRE IPv4 Decap Group	IPoIP	None
	VXLAN	VXLAN Tunnel Interface (VTI) and VNI mapping must be configured
	GUE	GUE Decap Group and relevant UDP destination port to payload mapping must be configured.
	NVGRE	VXLAN Tunnel Interface (VTI) must be configured. TNI in NVGRE packet must match a VXLAN VNI configured on switch.
GUE IPv4 Decap Group	GRE, IPoIP	None
IP-in-IPv4 Decap Group	GRE	None
	NVGRE	VXLAN Tunnel Interface (VTI) must be configured. TNI in NVGRE packet must match a VNI configured on switch.
	VXLAN	VXLAN Tunnel Interface (VTI) and VNI mapping must be configured
	GUE	GUE Decap Group and

		relevant UDP destination port to payload mapping must be configured.
IP-in-IPv6 Decap Group	GREv6	None
	GUEv6	GUE Decap Group and relevant UDP destination port to payload mapping must be configured.
GUE IPv6 Decap Group	IP-in-IPv6, GREv6	None

To check if the device is acting as a VXLAN VTEP:

```
switch>show interfaces vxlan 1
 Vxlan1 is up, line protocol is up (connected)
   Source interface is Loopback1 and is active with 10.0.0.1
   Listening on UDP port 4789
   ...
```

If the output contains "**Source interface is <interface> and is active with <IP>**", the device is acting as a VXLAN VTEP with <IP> as the tunnel termination address, and is potentially impacted.

To check if a GRE tunnel interface is configured:

```
switch>show interfaces Tunnel0
 Tunnel0 is up, line protocol is up
   Hardware is Tunnel
   Tunnel source 1.1.1.1, destination 1.1.1.2
   Tunnel protocol/transport GRE/IP
   ...
```

If the tunnel interface is up with a source and destination, the device is a GRE tunnel endpoint and is potentially impacted.

To check if decap-groups are configured:

```
switch>show ip decap-group
```

If none of the above outputs show the presence of any tunnel endpoint configurations, the device does not perform tunnel decapsulation and is not exposed to this issue.

Indicators of Compromise

When tunnel decapsulation configuration is present on the switch, the switch should decapsulate and forward only packets of the corresponding tunnel type. On impacted switches, it instead incorrectly decapsulates and forwards other unexpected tunneled packets whose destination IP matches a decap IP configured on the switch.

Indicators may include:

- Unexpected traffic appearing on internal network segments that should only be reachable via configured tunnel types.
- Tunneled traffic of a non-configured protocol (e.g., GRE) being decapsulated and forwarded when only a different tunnel type (e.g., VXLAN) is configured.

If per entry counters are enabled in ACL like what has been shown in the Mitigation section below, blocked unexpected traffic will be listed in the output from following show command:

```
switch>show mac access-lists bar
MAC Access List bar
        counters per-entry
        1 permit any any ip payload alias ip-next-protocol-gre alias ip-dip-decap-
ip [match 1 packets, 0:00:06 ago]
        2 deny any any ip pay
load alias ip-dip-decap-ip [match 3 packets, 0:20:06 ago]
        3 permit any any
        (implicit) deny any any

        Total rules configured: 3
        Configured on Ingress: Et30/1
        Active on     Ingress: Et30/1
```

This example is showing that 3 unexpected packets to the decap IP have been denied by the configured ACL.

Mitigation

There are two broad approaches to mitigate this issue - (1) applying ACLs on upstream devices or (2) applying ACLs on the devices where the unexpected decapsulation is happening. In both cases, the idea is to either selectively allow only legitimate tunnel traffic or to selectively block malicious tunnel traffic. For example, if a network is configured to forward VXLAN traffic, but

GRE traffic is being unexpectedly forwarded, then ACLs can be used to either selectively allow just VXLAN traffic or selectively block GRE traffic. More details about using the ACL feature can be found in the Arista User Manual.

The following configurations align with the recommendations outlined in the Arista EOS Hardening Guide.

Approach 1 - Applying ACL on Upstream Switches

On upstream devices, applying ACLs to allow specific tunneled traffic is straightforward. ACLs can be applied that match on tunnel destination IP, the IP next protocol field, and (optionally) UDP destination port to selectively allow or block specific tunnel protocols.

Example ACLs for Arista EOS follows.

ACL to permit VXLANv4 Only

This IPv4 ACL matches on VXLAN packets as follows:
(a) IP next protocol = UDP (17)
(b) IP DIP = VXLAN VTEP IP
(c) UDP destination port = VXLAN UDP Port (4789)

It allows VXLAN packets and drops all other packets to the VXLAN Decap IP.

```
ip access-list foo
   counters per-entry
   1 permit udp any host <vxlan-decap-ip> eq 4789
   2 deny ip any host <decap-ip>
   3 permit ip any any
```

ACL to permit GREv4 Only

This IPv4 ACL matches on GRE packets as follows:
(a) IP next protocol = GRE (47)
(b) IP DIP = GRE Tunnel Destination IP

It allows GRE packets and drops all other packets to the GRE Decap IP.

```
ip access-list foo
   counters per-entry
   1 permit gre any host <gre-decap-ip>
   2 deny ip any any host <gre-decap-ip>
   3 permit any any
```

ACL to permit IP-in-IPv4 Only

This IPv4 ACL matches on IP-in-IPv4 packets as follows:
(a) IP next protocol = IPv4 (4) or IPv6 (41)
(b) IP DIP = IP-in-IP Decap IP

It allows IP-in-IPv4 packets and drops all other packets to the IP-in-IPv4 Decap IP.

```
ip access-list foo
   counters per-entry
   1 permit 4 any host <ipip-decap-ip>
   2 permit 41 any host <ipip-decap-ip>
   3 deny ip any host <ipip-decap-ip>
   4 permit any any
```

ACL to Permit IP-in-IPv6 Only

This IPv6 ACL matches on IP-in-IPv6 packets as follows:
(a) IP next protocol = IPv4 (4) or IPv6 (41)
(b) IP DIP = IP-in-IP Decap IP

It allows IP-in-IPv6 packets and drops all other packets to the IP-in-IPv6 Decap IP.

```
ipv6 access-list foo
   counters per-entry
   1 permit 4 any host <ipip-decap-ip>
   2 permit 41 any host <ipip-decap-ip>
   3 deny ipv6 any host <ipip-decap-ip>
   4 permit ipv6 any any
```

ACL to permit GUEv4 Only

This IPv4 ACL matches on GUE packets as follows:
(a) IP next protocol = UDP (17)
(b) IP DIP = GUE Decap IP
(c) UDP destination port = UDP port configured per payload
(IP = Y or MPLS = Z)

It allows GUE packets and drops all other packets to the GUE Decap IP.

```
ip access-list foo
   counters per-entry
   1 permit udp any host <decap-ip> eq Y
```

```
   2 permit udp any host <decap-ip> eq Z
   3 deny ip any host <decap-ip>
   4 permit ip any any
```

ACL to Permit GUEv6 Only

This IPv6 ACL matches on GUE packets as follows:
(a) IP next protocol = UDP (17)
(b) IP DIP = GUE Decap IP
(c) UDP destination port = UDP port configured per payload
(IP = Y or MPLS = Z)

It allows GUE packets and drops all other packets to the GUE Decap IP.

```
ipv6 access-list foo
   counters per-entry
   1 permit udp any host <decap-ip> eq Y
   2 permit udp any host <decap-ip> eq Z
   3 deny ipv6 any host <decap-ip>
   4 permit ipv6 any any
```

Approach 2 - Applying ACL on Decapsulation Switches

Applying ACLs on the decapsulation device is more complicated. It requires the use of MAC ACLs on 7020R Series, 7280R/R2 Series, and 7500R/R2 Series systems and IP ACLs on 7280R3 Series, 7500R3 Series, and 7800R3 Series systems. In both cases, a TCAM profile update is also required.

7020R Series, 7280R/R2 Series, and 7500R/R2 Series

Mitigation involves using MAC ACLs to allow specific expected protocol packets and block all other traffic to the configured decap IPs. The suggested MAC ACLs use User Defined Fields (UDFs) to match on specific fields in the packet headers. This requires a TCAM profile update to include the following UDF qualifiers:

1. For IPv4 tunnels, 2 16b and 1 32b UDF qualifiers need to be included.
2. For IPv6 tunnels, 2 16b and 4 32b UDF qualifiers need to be included.

However, in order to make room for the UDF qualifiers, other TCAM features/qualifiers must be removed due to hardware constraints. Following are some suggested TCAM profile changes to accommodate the required UDF qualifiers:

1. TCAM profile that includes the UDF qualifiers for IPv4 tunnels, but removes support for

MPLS:

```
hardware tcam
   profile test copy default
      feature acl port mac
         no key size limit
         key field udf-16b-1 udf-16b-2 udf-32b-1
      no feature mpls
      no feature mpls pop ingress
      no feature pbr mpls
```

2. TCAM profile that includes the UDF qualifiers for IPv4 tunnels, but removes support for VXLAN:

```
hardware tcam
   profile test copy default
      feature acl port mac
         no key field src-mac
         key field udf-16b-1 udf-16b-2 udf-32b-1
```

3. TCAM profile that includes the UDF qualifiers for IPv6 tunnels, but removes support for VXLAN and PBR:

```
hardware tcam
   profile test1 copy default
      feature acl port mac
         no key size limit
         no key field src-mac dst-mac
         key field udf-16b-1 udf-16b-2 udf-32b-1 udf-32b-2 udf-32b-3 udf-32b-4
      no feature tunnel vxlan
      no feature tunnel vxlan routing
      no feature pbr ip
      no feature pbr ipv6
```

Please contact Arista TAC if further assistance is needed with TCAM profile construction.

ACL to permit VXLAN v4 Decap only

This MAC ACL uses UDF to match on VXLAN packets as follows:
(a) IP next protocol = UDP (0x11)
(b) IP DIP = VXLAN VTEP IP (0xXXXXXXXX)

(c) UDP destination port = VXLAN UDP Port (0x12b5)

It allows VXLAN packets and drops all other packets to the VXLAN Decap IP.

```
mac access-list payload alias ip-next-protocol-
udp offset 2 pattern 0x00110000 mask 0xff00ffff

mac access-list payload alias ip-dip-decap-
ip offset 4 pattern 0xXXXXXXXX mask 0x00000000

mac access-list payload alias udp-dport-
vxlan offset 5 pattern 0x000012b5 mask 0xffff0000

mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-udp alias ip-dip-decap-
ip alias udp-dport-vxlan
   2 deny any any ip payload alias ip-dip-decap-ip
   3 permit any any
```

ACL to permit GREv4 Decap Only

This MAC ACL uses UDF to match on GRE packets as follows:
(a) IP next protocol = GRE (0x2f)
(b) IP DIP = GRE Decap IP (0xXXXXXXXX)

It allows GRE packets and drops all other packets to the GRE Decap IP.

```
mac access-list payload alias ip-next-protocol-
gre offset 2 pattern 0x002f0000 mask 0xff00ffff

mac access-list payload alias ip-dip-decap-
ip offset 4 pattern 0xXXXXXXXX mask 0x00000000

mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-gre alias ip-dip-decap-ip
   2 deny any any ip payload alias ip-dip-decap-ip
   3 permit any any
```

If needed, the ACL can also be tweaked to match on specific GRE payloads as follows:

IPv4oGRE

ACL also matches on GRE next protocol = IPv4 (0x0800)

```
mac access-list payload alias gre-protocol-
ipv4 offset 5 pattern 0x00000800 mask 0xffff0000

mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-gre alias ip-dip-decap-
ip alias gre-protocol-ipv4
   2 deny any any ip payload alias ip-dip-decap-ip
   3 permit any any
```

IPv6oGRE

ACL also matches on GRE next protocol = IPv6 (0x86dd)

```
mac access-list payload alias gre-protocol-
ipv6 offset 5 pattern 0x000086dd mask 0xffff0000
mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-gre alias ip-dip-decap-
ip alias gre-protocol-ipv6
   2 deny any any ip payload alias ip-dip-decap-ip
   3 permit any any
```

MPLSoGRE

ACL also matches on GRE next protocol = MPLS (0x8847)

```
mac access-list payload alias gre-protocol-
mpls offset 5 pattern 0x00008847 mask 0xffff0000

mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-gre alias ip-dip-decap-
ip alias gre-protocol-mpls
   2 deny any any ip payload alias ip-dip-decap-ip
   3 permit any any
```

ACL to permit IP-in-IPv4 Decap Only

This MAC ACL uses UDF to match on IP-in-IP packets as follows:
(a) IP next protocol = IPv4 (0x04) or IPv6 (0x29)
(b) IP DIP = IP-in-IP Decap IP (0xXXXXXXXX)

It allows IP-in-ip packets and drops all other packets to the IP-in-IP Decap IP.

```
mac access-list payload alias ip-next-protocol-
ipv4 offset 2 pattern 0x00040000 mask 0xff00ffff

mac access-list payload alias ip-next-protocol-
ipv6 offset 2 pattern 0x00290000 mask 0xff00ffff

mac access-list payload alias ip-dip-decap-
ip offset 4 pattern 0xXXXXXXXX mask 0x00000000
mac access-list foo
   counters per-entry
   1 permit any any ip payload alias ip-next-protocol-ipv4 alias ip-dip-decap-ip
   2 permit any any ip payload alias ip-next-protocol-ipv6 alias ip-dip-decap-ip
   3 deny any any ip payload alias ip-dip-decap-ip
   4 permit any any
```

ACL to permit GUEv4 Decap Only

This MAC ACL uses UDF to match on GUE packets as follows:
(a) IP next protocol = UDP (0x11)
(b) IP DIP = GUE Decap IP (0xXXXXXXXX)
(c) UDP destination port = UDP port configured per payload
(IP = 0xYYYY or MPLS = 0xZZZZ)

It allows GUE packets and drops all other packets to the GUE Decap IP.

```
mac access-list payload alias ip-next-protocol-
udp offset 2 pattern 0x00110000 mask 0xff00ffff

mac access-list payload alias ip-dip-decap-
ip offset 4 pattern 0xXXXXXXXX mask 0x00000000

mac access-list payload alias udp-dport-gue-
ip offset 5 pattern 0x0000YYYY mask 0xffff0000

mac access-list payload alias udp-dport-gue-
mpls offset 5 pattern 0x0000ZZZZ mask 0xffff0000
```

```
mac access-list foo
   1 permit any any ip payload alias ip-next-protocol-udp alias ip-dip-decap-
ip alias udp-dport-gue-mpls
   2 permit any any ip payload alias ip-next-protocol-udp alias ip-dip-decap-
ip alias udp-dport-gue-ip
   3 deny any any ip payload alias ip-dip-decap-ip
   4 permit any any
```

ACL to permit GUEv6 Decap Only

This MAC ACL uses UDF to match on GUE packets as follows:
(a) IP next protocol = UDP (0x11)
(b) IP DIP = GUE Decap IP (0xAAAAAAAABBBBBBBBCCCCCCCCDDDDDDDD)
(c) UDP destination port = UDP port configured per payload
(IP = 0xYYYY or MPLS = 0xZZZZ)

It allows GUE packets and drops all other packets to the GUE Decap IP.

```
mac access-list payload alias ipv6-next-protocol-
udp offset 1 pattern 0x00001100 mask 0xffff00ff

mac access-list payload alias udp-dport-gue-
ip offset 10 pattern 0x0000YYYY mask 0xffff0000

mac access-list payload alias udp-dport-gue-
mpls offset 10 pattern 0x0000ZZZZ mask 0xffff0000

mac access-list payload alias ipv6-dip-decap-ip1 offset 6 pattern 0xAAAAAAAA mask 0

mac access-list payload alias ipv6-dip-decap-ip2 offset 7 pattern 0xBBBBBBBB mask 0

mac access-list payload alias ipv6-dip-decap-ip3 offset 8 pattern 0xCCCCCCCC mask 0

mac access-list payload alias ipv6-dip-decap-ip4 offset 9 pattern 0xDDDDDDDD mask 0

mac access-list foo
   counters per-entry
   1 permit any any ipv6 payload alias ipv6-next-protocol-udp alias ipv6-dip-decap-ip
1 alias ipv6-dip-decap-ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-
ip4 alias udp-dport-gue-ip
   2 permit any any ipv6 payload alias ipv6-next-protocol-udp alias ipv6-dip-decap-ip
1 alias ipv6-dip-decap-ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-
ip4 alias udp-dport-gue-mpls
```

```
   3 deny any any ipv6 payload alias ipv6-dip-decap-ip1 alias ipv6-dip-decap-
ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-ip4
   4 permit any any
```

ACL to permit IP-in-IPv6 Decap Only

The MAC ACL uses UDF to match on IP-in-IPv6 packets as follows:
(a) IP next protocol = IPv4 (4) or IPv6 (41)
(b) IP DIP = IP-in-IP Decap IP (0xAAAAAAAABBBBBBBBCCCCCCCCDDDDDDDD)

It allows IP-in-ip packets and drops all other packets to the IP-in-IP Decap IP.

```
mac access-list payload alias ipv6-next-protocol-
ipv4 offset 1 pattern 0x00000400 mask 0xffff00ff

mac access-list payload alias ipv6-next-protocol-
ipv6 offset 1 pattern 0x00002900 mask 0xffff00ff

mac access-list payload alias ipv6-dip-decap-ip1 offset 6 pattern 0xAAAAAAAA mask 0

mac access-list payload alias ipv6-dip-decap-ip2 offset 7 pattern 0xBBBBBBBB mask 0

mac access-list payload alias ipv6-dip-decap-ip3 offset 8 pattern 0xCCCCCCCC mask 0

mac access-list payload alias ipv6-dip-decap-ip4 offset 9 pattern 0xDDDDDDDD mask 0

mac access-list foo
   counters per-entry
   1 permit any any ipv6 payload alias ipv6-next-protocol-ipv4 alias ipv6-dip-decap-
ip1 alias ipv6-dip-decap-ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-ip4
   2 permit any any ipv6 payload alias ipv6-next-protocol-ipv6 alias ipv6-dip-decap-
ip1 alias ipv6-dip-decap-ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-ip4
   3 deny any any ipv6 payload alias ipv6-dip-decap-ip1 alias ipv6-dip-decap-
ip2 alias ipv6-dip-decap-ip3 alias ipv6-dip-decap-ip4
   4 permit any any
```

7280R3 Series, 7500R3 Series, and 7800R3 Series

Mitigation involves using IPv6 PACLs to allow specific expected protocol packets and block all other traffic to the configured decap IPs. This requires the following TCAM profile update with the specified packet types:

```
hardware tcam
   profile test
      feature acl port ipv6
         packet ipv6 ipv4 forwarding routed decap
         packet ipv6 ipv6 forwarding routed decap
         packet ipv6 gue ipv4 forwarding routed decap
         packet ipv6 gue ipv6 forwarding routed decap
         packet ipv6 gue mpls forwarding mpls decap
```

Note that introducing new packet types might also require specifying them under other features such as “acl vlan” or “qos ipv6”. Please reach out, if further assistance is needed with TCAM profile construction.

ACL to Permit GUEv6 Only

This IPv6 ACL matches on GUE packets as follows:
(a) IP next protocol = UDP (0x11)
(b) IP DIP = GUE Decap IP
(c) UDP destination port = UDP port configured per payload
(IP = Y or MPLS = Z)

It allows GUE packets and drops all other packets to the GUE Decap IP.

```
ipv6 access-list foo
   counters per-entry
   1 permit udp any host <decap-ip> eq Y
   2 permit udp any host <decap-ip> eq Z
   3 deny ipv6 any host <decap-ip>
   4 permit ipv6 any any
```

ACL to Permit IP-in-IPv6 Only

This IPv6 ACL matches on IP-in-IPv6 packets as follows:
(a) IP next protocol = IPv4 (4) or IPv6 (41)
(b) IP DIP = IP-in-IP Decap IP

It allows IP-in-IPv6 packets and drops all other packets to the IP-in-IPv6 Decap IP.

```
ipv6 access-list foo
   counters per-entry
   1 permit 4 any host <decap-ip>
   2 permit 41 any host <decap-ip>
```

```
3 deny ipv6 any host <decap-ip>
4 permit ipv6 any any
```

Resolution

No software upgrade path is planned to address this issue due to the risk of breaking existing configuration on deployments. The recommended resolution of this issue is to follow the appropriate mitigation instructions detailed above.

Hotfix

No hotfix is available for these issues.

For More Information

If you require further assistance, or if you have any further questions regarding this security notice, please contact the Arista Networks Technical Assistance Center (TAC) by one of the following methods:

Open a Service Request

Contact information needed to open a new service request may be found at:
https://www.arista.com/en/support/customer-support