Lightweight 4over6 Test-bed for Security Analysis

Ameen Al-Azzawi, and Gábor Lencse

Abstract—**In this paper, we focus on one of the most prominent IPv6 transition technologies, namely lw4o6 (Lightweight 4over6). We emphasize the uniqueness of lw4o6 and the difference between lw4o6 and the conventional DS-Lite (Dual-Stack Lite), their topology, functionality and security vulnerabilities. We analyze the potential vulnerabilities of lw4o6 infrastructure by applying the STRIDE threat modelling technique, which stands for Spoof- ing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. Moreover, we build a test-bed for lw4o6 using Snabb, which is an open source software. We test Snabb's tunneling and binding capabilities and most importantly, port allocation per subscriber. At the end, we present multiple attacking scenarios (Denial of Service, Information Disclosure, Spoofing, etc.) against lw4o6's main routers and come up with mitigation methods for such attacks.**

Index Terms—**4in6, lw4o6, DS-Lite, encapsulation, IPv6, DoS, Snabb, PSID, Scapy.**

I. INTRODUCTION

AFTER the depletion of the public IPv4 address pool in 2011 [1], several technologies were presented by the scientific community: research institutes, industrial vendors and ISPs (Internet Service Providers). All of them aimed to fulfill one commitment, reliable communication between two remote machines that have different IP versions (IPv4 and IPv6), or the same IP version but with the infrastructure between them adopting another IP version.

We have been focusing our research on the most prominent IPv6 transition technologies and conducted a survey on them [2], where we concluded and shortlisted some of the most promising technologies. For example, the combination of NAT64 [3] and DNS64 [4] proved to be effective in certain areas. However, it did not solve the issue of IPv4-only applications. Therefore, another technology called 464XLAT [5] has been developed to tackle this problem, which has a double translation mechanism using CLAT (customer-side translator) and PLAT (provider-side translator). We have published several papers [6], [7], where we have analyzed the security threats that the 464XLAT infrastructure might face, especially CLAT and PLAT routers using the STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, and Elevation of privilege) method [8].

Furthermore, we have published another article [9], where we have built a test-bed for 464XLAT, and have tested its capabilities and how it reacts to potential security threats. The `hping3` package was used to flood the PLAT router with an excessive amount of packets. We concluded that 464XLAT is a reliable technology when it comes to IPv4-only devices communicating over IPv6 island. However, it has some security vulnerabilities that can be leveraged by an adversary, such as DoS (Denial of Service) attack.

Moreover, another technology called DS-Lite was invented to tackle the same issue of IPv4 depletion, it consists of two main routers: B4 (Basic Bridging BroadBand) and AFTR (Address Family Transition Router) [10]. We have covered the security analysis of DS-Lite in [11]. However, DS-Lite has an issue with scalability as all of the NAPT operations were executed by the AFTR router, which makes it very hard to scale its operation. Therefore, lw4o6 technology was invented to function as an improved version of DS-Lite [12]. In our current paper, the target is to conduct an analysis of the potential security vulnerabilities that the lw4o6 infrastructure may face. To that end, the STRIDE threat modeling technique [8] is used. We plan to fulfill our objectives as follows:

- Utilizing the STRIDE threat modeling technique on the Data-Flow Diagram (DFD) of a lw4o6 lw4o6 system to identify and uncover any possible security vulnerabilities.
- Constructing a reliable test-bed for the lw4o6 system using open-source software to assess its functionality and translation process.
- Conducting multiple attack scenarios on the primary routers of the lw4o6 infrastructure to evaluate their resilience, effectiveness, and potential security weaknesses.
- Ultimately, suggesting practical mitigation strategies to address and counteract such attacks.

The remainder of this paper is organized as follows. In Section II, we provide an overview of the operation of the lw4o6 infrastructure and, more importantly, the core differences between lw4o6 and DS-Lite. We emphasize the fundamental differences between the two technologies in terms of their topology, functionality, and scalability. In Section III, we talk about the previous studies that tried to build a test-bed or analyze lw4o6 transition technology. In Section IV, we describe the implementation of lw4o6 infrastructure and how it can be built. In Section V, we analyze the potential security vulnerabilities within lw4o6 using the STRIDE method after we build its DFD (Data Flow Diagram). In Section VI, we present our lw4o6 test-bed, whereas, in Section VII, we present our results and analyze them, where we also present two attacking scenarios and their mitigation methods.

In Section VIII, we conclude our paper by summarizing the significance of our test-bed and the lesson learned from its results.

A. Al-Azzawi is with the Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary. (e-mail: alazzawi@hit.bme.hu).

G. Lencse is with the Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary. (e-mail: lencse@hit.bme.hu).

DOI: 10.36244/ICJ.2023.3.4

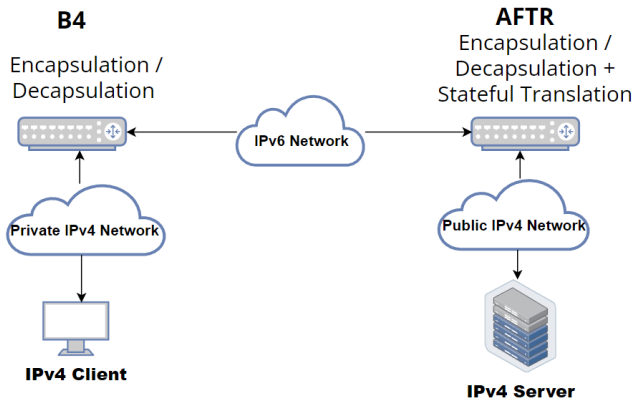

Fig. 1. DS-Lite Topology.

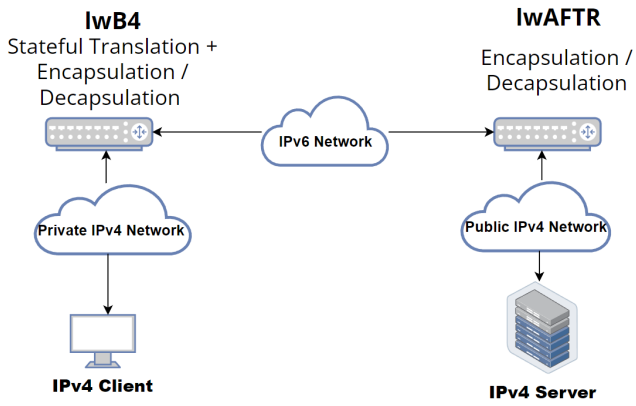

Fig. 2. Lw4o6 Topology

II. Lw4o6 Operation

Lw4o6 was defined in RFC-7596 [12], as an improved version of DS-Lite [10]. To understand the essence of this extended technology, we will analyze the added values that lw4o6 presents.

A. DS-Lite Topology

Fig.1 shows the topology of DS-Lite and its encapsulation / decapsulation process. DS-Lite is based on two core functions:

- B4 (Basic Bridging BroadBand): It is responsible for encapsulating IPv4 packets into IPv6 ones and forwarding the 4in6 traffic to the AFTR (Address Family Transition Router). It also decapsulates the returning 4in6 traffic by extracting the original IPv4 packet from the payload and forwards it to the IPv4 client residing behind it [10].
- AFTR: It decapsulates the 4in6 traffic that is generated by the B4 router, then statefully translates the original private IPv4 source address into a public one [10]. Moreover, AFTR works also as an encapsulator, when it encapsulates the returning packet from the IPv4 server into IPv6 tunnel and sends it to the designated B4 router. AFTR adopts a method called per-flow state [12]. It processes every packet as a part of a flow and compares this incoming packet with its binding table by checking the softwire ID (B4 IPv6 address) then scans its binding table for translation purposes.

B. Lw4o6 Topology

Fig. 2 shows the topology of lw4o6 with its main components. Lw4o6 works in a similar manner as DS-Lite, however, it has one main improvement, which made it more scalable than the conventional DS-Lite. The stateful translation has been moved from the centralized AFTR into the B4. This change made a big difference for the ISPs, and can be summarized by the below points:

- Lw4o6 optimizes the work for ISPs by avoiding the complicated process of stateful translation.
- The ”per flow state” method is replaced by ”per subscriber state”, which will save a lot in terms of CPU and memory consumption [12]. This is considered to be the most important feature of lw4o6, where every subscriber or CPE (Customer Premise Equipment) has an allocated and dedicated port set that can be used in his communication. This opens up the possibility for multiple lwB4 routers to share the same public IPv4 address in their softwire (tunnel) under the condition that they use two different port sets [12].
- The fact that no stateful translation is required on the lwAFTR side means that considerably less logging in the ISP side is actually required [12].

In other words, lw4o6 is an optimization of DS-Lite [12], as it reduces the overhead at the lwAFTR side by relocating the stateful translation to the lwB4 side.

Lw4o6 infrastructure consists of two main types of routers, lwB4 and lwAFTR:

- lwB4: works as stateful NAT44 translator and encapsulator / decapsulator.
- lwAFTR: works only as encapsulator / decapsulator.

Softwire is a mechanism that allows the encapsulation and transport of IPv4 traffic over an IPv6 network (or vice versa). Moreover, it is a tunnel (“virtual wire”) that carries IPv4 traffic over an IPv6 infrastructure. In the lw4o6 case, softwire encapsulates IPv4 packets within IPv6 packets and facilitates communication between two IPv4 networks using an IPv6 network [12].

C. Port Set Allocation

As described previously, lwAFTR allocates a specific port set for every lwB4 router using a method illustrated in RFC-7597 [13] Section 5.1, where PSID (Port Set Identifier) can be calculated as follows.

For example, to configure a softwire at the lwAFTR side, every softwire has to have the below three parameters (the numbers we used here are just examples):

- Port set size = 10 $\Rightarrow$ 2^{10} = 1024 ports per set.
- PSID length = 6 $\Rightarrow$ number of port sets: 2^6 = 64, it is also called ”the sharing ratio” [13].
- PSID = 1 $\Rightarrow$ allocated ports range = [1024-2047].

To explain it in a simpler way, let us calculate the number of concurrent lwB4s (subscribers) who can share the same public IPv4 address, while having different port sets. The total number of source ports numbers is 2^{16} = 65536, divided by 64 (number of port sets) is 1024. This number represents the size of one ports set, which can be also concluded from PSID size value (2^{10}). So, we have the number of sets, and the size of the sets themselves, all that is left is to select the PSID value, which will decide the exact port set to be allocated for the specific CPE (subscriber).

- PSID = 0 ⇒ allocated ports = [0 - 1023].
- PSID = 1 ⇒ allocated ports = [1024 - 2047].
- PSID = 8 ⇒ allocated ports = [8192 - 9215].

In conclusion, with PSID length = 6, we can support 64 different CPEs (subscribers) with 1024 ports for each of them. Furthermore, by selecting PSID = 1, the range of allocated ports will amount to [1024 - 2047]. However, if we exclude the first set, which contains the well-known ports [0 - 1023], we end up with 63 subscribers. That means 63 subscribers have the possibility of sharing the same public IPv4 address.

III. Related Work

There is a very limited amount of research available in the field of lw4o6 IPv6 transition technology.

Ahmed Al-hamadani proposed a test environment for benchmarking lw4o6 and especially its two main components (lwB4 and lwAFTR) [14]. The author carried out an analysis for the operational requirement to build such a tester, which aimed to be the world's first RFC-8219 [15] compliant lw4o6 tester.

Omar D'yab built a test-bed for lw4o6 [16], where the author demonstrated the operation of lw4o6 with its encapsulation/decapsulation mechanism.

As for the lwB4 router's implementation, Marcel Wiget [17] has built a complete and functioning lwB4 machine, where he used several Linux commands to build the NAT44 and IPv4-in-IPv6 tunnel. In addition, this proposed lwB4 network function is isolated into its own dedicated network namespace, which gives users the flexibility and the benefit of avoiding the use of a separate VM (virtual machine) [17].

Previous trials had been carried out to build lwB4 router using OpenWrt software [18]. However, they have proven to be complicated and not reliable [17].

IV. Lw4o6 Implementation

A. LwB4 Implementation

In our test-bed, we followed a similar approach to Marcel Wiget's lwB4 router implementation [17], where we used Linux commands, such as `ip -6 tunnel` and `ip route` to build the IPv4-in-IPv6 tunnel and `iptables` to implement NAPT44.

The full bash script to configure the lwB4 router is available through the "`lwB4.sh`" script in our public GitHub repository [19]. The main commands that we used to create a tunnel for encapsulation/decapsulation plus NAPT44, were as follows:

```
ip -6 tunnel add tun-lw4o6 remote 2001:db8:2::2 \
 local 2001:db8:0:1::2 mode ipip6
ip route add 192.0.2.0/24 dev tun-lw4o6 proto static
iptables -t nat -A POSTROUTING -p tcp -o tun-lw4o6 \
 -j SNAT --to 203.0.113.1
```

Below is an explanation for the IP addresses that we used in the commands above and illustrated in Fig. 3:

- 2001:db8:2::2 is the tunnel endpoint at lwAFTR side.
- 2001:db8:0:1::2 is the IPv6 address of lwB4.
- 192.0.2.0/24 is the network address of the IPv4 server.
- 203.0.113.1 is the public IPv4 address that will be used as source IP address by lwB4 when it forwards packets to lwAFTR through the softwire tunnel.

B. LwAFTR Implementation

Several software solutions were presented to build lwAFTR routers. It was featured in VPP [20] since version v16.09. However, VPP has demonstrated complexity in its configuration, and certain modules such as lw4o6 have become outdated and lack proper maintenance from developers. In contrast, Snabb software has received better maintenance and documentation [21]. Therefore, we have decided to deploy Snabb to build our lwAFTR router. Snabb in general is a toolkit that can be used for developing network functions in user-space, which means it bypasses the kernel to process network packets [21].

While Snabb can be used on Linux systems, it does not rely on the Linux kernel networking stack. Instead, it leverages technologies like Intel's Data Plane Development Kit (DPDK) or the Solarflare OpenOnload library to directly access the NIC (Network Interface Card) and perform packet processing in user space. By bypassing the kernel, Snabb aims to achieve lower latency and higher throughput [21].

Snabb divides the machine into two separate spheres:

- Internal interface, where IPv6 packets are received and processed.
- External interface, where public IPv4 packets are forwarded to the outside world.

Snabb resides in between those interfaces and creates a binding table. The core of Snabb's configuration is a file called "lwaftr.conf", which can be used while running the "`snabb lwaftr run lwaftr.conf`" command. The full configuration script of the lwAFTR router is available through the "`lwaftr.conf`" file in our GitHub repository [19].

For a better understanding of the packet translation, encapsulation, and decapsulation process of lw4o6, we follow the packet flow in the next Subsection. It is worth mentioning that the IP scheme which we used was based on documentation IP addresses because we used Snabb in a test environment that had internet access, while we did not want to cause any sort of routing conflict.

TABLE I
LwAFTR Binding Table

Public IPv4	PSID	PSID length	b4-IPv6
203.0.113.1	1	6	2001:db8:0:1::2

On the lwAFTR router, Snabb has a pre-configured binding table (see Table I), where it stores the relevant information

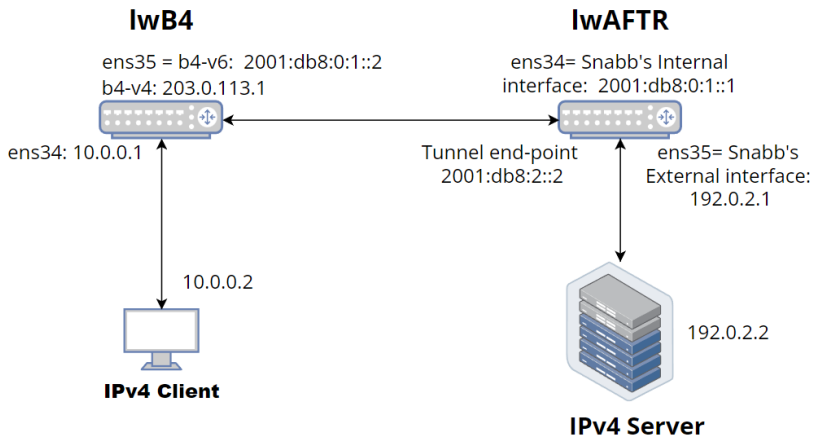

Fig. 3. Lw4o6 Testbed.

for every lwB4 router (every softwire), such as lwB4's public IPv4 address, IPv6 address, and the allocated port set for it.

The binding table is directly generated from the set of the configured softwires. It is never changed in response to data-plane traffic.

C. Packet Path Through the Lw4o6 Infrastructure

Fig. 3 shows the topology of our lw4o6 test-bed with its elements, where its operation can be summarized as below:

- The IPv4 client sends a packet with the following details:
  - Source IP address: Client IPv4 address (10.0.0.2)
  - Source port number: 5000
  - Destination IP address: IPv4 server address (192.0.2.2).
- The lwB4 receives the packet and performs the following steps:
  - NAPT44 function: the private source IPv4 address is replaced with a public IPv4 address (203.0.113.1). The source port number is replaced with an unused one from the range assigned to the subscriber. Assuming that the assigned range is 1024-2047, let the new source port number be 1050. At lwB4, the entry of the NAPT binding table is shown in Table II.
  - Encapsulation: lwB4 encapsulates the IPv4 packet into an IPv6 packet by prepending an IPv6 header to it and forwards the 4in6 packet to the lwAFTR through the softwire tunnel with the following details:
    - Source IP address: lwB4's IPv6 address: 2001:db8:0:1::2
    - Destination IP address: lwAFTR tunnel end-point IPv6 address: 2001:db8:2::2
    - Encapsulated IPv4 packet content:
      - Source IP address + port number: 203.0.113.1:1050
      - Destination IP address: 192.0.2.2
- The lwAFTR router receives the 4in6 packet with the same content as above. Snabb decapsulates the IPv4 packet and scans its binding table (Table I), then acts accordingly:
  - If a matching entry is found, then the IPv4 packet is forwarded to the IPv4 Internet via the external interface.
  - Otherwise the packet is dropped.
- Finally, the IPv4 packet arrives to the IPv4 server.

Packets in the reverse direction: IPv4 server ⇒ lwAFTR ⇒ lwB4 ⇒ IPv4 client, are processed as below:

- IPv4 server replies and sends the packet to lwAFTR with the following details:
  - Source IP address: 192.0.2.2
  - Destination IP address + port number: 203.0.113.1:1050
- LwAFTR receives the above packet on Snabb's external interface, then Snabb scans Table I, looking for a matching entry. Port number 1050 is a part of the port range [1024-2047], where PSID 1 refers to it explicitly. Therefore, we have a matching entry. Snabb encapsulates the IPv4 packet into an IPv6 packet using "b4-ipv6" as the IPv6 destination address and forwards the resulting 4in6 packet to the lwB4 router with the following details:
  - Source IPv6 address: 2001:db8:2::2 (tunnel end-point).
  - Destination IPv6 address: 2001:db8:0:1::2
  - Encapsulated IPv4 packet content:
    - Source IPv4 address: 192.0.2.2
    - Destination IPv4 address + port number: 203.0.113.1:1050
- LwB4 receives the above 4in6 reply packet and performs the following:
  - In case of incorrect parameters, such as port number, then the packet will be dropped immediately.
  - Decapsulates the IPv4 packet and scans its NAPT binding table (Table II) for a match. The lwB4 router then rewrites the destination IPv4 address and port

TABLE II
LWB4 ROUTER NAPT44 BINDING TABLE

Private IPv4	Source port	External IPv4	Temporary Port	Transport Protocol
10.0.0.2	5000	203.0.113.1	1050	TCP

TABLE III
VULNERABILITY OF DIFFERENT DFD ELEMENTS TO DIFFERENT THREATS [1]

	Spoofing	Tampering	Repudiation	Information Disclosure	Denial of Service	Elevation of Privilege
Data Flow		✓		✓	✓	
Data Stores		✓		✓	✓	
Processes	✓	✓	✓	✓	✓	✓
Interactors	✓		✓			

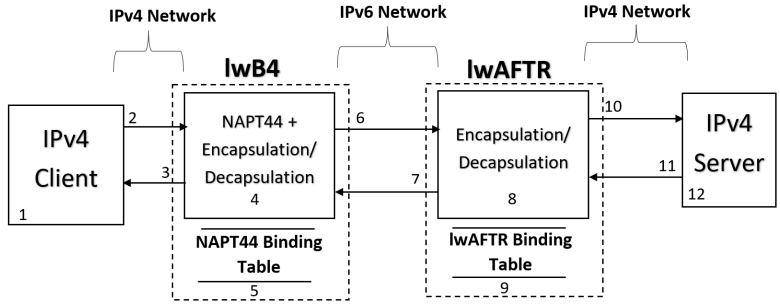

Fig. 4. Data Flow Datagram of Lw4o6

number of the packet based on the found entry to 10.0.0.2:5000, forwarding it to 10.0.0.2.

- IPv4 client receives the packet with the below details, which concludes the packet's journey:
  - Source IPv4 address: 192.0.2.2
  - Destination IPv4 address + port number: 10.0.0.2:5000

Snabb can have multiple softwires configured in its binding table and provisioned to communicate with multiple lwB4 routers [12]. One of Snabb's (or lw4o6 in general) challenges is the liability of lwAFTR to have millions of softwires (tunnels) configured [22]. on the other hand, the capacity of the binding table could be exhausted with an excessive number of entries.

V. SECURITY ANALYSIS

The lw4o6 technology plays a crucial role in enabling the coexistence of IPv4 and IPv6 networks during the transition phase. However, ensuring the security of lw4o6 deployments is of paramount importance to maintain the integrity and confidentiality of network communications. In this section, we conduct a comprehensive security analysis of the lw4o6 technology and evaluate its potential vulnerabilities and threats.

A. Threat Modeling

We have selected STRIDE to be our threat modeling technique, which can be used to assess the potential security vulnerabilities of any given IT system. It was explained by A. Shostack [8]. Below is a brief overview of the STRIDE components:

- Spoofing: an attacker's impersonation of legitimate nodes and pretending to be someone else by using an innocent IP address for example and sending harmful packets [8].
- Tampering: threats related to the modification of data or configuration in the network could allow an attacker to disrupt service or steal sensitive information [8].
- Repudiation: an attacker denies the responsibility of an act such as a DNS query or money transaction [8].
- Information Disclosure: threats related to the leakage of sensitive information from the network, such as IP addresses or routing information, could allow an attacker to track or compromise nodes [8].
- DoS: an attacker floods the targeted server with an excessive amount of packets that lead to resource depletion or disruption of service in the network and prevent legitimate users from communicating with the targeted machine [8].
- Elevation of Privileges: an adversary getting access to sensitive resources by bypassing some security control protocols that lead to him gaining unauthorized access or control over the network [8].

The type of vulnerabilities depends on what is being done with the data (processing, storing, etc..) [8]. Table III shows those vulnerabilities accordingly.

B. Applying STRIDE on lw4o6

According to [8], a DFD (Data Flow Diagram) of the examined system needs to be drawn for STRIDE to spot the potential vulnerabilities of the given system. Therefore, we

have drawn the required diagram as shown in Fig. 4, where the system has 12 vulnerable areas that can be targeted by an attacker.

C. Attacking Possibilities

We conducted a comprehensive analysis of all potential attacking scenarios on lw4o6 by leveraging its DFD using the STRIDE method. For a detailed examination, please refer to Appendix A.

D. Vulnerability Assessment

Upon conducting a comprehensive examination of the potential security threats associated with lw4o6, we have reached the conclusion that the exploitable threats accessible to an attacker can be succinctly summarized in Table IV. To facilitate a clearer understanding, we have classified the severity of these attacks into distinct levels, namely low, medium, high, and critical. The categorization is based on the detrimental impact inflicted upon the targeted system, as well as the intricacy involved in performing and mitigating the respective attack.

VI. LW4O6 TEST-BED

To build our test-bed, we used a "P" series node of NICT StarBED, Japan [23], which is a Dell PowerEdge 430 server with the following details: two 2.1GHz Intel Xeon E5-2683v4 CPUs with 16 cores each, 348 GB 2400MHz DDR4 SDRAM. We have installed a Windows 10 Pro operating system.

As shown in Fig. 3, our test-bed consists mainly of 4 machines created with Linux-based CentOS-7 Virtual machines built on top of VMware Workstation player virtualization software. Every machine has 8 GB RAM, 6 CPU cores (except lwAFTR which has 8 cores) and 20GB HDD.

As for our lwB4 and lwAFTR implementation, we already described it in details in Section IV-A and IV-B. Moreover, full configuration of lwB4 and lwAFTR (Snabb) is available in our GitHub repository [19].

VII. RESULTS

A. Normal lw4o6 Process

During pinging the IPv4 server from IPv4 client side (see Fig. 3), we monitored the traffic with the `tcpdump` command on different locations as shown in Table V, where the NAT44 + encapsulation / decapsulation on lwB4 side and encapsulation / decapsulation processes on lwAFTR side are quite obvious.

B. PSID Test

In our test-bed, we send packets from lwB4 with a specific port set [1024 - 2047]. As a result, we calibrated that on the lwAFTR side with the below details:

- PSID length : 6
- PSID : 1

Packets went through without any drop, however, when we changed the PSID value on lwaftr softwire configuration to 2, while keeping the port range on lwB4 the same, packets did not go through lwAFTR and an ICMP6 "Destination Unreachable" message was sent back to lwB4. The reason behind the packet drop is that PSID 2 refers to port set [2048 - 4095], while we kept sending packets using the old ports range [1024 - 2047].

C. Attacking Scenarios

1) DoS Attack: As shown in Fig. 5, an attacker machine was deployed to flood the lw4o6 infrastructure with too many TCP synchronization requests and thus perform a DoS attack against lwB4 and lwAFTR.

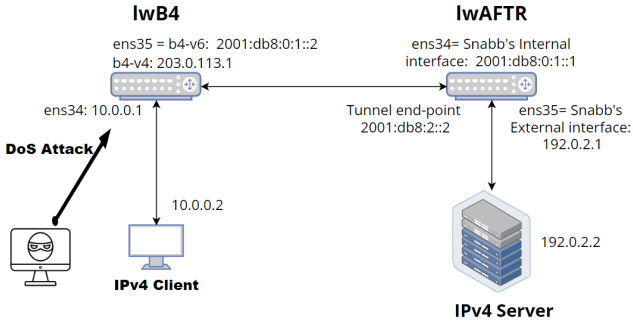

Fig. 5. DoS Attack Against Lw4o6 Infrastructure.

The attack was executed while the IPv4 client communicating with the IPv4 server normally, and it took around 3-5 seconds for the IPv4 client to show 75% packet loss. The attack was performed using `hping3` package:

```
hping3 -S --flood -V -p 80 192.0.2.2
```

In this situation, we explored a scenario where an intruder gains entry to the client's local network and engages in harmful actions aimed at overwhelming the access point. The objective is to obstruct the client(s) from receiving responses to their requests.

Moreover, Fig. 6, shows the CPU utilization of the lwB4 machine before and after the attack, where the CPU of the machine was fully utilized within 5 seconds.

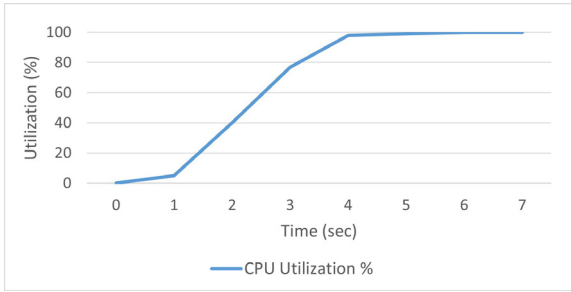

Fig. 6. CPU utilization for lwB4 Machine

2) Information Disclosure: As shown in Fig. 7, we carried out an information disclosure attack against the traffic between lwB4 and lwAFTR, where we used Scappy script to sniff the communication channel and print out the payload of the TCP/UDP packets. The attack was successful, as it printed out the content of the payload in plain text. The script can be found under the name of "`info-disclosure.py`" in our GitHub repository [19].

3) ICMP Spoofing: Since ICMP packets do not have port numbers, lw4o6 handles ICMP packets differently than it does with TCP /UDP packets, especially when it comes to packet filtering at lwB4 and lwAFTR routers. The solution

TABLE IV
SUMMARY OF THE POTENTIAL VULNERABILITIES OF LW4O6

Attack Name	Intricacy of Performing the Attack	Intricacy of Mitigation	Attack Impact (Severity)
TCP RST Signal	Average	Average	Low
IP Address Spoofing	Average	Difficult	Medium
Packet Injection	Average	Difficult	Medium
Information Disclosure	Average	Easy	Medium
Packet's Payload Tampering	Difficult	Difficult	Medium
ARP Poisoning	Average	Difficult	High
Source Port Exhaustion	Easy	Average	High
TCP Session Hijack	Easy	Average	Medium
Network Mapping	Easy	Easy	Low
DoS using TCP SYN Flood	Easy	Difficult	Critical

TABLE V
LW4O6 PACKET ENCAPSULATION/DECAPSULATION PROCESS

Packet capture at lwB4 ens34
17:22:15.047747 IP 10.0.0.2 > 192.0.2.2: ICMP echo request, id 21142, seq 1, length 64
17:22:15.054865 IP 192.0.2.2 > 10.0.0.2: ICMP echo reply, id 21142, seq 1, length 64
Packet capture at lwB4 ens35
17:22:15.047840 IP6 2001:db8:0:1::2 > 2001:db8:2::2: IP 203.0.113.1 > 192.0.2.2: ICMP echo request, id 1026, seq 1, length 64
17:22:15.054630 IP6 2001:db8:2::2 > 2001:db8:0:1::2: IP 192.0.2.2 > 203.0.113.1: ICMP echo reply, id 1026, seq 1, length 64 1
Packet capture at lwAFTR ens35
17:22:15.050075 IP 203.0.113.1 > 192.0.2.2: ICMP echo request, id 1026, seq 1, length 64
17:22:15.051435 IP 192.0.2.2 > 203.0.113.1: ICMP echo reply, id 1026, seq 1, length 64

was presented in RFC-7596 Section 8.1 [12], where the lwB4 router encapsulates a port number (out of the assigned pool of ports) into the ICMP ID field. Therefore, when the ICMP packet reaches the lwAFTR router, its ICMP ID field will be inspected and treated as the source port number, where the same process of packet filtering (Section IV-C) will be applied to the packet.

The attack was made possible by a script based on a powerful Python library called Scapy [24], which is an interactive packet manipulation program. The script can be accessed under the name of "icmp-spoofer.py" in our GitHub repository [19].

As illustrated in Fig. 8, the idea behind this attack is to sniff the communication channel for any ICMP packet being forwarded from the lwB4 towards lwAFTR, then send a crafted ICMP packet to the lwAFTR.

Before sending the crafted packet, Scapy makes sure that the new packet has similar details to the original one while altering the payload's content (the transmitted data), uses the same ICMP ID (for the sake of port mapping), and then forwards it to the lwAFTR.

The packet journey of the spoofed packet follows the below path:
Attacker ⇒ lwAFTR ⇒ IPv4 Server ⇒ lwAFTR ⇒ lwB4 ⇒

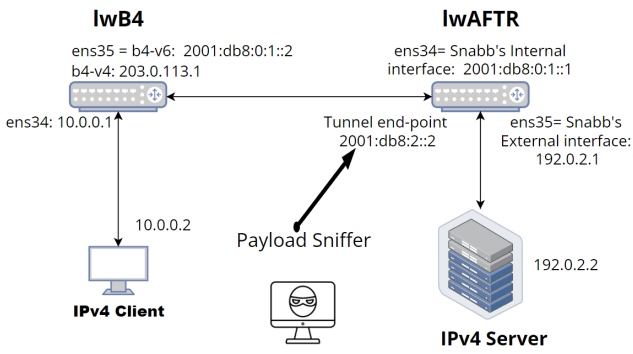

Fig. 7. Information Disclosure attack using Scapy script

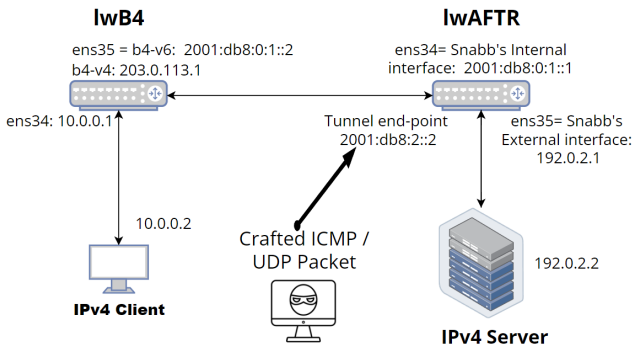

Fig. 8. ICMP / UDP packet Spoofing Attack.

IPv4 Client. As a result, the IPv4 client received a reply for a packet that it never sent.

4) UDP Spoofing: As shown in Fig. 8, We repeated the same attack but targeted UDP traffic. Therefore, we applied a different Python script, which can be accessed under the name of "`udp-spoofer.py`" in our GitHub repository [19]. The script sniffs the traffic between lwB4 and lwAFTR routers, looks for UPD traffic, and crafts new UDP packets based on the same details, especially the source port number. In addition, the newly crafted packet has the wrong payload (random text generated by the script). Eventually, the attacker machine forwards the crafted packet to the lwAFTR router. The attack was successful and the lwAFR router processed the malicious packet normally and forwarded it to the IPv4 server.

On the other hand, we reiterated the same attack with a minor adaptation. In contrast to the initial method of extracting the source port from the UDP packet, we devised a customized packet with a randomly generated source port number and subsequently directed it toward the lwAFTR. Consequently, the lwAFTR promptly discarded the packet owing to the absence of the appropriate source port. Further details and implementation of this script can be accessed in our GitHub repository under the file name "`random-source-port.py`" in our GitHub repository [19]

5) Source Port Exhaustion: Our lwB4 router was equipped with a specific port range: [1024-2047]. Therefore, we decided to exploit this vulnerability. As illustrated in Fig. 9, we used a tool called dns64perf++ that generates an excessive amount of DNS queries toward the IPv4 server [25]. DNS queries are UDP packets and they require a UDP port to be assigned for each packet. As a result, we managed to exhaust the pool ports in less than one second. Dns64perf++ tool generated 2500 packets/second, which can be found under the name of "`port-exhaust.sh`" in our GitHub repository [19].

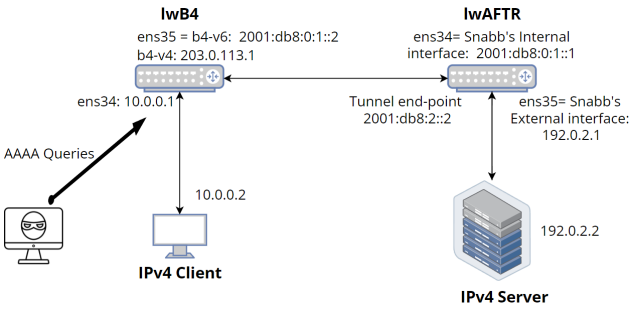

Fig. 9. Source Port exhaustion

Fig. 11 shows the last three lines of Wireshark capture on lwB4 ens35, where we managed to exhaust the source port pool [1024-2027] in less than one second. It also shows traffic stopped in less than half of a second and then resume at the 30th second. This was due to the default timeout of the UDP connection, where ports are re-assignable after 30 seconds.

It is worth mentioning that DNS64perf++ was designed to be used as a measurement tool, however, we used it as an attacking method.

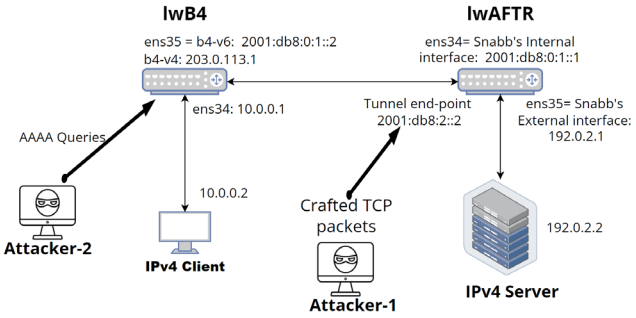

Fig. 10. TCP Session Hijacking Attack

6) TCP Reset Signal: The idea behind the attack is to send a TCP RST signal to the IPv4 server (via the lwAFTR router) and terminate an existing TCP connection with the IPv4 client. We achieved our goal by writing a Python script that sniffs the traffic between lwB4 and lwAFTR routers, searches for TCP packets with SYN and ACK flags, and crafts new TCP RST packet accordingly. The crafted TCP RST packet was forwarded by the attacker to the lwAFTR router, which forwarded the packet again to the IPv4 server. The crafted packet forced the IPv4 server to terminate the TCP connection with the IPv4 server (until the next TCP Sync packet comes again from the IPv4 client).

The script can be found under the name of "`lw-tcp-reset.py`" in our GitHub repository [19].

7) TCP Session Hijacking: TCP session hijacking attack is a cyberattack where an unauthorized party intercepts and takes control of an established TCP connection between two communicating entities, potentially gaining unauthorized access and control over the communication or data exchange [26].

We conducted this attack by employing the Scapy software to intercept communication between lwB4 and lwAFTR machines, as depicted in Figure 10. The attack is initiated when the "attacker-1" machine detects a TCP ACK packet moving from the IPv4 client to the IPv4 server. Upon identification of this packet, the attacker extracts pertinent information such as IP addresses, port numbers, sequence, and acknowledgment numbers from the exchanged TCP packets and sends crafted packets to the IPv4 server accordingly.

Subsequently, the "attacker-1" machine fabricates a TCP packet with the "`RST`" flag and relays it to the IPv4 client via the lwB4 machine in order to deceive the IPv4 client with a fake TCP session abortion signal. Furthermore, the attack script continuously monitors the communication channel, responding to relevant TCP packets and adapting responses with accurate information and appropriate flags.

Concurrently, the script establishes an SSH connection from the "attacker-1" to the "attacker-2" machine and initiates a DoS attack against the lwB4 router with the goal of exhausting its CPU resources. This action aims to obstruct any future communication between the IPv4 client and the IPv4 server via the lwB4 router. Consequently, the "attacker-1" machine gains the ability to communicate with the IPv4 server and hijacks the ongoing TCP session.

No.	Time	Src. IP	Dst. IP	Protocol	Src. port	Dst. port	Info.
1028	0.561305919	203.0.113.1	192.0.2.2	DNS	2045	53	Standard query 0x03fd AAAA 000-000-003
1029	0.562221123	203.0.113.1	192.0.2.2	DNS	2046	53	Standard query 0x03fe AAAA 000-000-003
1030	0.562592491	203.0.113.1	192.0.2.2	DNS	2047	53	Standard query 0x03ff AAAA 000-000-003
1046	30.010075937	203.0.113.1	192.0.2.2	DNS	1024	53	Standard query 0xd524 AAAA 000-000-213

Fig. 11. Wireshark Capture on lwB4 ens35.

Notably, the script is also designed to send meticulously crafted TCP packets containing the "PSH" and "ACK" flags (with falsified payloads) to the IPv4 server. This process is iterated each time the IPv4 server responds with a TCP packet bearing an "ACK" flag. The script can be found under the name "tcp-session-hijack.py" in our GitHub repository [19].

It's important to note that the use of the "PSH" and "ACK" flags is not mandatory in all TCP connections. By utilizing it, the server can ensure that data is delivered to the application as soon as possible, reducing latency, and improving the user experience. Results of the attack were made public under this path: "files/tcp-session-hijack-results.txt" in our GitHub repository [19], where it shows the full chain of TCP communication between the attacker and the IPv4 server via the lwAFTR machine.

8) Network Mapping: In order to expose the topology of lw4o6, we conducted an experiment with a simple attacking command:

```
traceroute 192.0.2.2
```

The result was as follows:

```
30 hops max, 60 byte packets
1 10.0.0.1 (10.0.0.1) 1.154 ms  0.739 ms  1.911 ms
2  192.0.2.2 (192.0.2.2)  36.101 ms  36.302 ms \
   36.627 ms
```

Such a result tells the attacker that there are two hops till he can reach the target. It also can help the attacker to map the network topology and reveals the network infrastructure, routers, and servers in between the source and target. This knowledge can aid in identifying potential points of entry or weak links in the network.

D. Attacks summary

In conclusion, by employing the STRIDE method to analyze lw4o6 security, our investigation has successfully established a link between STRIDE and actual testbed attack scenarios. We have identified and validated the following attack possibilities:

- A DoS attack (using TCP SYN flood) directed at the lwB4 machine, detailed in Appendix A4: (v).
- An Information Disclosure attack against the traffic between lwB4 and LwAFTR routers, detailed in Appendix A6: (ii).
- Two different types of Spoofing attacks targeting the lwB4 IPv6 address, detailed in Appendix A4: (i).
- A source port exhaustion attack against the lwB4 router, detailed in Appendix A2: (iii).
- A TCP RST signal attack against the lwAFTR router / IPv4 server, detailed in Appendix A6: (i).
- TCP session hijacking attack against the lwAFTR router, detailed in Appendix A6: (i).
- Network Mapping attack against the lwB4 router, detailed in Appendix A4: (iv).

E. Mitigation Methods

1) DoS Attack Mitigation: We mitigated the attack by deploying iptables rules on the lwB4 machine to perform a rate-limiting mechanism and setting a filter with two rules, one that allows packets to be forwarded with certain limits (10 p/s for instance), while the second rule drops everything else.

```
iptables -A FORWARD -p tcp --syn -m limit \
 --limit 10/s -j ACCEPT
iptables -A FORWARD -p tcp --syn -j DROP
```

2) Information Disclosure Mitigation: We have successfully mitigated the information disclosure attack, which we presented in Section VII-C2. We achieved that by encrypting the payload of TCP/UDP packets generated by the IPv4 client using the Python Fernet module. The script can be found under the name of "payload-encryptor.py" in our GitHub repository [19]. Therefore, after running the same attacking script, the actual payload of the packet was not visible, only its encrypted value.

3) ICMP Spoofing Mitigation: An advanced packet crafting software packages, such as Scapy [24], can create a very realistic crafted packet with almost the exact anticipated values (IP addresses, port numbers, Packet sequence, and even MAC addresses). Therefore, a sophisticated tool such as SNORT [27], which functions as IDS (Intrusion Detection System) and IPS (Intrusion Prevention System) as well, would be required. The tool performs a deep inspection mechanism, where it detects the suspicious (spoofed) packet and drops it eventually. SNORT proved to be complicated to configure. Therefore, we omitted it as it is out of our scope.

4) UDP Spoofing Mitigation: Please refer to the above Subsection (VII-E3).

5) Source Port Exhaustion Mitigation: To counter the attack, we implemented rate limiting for DNS queries, ensuring that only 100 packets per second are allowed. We achieved this by configuring "iptables" rules to drop incoming DNS query packets by default while permitting the specified rate (100 packets per second).

As a result, the pool of the allocated ports within the lwB4 router could not be exhausted with such a low rate of DNS queries. The complete script can be found under the name of "exhaust-mitigate.sh" in our GitHub repository [19].

6) TCP Reset Signal Mitigation: The mitigation proved to be very complicated, an IDS / IPS device is required for deep inspection, please refer to the above Sub-section (VII-E3).

7) TCP Session Hijack: Please refer to the above Subsection (VII-E3).

8) Network Mapping: There are several possible mitigation methods for such an attack. For example, disabling ICMP echo replies at the network perimeter or on specific routers, will prevent outsiders from using "`traceroute`" to discover the internal network structure. Moreover, network segmentation strategy can help protect against network mapping attacks and enhance overall network security. Network segmentation involves dividing a large network into smaller, isolated segments or sub-networks. Each segment is logically separated from the others and may have its own security policies, access controls, and communication rules [28].

VIII. CONCLUSION

Lw4o6 proved to be an optimization over DS-Lite, that reduces the overhead from lwAFTR. It also implements IPv4aaS (IPv4 as a service) in IPv6 only environment. The fact that it is stateless in the center of the network, made it possible to scale its routers capabilities, especially lwAFTR. Lw4o6 can be built using open-source software packages such as VPP, Snabb, etc. Moreover, Snabb proved to be reliable software to build lwAFTR over Linux-based virtual machines. Our test-bed was a success, where we performed the normal packet NAT44 operation + packet encapsulation/decapsulation smoothly. We managed to find some vulnerabilities using our attacking scenarios such as DoS, spoofing, Tampering and Information Disclosure, where we implemented and proposed some mitigation methods against them.

ACKNOWLEDGMENT

The authors thank Max Rottenkolber and Ian Farrer from Snabb's community for their valuable input regarding Snabb's configuration.

The authors thank Shuuhei Takimoto for the opportunity to use the NICT StarBED, Japan, which was used in our test-bed implementation.

REFERENCES

[1] G. Lencse, Y. Kadobayashi, Methodology for the identification of potential security issues of different IPv6 transition technologies: Threat analysis of DNS64 and stateful NAT64, Computers & Security 77 (2018) 397–411. DOI: 10.1016/j.cose.2018.04.012.

[2] A. Al-Azzawi, Towards the security analysis of the five most prominent IPv4aas technologies, Acta Technica Jaurinensis 13 (2) (2020) 85–98. DOI: 10.14513/actatechjaur.v13.n2.530.

[3] M. Bagnulo, P. Matthews, I. van Beijnum, Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers, RFC 6146, IETF (2011). DOI: 10.17487/RFC6146.

[4] M. Bagnulo, A. Sullivan, P. Matthews, I. Van Beijnum, DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers, RFC 6147, IETF (2011). DOI: 10.17487/RFC6147.

[5] M. Mawatari, M. Kawashima, C. Byrne, 464XLAT: Combination of Stateful and Stateless Translation, IETF RFC 6877, IETF (2013). DOI: 10.17487/RFC6877.

[6] A. Al-Azzawi, G. Lencse, Towards the Identification of the Possible Security Issues of the 464XLAT IPv6 Transition Technology., in: TSP, 2020, pp. 439–444. DOI: 10.1109/TSP49548.2020.9163487.

[7] A. AL-Azzawi, G. Lencse, Testbed for the Security Analysis of the 464XLAT IPv6 Transition Technology in a Virtual Environment, in: 2021 44th International Conference on Telecommunications and Signal Processing (TSP), IEEE, 2021, pp. 5–9. DOI: 10.1109/TSP52935.2021.9522598.

[8] A. Shostack, Threat modeling: Designing for security, John Wiley & Sons, 2014.

[9] A. Al-Azzawi, G. Lencse, Identification of the Possible Security Issues of the 464XLAT IPv6 Transition Technology, Infocommunications Journal 13 (4) (2021) 10–18. DOI: 10.36244/ICJ.2021.4.2.

[10] A. Durand, R. Droms, J. Woodyatt, Y. Lee, Dual-stack lite broadband deployments following IPv4 exhaustion, Rfc 6333, IETF (2011). DOI: 10.17487/RFC6333.

[11] A. Al-Azzawi, G. Lencse, Analysis of the Security Challenges Facing the DS-Lite IPv6 Transition Technology, Electronics 12 (10) (2023). DOI: 10.3390/electronics12102335.
URL https://www.mdpi.com/2079-9292/12/10/2335

[12] Y. Cui, Q. Sun, M. Boucadair, T. Tsou, Y. Lee, I. Farrer, Lightweight 4over6: An extension to the dual-stack lite architecture, Rfc 7596, IETF (2015). DOI: 10.17487/RFC7596.

[13] O. Troan, W. Dec, X. Li, C. Bao, S. Matsushima, T. Murakami, T. Taylor, Mapping of address and port with encapsulation (MAP-E), Rfc 7597, IETF (2015). DOI: 10.17487/RFC7597.

[14] A. Al-hamadani, G. Lencse, Design of a software tester for benchmarking lightweight 4over6 devices, in: 2021 44th International Conference on Telecommunications and Signal Processing (TSP), 2021, pp. 157–161. DOI: 10.1109/TSP52935.2021.9522607.

[15] M. Georgescu, L. Pislaru, G. Lencse, Benchmarking methodology for IPv6 transition technologies, Rfc 8219, IETF (2017). DOI: 10.17487/RFC8219.

[16] O. D'yab, G. Lencse, Testbed for the Comparative Analysis of DS-Lite and Lightweight 4over6 IPv6 Transition Technologies, in: 2022 45th International Conference on Telecommunications and Signal Processing (TSP), IEEE, 2022, pp. 371–376. DOI: 10.1109/TSP55681.2022.9851309.

[17] D. P. García, The B4 network function, accessed: 2022-08-15 (2018). URL https://blogs.igalia.com/dpino/2018/02/15/the-b4-network-function/

[18] OpenWrt Software, http://openwrt.org (2004).

[19] A. Al-Azzawi, Lightweight 4 over 6 test-bed (9 2022). URL https://github.com/ameen-mcmxc/lw4o6-automation

[20] FDio, VPP software (2016). URL https://github.com/FDio/vpp

[21] D. P. Garcia, Snabb explained in less than 10 minutes, accessed: 2022-09-04 (2017). URL http://blogs.igalia.com/dpino/2017/11/13/snabb-network-toolkit/

[22] D. Garcia, Lightweight 4-over-6: a compelling IPv4+IPv6 architecture, accessed: 2023-04-24 (2017). URL https://www.igalia.com/project/lw4o6

[23] Making a synthesis emulation in IOT ERA possible Starbed5 Project. StarBED5 Project website, accessed: 2022-09-05 (2023). URL https://starbed.nict.go.jp/en/equipment/

[24] P. Biondi, Scapy, Packet Manipulation Program, accessed: 2023-01-02 (2003). URL https://scapy.net/index

[25] B. Dániel, DNS64perf++ Measurement Tool (2016). URL https://github.com/bakaid/dns64perfpp

[26] Z. Qian, Z. M. Mao, Off-path TCP Sequence Number Inference Attack - How Firewall Middleboxes Reduce Security, in: IEEE Symposium on Security and Privacy, 2012, pp. 347–361. DOI: 10.1109/SP.2012.29.

[27] B. Caswell, J. Beale, Snort 2.1 intrusion detection, Elsevier, 2004. [28] N. Mhaskar, M. Alabbad, R. Khedri, A Formal Approach to Network Segmentation, Computers & Security 103 (2021) 102162. DOI: 10.1016/j.cose.2020.102162.

[29] S. Naval, V. Laxmi, M. Rajarajan, M. S. Gaur, M. Conti, Employing program semantics for malware detection, IEEE Transactions on Information Forensics and Security 10 (12) (2015) 2591–2604.

[30] C. L. Abad, R. I. Bonilla, An Analysis on the Schemes for Detecting and Preventing ARP Cache Poisoning Attacks, in: 27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07), IEEE, 2007, pp. 60–60. DOI: 10.1109/ICDCSW.2007.19.

[31] S. Deng, X. Gao, Z. Lu, X. Gao, Packet Injection Attack and Its Defense in Software-Defined Networks, IEEE Transactions on Information Forensics and Security 13 (3) (2018) 695–705. DOI: 10.1109/TIFS.2017.2765506.

Ameen Al-Azzawi received his MSc in Communication Engineering from Northumbria University, Newcastle, England in 2014. He is now a PhD student at the Budapest University of Technology and Economics, Budapest, Hungary. He has been working full-time on his research at MediaNets Laboratory in the Department of Networked Systems and Services since September 2019. His research focus is on IPv6 transition technologies and their security analysis.

Gábor Lencse received his MSc and PhD in computer science from the Budapest University of Technology and Economics, Budapest, Hungary in 1994 and 2001, respectively.
He has been working full-time for the Department of Telecommunications, Széchenyi István University, Győr, Hungary since 1997. Now, he is a Professor. He has been working part-time for the Department of Networked Systems and Services, Budapest University of Technology and Economics, Budapest, Hungary as a Senior Research Fellow since 2005.
His research interests include the performance and security analysis of IPv6 transition technologies.

APPENDIX

A. Lw4o6 Attacking possibilities using STRIDE

1) IPv4 Client:

(i) Spoofing: an attack spoofs the source IP address of the IPv4 client and misuses the possession of such source IP address by sending harmful packets towards the lwB4 router [8].

(ii) Repudiation: an attacker denies the responsibility of doing any sort of activity such as sending a malicious packet towards the lwB4 router or a packet with a spoofed IP address [8].

2) Data Flow from IPv4 Client towards lwB4 Router:

(i) Tampering: an attacker intercepts and alters the data being sent from the client to the lwB4 router such as changing the client's original sent information or manipulating an order placed on an e-commerce website [8].

(ii) Information Disclosure: an attacker intercepts and views sensitive information being sent by the client to the lwB4 router, such as viewing a client's credit card information or login credentials [8].

(iii) DoS: an attacker floods the lwB4 router with a large number of requests from the IPv4 client side (or multiple clients), causing the router to become overwhelmed and unable to process legitimate requests. This can cause the router to crash or become unresponsive and deny service to legitimate clients [8]. Moreover, an attacker might send an excessive amount of DNS queries in order to exhaust the allocated pool of ports for the lwB4 router.

3) Data Flow from lwB4 Router to IPv4 Client:

(i) Tampering: an attacker alters data sent from the lwB4 router to the client in order to disrupt or gain unauthorized access to the IPv4 client's system. For example, an attacker might modify a software update file sent from a server to the IPv4 client in order to include malware [29]

(ii) Information Disclosure: an attacker intercepts or otherwise gains access to sensitive information sent from a server to the IPv4 client. For example, an attacker might use a man-in-the-middle attack to intercept and read login credentials sent from a server (via lwB4 router) to the IPv4 client in clear text [8].

(iii) DoS: an attacker floods the IPv4 client with an excessive amount of requests from the lwB4 router side, causing the client to become unavailable to legitimate incoming requests. For example, an attacker might launch a distributed denial of service (DDoS) attack against the client in order to disrupt its access to a specific website [8].

4) The lwB4 Router:

(i) Spoofing: an attacker spoofs the source IP address of the lwB4 router (impersonates it) and initiates communication with neighbouring (or remote) devices, while sending all sorts of malicious packets, that could harm the reputation of the organization that operates the router itself [8]. Moreover, another attack scenario is possible such as the potential for an Address Resolution Protocol (ARP) cache poisoning attack. The attack involves the interception of network traffic between the lwB4 router and the lwAFTR router, enabling an attacker to exploit the situation. By sending deceptive ARP messages to the lwB4 router, the attacker can assume the identity of the lwAFTR router. Consequently, the lwB4 router updates its ARP cache with the attacker's Media Access Control (MAC) address, erroneously associating it with the lwAFTR router. Once the ARP cache has been successfully poisoned, the attacker gains the ability to intercept, manipulate, or extract sensitive information from the network traffic. This form of attack poses a significant threat, capable of compromising the security of the system [30].

(ii) Tampering: an adversary might alter the actual data that is being processed or stored within the router such as IP addresses, port numbers, TTL (Time To Live) values, etc. It may lead to redirecting the packet to a malicious server and prevent the legitimate recipient of the packet from getting a response to his request [8].

(iii) Repudiation: after spoofing the source IP address of the lwB4 router and sending harmful packets, the attacker will be able to deny the fact that he was behind those suspicious packets [8].

(iv) Information Disclosure: an attacker having unauthorized access to confidential data within the lwB4 router such as packet's payload or the routing table of the lwB4 router itself [8]. Another type of information disclosure attack can be performed by an attacker gaining access to the network topology and the number of hops to reach a specific target.

(v) DoS: an attacker might target the lwB4 router with the well-known DoS attack. This can be done either from the IPv4 client side or from the lwAFTR side, where a huge amount of useless packets can be sent to the lwB4 router with the aim to overwhelm its processing power [8], please refer to Appendix A3.(iii).

(vi) Elevation of Privileges: an adversary bypasses the authority matrix within the organization to gain access (such as read/write permission) and therefore pushes a destructive configuration to the lwB4 router, which affects the whole network [8].

5) The NAPT44 Binding Table of lwB4 Router:

(i) Tampering: an attacker could potentially tamper with the NAPT44 binding table (see Table II) by altering the information stored in it, such as the IP addresses, ports, or protocol information. This causes the lwB4 to misroute or block legitimate traffic, leading to a communication breakdown [8].

(ii) DoS: an attacker could launch a Denial of Service attack on the NAPT44 binding table by overloading it with a large number of fake or malformed connection entries. This could cause the table to become full, preventing the lwB4 router from tracking legitimate connections and resulting in a communication breakdown [8].

6) Data Flow from lwB4 to lwAFTR:

(i) Tampering: an attacker has the capability to introduce a malevolent packet into the network traffic, leading to detrimental consequences for the lwAFTR. This act can be classified as a packet injection attack [31]. As an example, the malicious packet could manifest as a TCP RST packet, which promptly terminates an existing TCP connection. Moreover, the attacker might hijack the TCP session by eavesdropping on the communication between the lwB4 and the lwAFTR, then start communicating with the IPv4 server via the lwAFTR.

(ii) Information Disclosure: an attacker intercepts and views sensitive information, such as the TCP/UDP packet's payload, please refer to Appendix A2: (ii).

(iii) DoS: an attacker floods the lwAFTR router with useless packets to overwhelm it, please refer to Appendix A2: (iii).

7) Data Flow from lwAFTR to lwB4:

(i) Tampering: an attacker performs a man-in-the-middle attack and alters the packet's details such as the destination IP address, which will re-direct the packet to a potentially malicious server and deprive the legitimate lwB4 router of the response it was anticipating.

(ii) Information Disclosure: as the attacker performs a man-in-the-middle attack, he also exposes the content of the sent data, which is a data confidentiality breach.

(iii) DoS: an attacker floods the lwB4 router with an excessive amount of packets in order to overwhelm its computation power, please refer to Appendix A2: (iii).

8) The lwAFTR Router:

(i) Spoofing: an attacker might impersonate the lwAFTR router and initiate a communication with the IPv4 Server or the lwB4, which will make all of those network elements liable to the risk of exchanging sensitive data with an adversary [8].

(ii) Tampering: an attacker alters the content of the sensitive data within the lwAFTR router, such as the provisioned PSID value assigned for a specific lwB4 router, please refer to Appendix A4: (ii).

(iii) Repudiation: the packet sender hides his own identity, please refer to Appendix A4: (iii).

(iv) Information Disclosure: an attacker getting access to confidential data inside the lwAFTR router, such as packet's payload [8], please refer to Appendix A4: (iv).

(v) DoS: an attacker can exhaust the computation power of the lwAFTR router by sending too many useless packets, which will prevent it from being able to process any incoming packets from the lwB4 router.

(vi) Elevation of Privileges: an attacker getting high privilege access to the lwAFTR router such as admin permission [8]. Such attacks happen most of the time due to inside job [1].

9) LwAFTR Binding Table: This table stores all of softwires (tunnels) configured into the lwAFTR, see Table I.

(i) Tampering: an attacker alters the content of an entry within the table such as PSID value or lwB4's public IPv4 address. Such changes will pave the way for a malicious spoofed packet to be processed by the lwAFTR, while dropping the legitimate requests.

10) Data Flow from lwAFTR Router towards IPv4 Server:

(i) Tampering: please refer to Appendix A3: (i)

(ii) Information Disclosure: please refer to Appendix A3: (ii)

(iii) DoS: please refer to Appendix A3: (iii)

11) Data Flow from IPv4 Server towards lwAFTR Router:

(i) Tampering: please refer to Appendix A2: (i).

(ii) Information Disclosure: please refer to Appendix A2: (ii).

(iii) DoS: please refer to Appendix A2: (iii).

12) IPv4 Server:

(i) Spoofing: please refer to Appendix A1: (i).

(ii) Repudiation: please refer to Appendix A1: (ii).