electronics

MDPI

Article

Analysis of the Security Challenges Facing the DS-Lite IPv6 Transition Technology

Ameen Al-Azzawi * and Gábor Lencse

Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary; lencse@hit.bme.hu

* Correspondence: alazzawi@hit.bme.hu

Abstract: This paper focuses on one of the most prominent IPv6 transition technologies named DS-Lite (Dual-Stack Lite). The aim was to analyze the security threats to which this technology might be vulnerable. The analysis is based on the STRIDE method, which stands for Spoofing, Tampering, Repudiation, Information Disclosure, and Elevation of Privilege. A testbed was built for the DS-Lite topology using several virtual machines, which were created using CentOS Linux images. The testbed was used to perform several types of attacks against the infrastructure of DS-Lite, especially against the B4 (Basic Bridging Broadband) and the AFTR (Address Family Transition Router) elements, where it was shown that the pool of source ports can be exhausted in 14 s. Eventually, the most common attacks that DS-Lite is susceptible to were summarized, and methods for mitigating such attacks were proposed.

Keywords: DS-Lite; 464XLAT; DNS; IPv4aaS; STRIDE; translation; encapsulation; spoofing; thc-ipv6 toolkit; port number exhaustion

check for updates

Citation: Al-Azzawi, A.; Lencse, G. Analysis of the Security Challenges Facing the DS-Lite IPv6 Transition Technology. *Electronics* **2023**, *12*, 2335. https://doi.org/10.3390/electronics12102335

Academic Editors: Vladimir Laslo Tadić and Peter Odry

Received: 21 April 2023
Revised: 16 May 2023
Accepted: 19 May 2023
Published: 22 May 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the depletion of the public IPv4 address pool in 2011 [1], the integration of IPv4 and IPv6 has become a pressing issue in the field of networking. Various transition technologies have been proposed to address this challenge, but each comes with its own set of drawbacks and vulnerabilities. In previous research work, a survey of the most prominent IPv6 transition technologies was conducted [2], in which it was pointed out that the combination of DNS64 [3] and NAT64 [4] could be a working solution for the communication of IPv6 clients with IPv4 servers.

However, this technology still faces challenges in terms of IPv4 literals communications and establishing connections from the IPv4 host side. To address these challenges, the 464XLAT technology has been developed to tackle some of the DNS64 + NAT64 issues [5]. 464XLAT uses a double translation mechanism by deploying two separate translators: CLAT (client-side translator) and PLAT (provider-side translator).

Several papers have been published regarding the 464XLAT transition technology. In [6], a security analysis for 464XLAT using the STRIDE method was presented, which highlighted the vulnerabilities and potential security threats of the technology. In [7], a testbed of 464XLAT was built using Debian-based virtual machines to evaluate the performance of the PLAT under a DoS (Denial of Service) attack, specifically testing the CPU performance and the pool of source port numbers. In a subsequent paper [8], the previous work was extended using a more powerful computer, allowing for an increase in the number of attacking clients (virtual machines) from 1 to 8. The PLAT performance was then tested after an attack using the hping3 command, ultimately demonstrating the susceptibility of the PLAT to DoS attacks. These studies underscore the importance of further investigating the potential security threats and vulnerabilities of 464XLAT and other IPv6 transition technologies.

On the other hand, DS-Lite (Dual-Stack Lite) has its own unique topology and application. It is a promising technology that enables the ISPs (Internet service providers) to use only IPv6 in their access and core network, while providing the users with fully functional IPv4 Internet access, too. DS-Lite has the highest deployment rate among the five important IPv4aaS (IPv4-as-a-Service) technologies (464XLAT, DS-Lite, Lw4o6, MAP-E, MAP-T) [9]. The literature is very scarce when it comes to security threat analysis for DS-Lite, which is why the security analysis of DS-Lite was chosen as the topic of the current research.

The main target of this paper is to analyze the security threats that the DS-Lite IPv6 transition technology might face and to come up with mitigation methods for such attacks. The goals of the research are intended to be accomplished through the following steps:

- Applying the STRIDE threat modeling technique to DS-Lite, where some of its potential security threats were examined at every inbound and outbound gateway within the infrastructure;
- Building a testbed for the DS-Lite topology;
- Testing the real capabilities of DS-Lite under several kinds of attack scenarios;
- Coming up with potential mitigations of the tested attacks.

In Section 2, the related work is discussed. In Section 3, the operation of DS-Lite and its structure is explained, while Section 4 is devoted to tunneling in general and its security concerns. Section 5 discusses the operation of the STRIDE method, its elements, and how it works. In Section 6, the STRIDE method is applied to the DS-Lite topology. In Section 7, the testbed is built and its infrastructure, topology elements, attacking scenarios, and possible mitigation methods are explained. In addition, Section 7 also points out the importance of the research and narrows down potential areas for further research. In Section 8, the results of the study are summarized and concluded. In the same section, the efficiency of DS-Lite and its flexibility to deal with all connection scenarios are proven, especially when a private IPv4 address client wants to communicate with a public IPv4 address server while there is an IPv6 address island in the middle. Finally, the security analysis of DS-Lite and its vulnerabilities and their mitigation methods are summarized.

2. Related Work

Very few experiments have been published regarding DS-lite and its security analysis. Therefore, we seized the opportunity to take the lead in this uncharted territory. A survey of the most prominent IPv6 transition technologies and their security analysis was carried out in [9], where DS-Lite was mentioned, and its security analysis has been classified as important but replaceable due to several issues mentioned by [10], such as the following:

- The need for two separate physical interfaces at the AFTR;
- The need for high scalability at the AFTR side due to the fact that many B4 routers may be connected to the same AFTR [10];
- The location of deploying AFTR router within the ISP network and the trade-off it creates between the high operation cost and installing an extremely powerful AFTR [10]. The trade-off can be explained by dividing the issue at hand into two options:
    - o Deploying AFTR at the edge of the network to cover a small area serves few B4s and requires less-powerful AFTR;
    - o Deploying AFTR at the core of the network to cover a big area covers more B4s and requires extremely powerful AFTR (or even more than one AFTR);
- The complexity of deploying a proxy DNS resolver, which will proxy every DNS query stemming from all IPv4 clients heading towards a DNS server that resides in an IPv6 network [10].

Another work [11] conducted a security analysis for DS-Lite in terms of its MIB (Management Information Base), and it consists of several objects. MIB is a module that can be used to monitor the AFTR router within the DS-Lite infrastructure by leveraging SNMP (Simple Network Management Protocol). According to [11], the most vulnerable objects that are susceptible to attacks are:

- Notification threshold objects: an attacker manipulating a threshold's value to a very low level, which will lead to a flood of useless alarms and thus disrupt the AFTR and its monitoring mechanism, or the attacker sets it to a very high level that makes the idea of setting the alarm literally useless:
  - o DsliteAFTRAlarmConnectNumber: The alarm is sent when the number of current DS-Lite tunnels reaches the threshold, which means for every B4 router, the AFTR has to have a separate tunnel for it;
  - o DsliteAFTRAlarmSessionNumber: An alarm will be sent when the threshold of sessions per IPv4 user is reached. This metric goes hand in hand with RFC-6333 [12], where AFTR has to be able to log softwire-ID, IP, ports, and protocol (see Table 1) in order to keep track of user sessions;
  - o DsliteAFTRAlarmPortNumber: An alarm is to be sent when the threshold for the number of ports used by a user is reached or even crossed;
- Table entry objects: An attacker can alter the content of such entries causing the drop of legitimate entries or adding harmful and faulty ones:
  - o DsliteTunnelTable: Consists of mapping entries of B4 address to AFTR address;
  - o DsliteNATBindTable: Contains entries about the current active bindings within the NAT table of the AFTR (see Table 1).

Table 1. Dual-Stack Lite Carrier-Grade NAT translation table [12].

Softwire-ID/IPv4/Protocol/Port	IPv4/Protocol/Port
2001:db8:0:1::2/10.0.0.1/TCP/10000	192.0.2.1/TCP/5000

What makes these table entries a potential security threat is the possibility of an attacker assessing the number of hosts being served by a single AFTR router, which reveals sensitive information about the whole topology of DS-Lite [11]. A chance of an inside job is also possible, where an internal employee can access the list of hosts that are in active sessions, which will be a violation of the subscriber's privacy [11]. Moreover, RFC-6334 [13] referred to DS-Lite security briefly and recommended that an IP firewall be implemented in order to avoid MitM (Man-in-the-Middle) attacks along the softwire connection of DS-Lite.

RFC-8513 [14] proposed another method to implement DS-Lite using the YANG module, which is a schema that facilitates data assessment mechanisms through network management protocols such as NETCONF and RESTCONF. This module allows the administrator to add some features and add-ons to the B4 and AFTR interfaces such as "b4-address-change-limit", "Tunnel-MTU (Maximum Transmission Unit)", etc.

Moreover, RFC-8513 [14] suggested a solution for the DoS attack by raising the "b4-address-change-limit". This value specifies the minimum time between two consecutive changes in the IPv6 address of the B4 device. Setting it to a low value would enable the attacker to send a higher number of attacking packets with different source addresses. The recommended mitigation is to set its value to 30 min.

In RFC-8513 [14], the authors presented the security analysis of their DS-Lite architecture, which emphasized that the main vulnerability is an attacker having access to either B4 or AFTR router and undertaking several kinds of malicious activities:

- Manipulating the AFTR IPv6 address on the B4 tunnel endpoint, which will deceive the B4 router and force it to forward the 4in6 traffic to the wrong recipient;
- Altering the value of the "b4-address-change-limit", which gives the B4 more flexibility in configuring the softwire. An attacker lowering this value will boost the possibility of a DoS attack against the B4 router.

Similar research was conducted on vehicular network security, where the author investigated the possibility of preventing a selfish or malicious user from occupying limited resources in a mobile edge network and proposed a trusted deep reinforcement learning (DRL) cybersecurity approach [15]. The author also presented the idea of a reputation record

table, which contains a list of untrusted interference vehicle devices [15]. Furthermore, another research work proposed a Truth Detection-based Task Assignment (TDTA) scheme to assign micro-tasks to reliable workers and establish a credible task execution environment for crowdsourced industrial Internet of Things (IoT) [16].

3. The Operation of DS-Lite

The general purpose of DS-Lite is to provide the home network with IPv4 connectivity across an IPv6-only access network. DS-Lite was presented in RFC 6333 [12], where it consists of two main parts (B4 and AFTR). Figure 1 illustrates the infrastructure of DS-Lite and how it functions (with some simplification, please refer to the operation of the AFTR below).

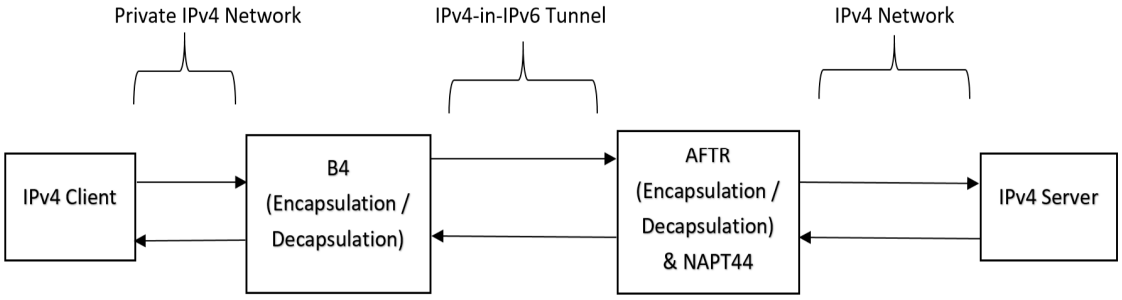

Figure 1. Overview of the DS-Lite architecture [17].

3.1. B4 (Basic Bridging Broadband)

This router is responsible for encapsulating IPv4 packets into IPv6 ones and then sending them over the IPv6 network until they reach the AFTR router. It also has another vital role when it processes the returning packets from the AFTR toward the B4, where it decapsulates the IPv6 packet and extracts the original IPv4 packet from the payload [12]. The tunnel that is created by the CPE (customer-provided equipment), which is the B4, is called the softwire tunnel that connects the B4 with the AFTR.

Some facts about the B4 router that were mentioned by RFC-6333 [12]:

- B4 announces itself as the default IPv4 router, and this route is applicable for all IPv4 clients sitting behind this B4 router;
- The B4 router should also announce itself as a DNS server in the DHCP (Dynamic Host Configuration Protocol) option 6 [12];
- As for the operation side, it acts as a DNS proxy for all IPv4 clients who are willing to connect with it and beyond, it forwards those requests to the DNS server of the ISP (Internet Service Provider) [12];
- B4 also forwards native IPv6 packets to the AFTR without any intervention;
- The default structure of DS-Lite is as explained above (client → CPE → ISP). However, some devices (IPv4 clients) are connected directly to the ISP without the home gateway. The reason behind this is that those devices have the ability to act as CPEs themselves [12].

3.2. AFTR (Address Family Transition Router)

The AFTR decapsulates the 4in6 traffic that comes from the B4 router; then, it translates the IPv4 packet into public IPv4, which is a function similar to the well-known stateful NAT44 [12]. However, it is a more complex function than stateful NAT44, because here the translation table contains also the Softwire-ID, which is the IPv6 address of the CE device, as shown in Table 1. Of course, it also performs the reverse functions for the returning packets.

The technology in general has two types of topologies (Gateway-based and Host-based architecture), and each one has its own applications:

- Gateway-based: Mostly based in residential broadband infrastructure, where the client and the CPE are based in different but directly connected machines [12];
- Host-based: Designed for large-scale deployments and especially when the client is directly connected to the service provider network, which means the IPv4 client and the CPE are both mounted on the same device [12].

In this paper, the focus is on the Gateway-based architecture topology because this topology is easier to build for testing purposes and a virtual environment. It is the most represented topology in residential households; for example, IPv4 client >> router >> ISP.

The AFTR has two main interfaces (softwire interface and network interface), and each one of them faces one end of the device:

- Softwire interface: Its main function is connecting B4 with AFTR and translating the datagram of (softwire identifier + IPv4 + source port) to another source IPv4 and source port. So, it decapsulates the IPv4 in the IPv6 datagram (for the packets coming from B4 side), and performs the reverse by encapsulating IPv4 in the IPv6 datagram (for the packets heading towards the B4 side);
- Network interface: Resides on the other side of the AFTR—the WAN (wide-area network) side—and translates the decapsulated source IPv4 and source port into the interface IP (192.0.2.1) and source port 5000 (for example purposes only). The interface is also in charge of the translation in the reverse direction (IPv4 server → AFTR → B4), where it translates the original packet's destination IP and destination port (according to AFTR's Carrier-Grade NAT translation table) to the destination IP and destination port of the softwire.

Figure 2 shows the outbound communication between B4 and AFTR. The NAT table in this example is configured in a way that translates any incoming packet with the source IP address 10.0.0.1 and source port number 10,000 to IP/port pair of 192.0.2.1:5000.

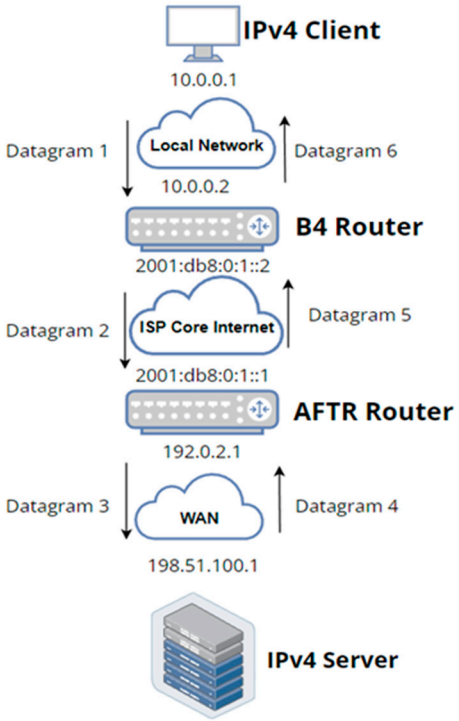

Figure 2. Overview of the DS-Lite datagram path.

- Once Datagram 1 reaches the B4 router, it will be encapsulated into Datagram 2, which is an IPv4 in IPv6 datagram, and then forwarded to the AFTR through the softwire tunnel;

- When the AFTR receives Datagram 2, it decapsulates it, extracts the IPv4 datagram from it, and forwards it to the stateful NAT function, which performs the following (according to its NAT table);
- The received datagram with source IP and port pair of 10.0.0.1:10000 should be translated to Datagram 3 with the following specifications:
    - Source IP and port pair: 192.0.2.1:5000.

How does the AFTR translation table function?

The IPv6 address for the B4 router is called the softwire-ID. This ID is being shared by every client that wants to connect with the B4 router and triggers the DS-Lite communication process. Every single client of those is equipped (by the B4 router) with a source IPv4 address [RFC 1918] such as 10.0.0.1, and it is unique within its network.

When an embedded packet (source IP address: 10.0.0.1, source port number: 10,000) is forwarded from B4 to AFTR through the softwire tunnel, AFTR combines the softwire-ID with this packet's details as one entry: softwire-id/IPv4/TCP/10000. In fact, this is only half of the entry. The other half will be the IP address of the AFTR network interface, the protocol identifier (e.g., TCP or UDP), and the source port number (192.0.2.1/TCP/5000), as shown in Table 1. The reverse direction of the packet (IPv4 server > AFTR > B4 > IPv4 client) functions in a similar manner:

- AFTR receives Datagram 4 through the network interface with a public IPv4 address, then processes it by checking the internal NAT table and looking for matching entries. In this case, Datagram 4 has the following details:
    - IPv4 destination address: 192.0.2.1;
    - Destination port: 5000;
    - Protocol is TCP;
- The corresponding entry does exist in the NAT table (see Table 1); AFTR, therefore, translates the IPv4 packet of Datagram 4 to IPv4 destination address 10.0.0.1 and TCP destination port 10,000;
- AFTR then encapsulates the translated IPv4 packet into an IPv6 packet and sends Datagram 5 over to the B4 router at 2001:db8:0:1::2 IPv6 address (which the AFTR knows from the table entry itself as softwire-ID);
- B4 receives Datagram 5, decapsulates the embedded packet, extracts the original IPv4 packet from the 4in6 Packet, and forwards it accordingly to the IPv4 client (10.0.0.1:10000) as Datagram 6.

Table 2 lists all IP addresses and port numbers for the DS-Lite packet route for Datagrams 1–6.

Table 2. DS-Lite datagrams.

Datagram	Header Details
IPv4 Datagram 1	IPv4 Src: 10.0.0.1 IPv4 Dst: 198.51.100.1 TCP Src: 10000 TCP Dst: 80
IPv6 Datagram 2	IPv6 Src: 2001:db8:0:1::2 IPv6 Dst: 2001:db8:0:1::1 IPv4 Src: 10.0.0.1 IPv4 Dst: 198.51.100.1 TCP Src: 10000 TCP Dst: 80
IPv4 Datagram 3	IPv4 Src: 192.0.2.1 IPv4 Dst: 198.51.100.1 TCP Src: 5000 TCP Dst: 80

Table 2. *Cont.*

Datagram	Header Details
IPv4 Datagram 4	IPv4 Src: 198.51.100.1 IPv4 Dst: 192.0.2.1 TCP Src: 80 TCP Dst: 5000
IPv6 Datagram 5	IPv6 Src: 2001:db8:0:1::1 IPv6 Dst: 2001:db8:0:1::2 IPv4 Src: 198.51.100.1 IPv4 Dst: 10.0.0.1 TCP Src: 80 TCP Dst: 10000
IPv4 Datagram 6	IPv4 Src: 198.51.100.1 IPv4 Dst: 10.0.0.1 TCP Src: 80 TCP Dst: 10000

To understand the functionality of AFTR translation even more, please refer to RFC 6333 [12], where further details about the translation steps are explained.

4. Tunneling

4.1. Tunneling History

Tunneling goes back to the 1990s when PPTP (point-to-point tunneling protocol) was presented by Microsoft engineers and some other vendors, where the original purpose was to support Windows users with data encryption, which gained huge popularity among small and medium-sized corporations later on and is in fact still being used to some extent [18]. Since then, it has consistently faced scrutiny from critics. It was even presented as an RFC [19] in 1999. PPTP functions at Data Layer 2 and uses general routing encapsulation (GRE) as a packet creation system, where GRE encapsulates the original packet inside another packet. Some vulnerabilities were discovered by Schneier [20]; he showed that PPTP has several weak points such as its Challenge/Response authentication protocol (CHAP), and he also presented the fact that PPTP uses a very low-security aware hashing algorithm, where eavesdropping was also quite easy to implement. Schneier [20] also emphasized that PPTP uses a low-security encryption standard called MPPE, which has a key that could be easily broken.

Later, Microsoft presented another solution called MS-CHAPv1 [21] and then MS-CHAPv2 [22], which is used as one authentication option in Microsoft's implementation of the PPTP protocol for virtual private networks [21].

The most secure tunneling is OpenVPN and it has proven to be very hard to crack [18]. In conclusion, there are alternatives to the regular PPTP such as the IKEv2 and a combination of IPsec and L2TP, which is used in the Microsoft VPN (virtual private network), and it is safer and quicker than regular PPTP [18]. Tunneling in general can come in so many forms such as MAP-E, 6over4, 6to4, Teredo, 6rd, DS-Lite, etc. [9]. It makes the connection between two devices possible despite having a different IP version island in the middle, as is the case in the DS-Lite tunnel example, though it brings some security threats with it.

4.2. Tunneling Issues and Solutions

The most serious issue with tunneling is that tunneled IP traffic does not go through the same inspection process that the normal traffic (non-encapsulated packets) will be liable to unless extra dedicated devices are installed on the premise to double-check the traffic as deep packet inspection [23]. For instance, in the Teredo tunnel, the router will check the IP and UDP layer normally. However, it cannot find out that there is actually another IP layer encapsulated within the UDP payload.

Another issue with tunneling is when the already encapsulated packet is targeting a host that lies further beyond the tunnel endpoint. In this case, the machine will normally forward the packet simply to the built-in next hop [23].

The tunneled data endpoints are aware of the tunneled data and the encapsulated packets. The network devices in the middle are not, and that could be an issue [23].

A solution is proposed in [23], where in case IPv6 transition is required, native IPv6 is the way forward in terms of connecting two sides with tunneling solutions, such as ISATAP, 6over4, etc. [24], in order to encapsulate traffic between a device and the router on the other side that resides in the same network. The difficulties that come with inspecting the inner content of the encapsulated packet make it a quite complicated process.

NAT process also presents another challenge when it comes to tunneling, opening more doors for attacking possibilities on the incoming NAT interface. Therefore, it is recommended that the NAT interface should not be configured by default and only used when it is the last resort [23]. Another recommendation is to deactivate the interface itself after its usage [23].

Furthermore, IP address guessing emerged as a potential risk to the tunnel endpoint, where some protocols use a regular or a well-known IP address or range of IP addresses. For instance, Teredo uses a specific IP address range in its infrastructure, which makes the tunnel liable to IP address guessing attacks. Furthermore, sometimes guessing the IP address gives an indication about which kind of OS is being used. For instance, Teredo implies that the machine is most probably running Microsoft Windows. The solution is to avoid those well-known IP address ranges and use random ones [23].

On the other hand, an adversary can have the ability to alter the tunnel's server settings on the client side while the client itself has no idea about it. One way to avoid such a breach is to use authentication for tunnel endpoints such as https [23]. A second mitigation method can be the use of secure ND (neighbor discovery) [24] whenever a client receives a router advertisement packet. Tunnels, in general, are less secure than normal conventional links due to the fact that an attacker can send an already encapsulated packet to the tunnel end-node where it is supposed to be decapsulated [23]. This action might cause an injection of the faulty packet into the decapsulator side. This threat might be avoided by turning on a decapsulation check, which will drop such malicious packets [25]; it is also highly discouraged to set the tunnel interface to reply by acknowledging the existence of a tunnel, such as an "ICMP error message", representing another valuable recommendation.

In general, [26] has mentioned two important general tunnel security points:

- There is no correlation between the amount of security regulations and procedures on the tunnel packets and the level of security measures applied to the original packet inside the tunnel (the payload) [26];
- The security header authentication (HA) or encryption security payload (ESP) parameters are the deciding factors for the three security pillars (integrity, authentication, and confidentiality) of any tunnel endpoint [26].

4.3. Attack Scenarios

In [27], the attack possibilities were categorized into three types:

- Denial of Service (DoS): The act of sending too many requests to a specific device to overwhelm its computation and processing power or bandwidth in order to force the targeted machine to drop or not serve the useful requests;
- Reflecting DoS: Where the attacker will reflect the traffic from normal (innocent) machines toward the targeted machine to inflict harm;
- Service theft: Gaining unauthorized access.

One of the most used types of tunnels is the well-known 6in4 [28], which has several attacking scenarios. Each element within the IPv6 network sends an RA message to declare that it is there and to allow other routers to update their table of existing IPv6 addresses. An attacker might take advantage of this process especially when all routers are sitting on the same network, which gives the possibility to attack the 6to4 router with a route

advertisement (RA) message, by sending spoofed packets to the victim [27]. The second type is spoofing, as explained by [28], where an attacker can take advantage of an existing tunnel and spoof one side of the tunnel endpoint and attack the other side, as shown in Figure 3. The reason for executing such an attack is that tunnels are not aware of what is behind the requester due to the absence of authentication, so a specific tunnel endpoint does not know which other endpoints reside in the same tunnel. The tunnel endpoint only assumes that packets come from the other tunnel endpoint [29]. This lack of knowledge has drawn the attention of attackers to exploit it, especially if the attacker's packet is difficult to trace. That means the attacker is able to conduct sniffing, spoofing, and DoS attacks by leveraging this vulnerability.

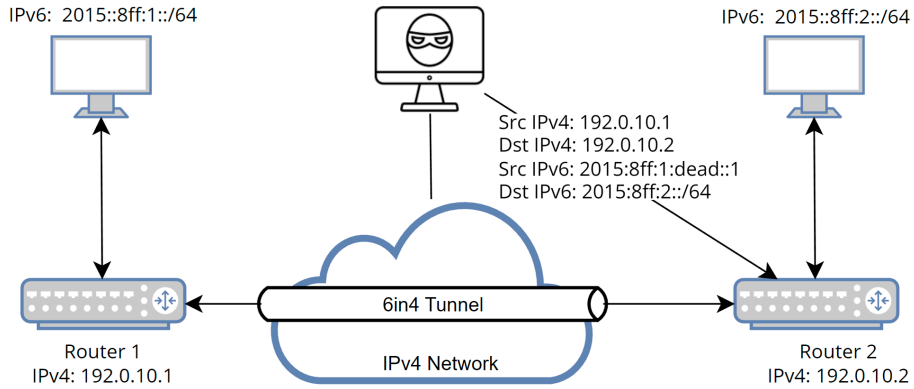

Figure 3. 6in4 tunnel attacking scenario [28].

IPv6 tunnels have some other weaknesses, being liable to password and "internet key exchange" attacks [30]. The golden rule according to [31] is that the security of the tunnel comes from the security of the tunnel endpoints themselves and the inner packets of the tunnel. If they are secured, then the tunnel in general is secured. So, the attacks to which tunnels are generally susceptible are as follows:

- Sniffing;
- Spoofing;
- DoS;
- Password and Internet key exchange.

4.4. Mitigation Methods

End-to-end secure communication systems such as IPsec [32] can help to protect the tunnel endpoint nodes. Packet filters such as firewalls can also be helpful [32]. The only drawback with this solution is the inability to see the content of the packet at the edge firewall when applying the filtering method, which means the authentication and host validation steps must be implemented.

Furthermore, some research works suggest the usage of separate firewall filters, one for IPv4 packets and another for IPv6 packets, in order to make sure that every packet is being filtered and examined [28]. Another method is to deny the IPv6 tunnel by blocking a specific protocol and its associated ports [28]. In the case of a 6in4 tunnel, an attacker might inject an already encapsulated packet within the tunnel, which is why firewalls should be enabled to inspect tunneled packets and the firewall should also permit any 6in4 traffic which is generated from the same segment or from outside it, to be able to filter traffic behind the segment as well [28]. Finally, IPsec has the ability to block any traffic that did not pass the authentication phase [28].

Another proposal was made by [33], which is based on the validation of the real source IP address by using the IP trace-back method; by doing so, the tunnel endpoint can drop any incoming spoofed IP packet. However, this technique comes also with its limitations: it cannot trace the packet source IP address until the tunnel endpoint node due to the

stateless IP routing, where a router can only see the next hop IP address, in which the tunnel end-node cannot trace the incoming packet to its original sender, which means a full end-to-end route knowledge is unachievable [33].

One more spoofing mitigation trial has been introduced by [34], which focuses on preventing the spoofed packet at the network edge. The principle is based on using an authentication algorithm by the sender, where every packet has a uniquely generated signature attached to its extension header.

Finally, Lee [35] has presented his own spoofing mitigation method, where he tackles sneaky packets that are trying to avoid filters by presenting a packet-filtering mechanism using the well-known Linux Netfilter framework [36].

4.5. DS-Lite Tunnel Challenges

The literature is rich with attack scenarios regarding 6in4 tunnels. However, little is known about the 4in6 tunnel in terms of security vulnerabilities. The concept might seem the same, but the implementation might differ a bit, leading us to focus in this paper on DS-Lite and its security analysis.

DS-Lite uses 4in6 tunneling. This type of tunneling was first presented in RFC 2473 [26], where IPv6 tunneling basically creates a link (virtual one) between the two ends of the tunnel. The IPv4 original packet will be sent as a payload encapsulated in the IPv6 tunnel packet, as shown in Figure 4.

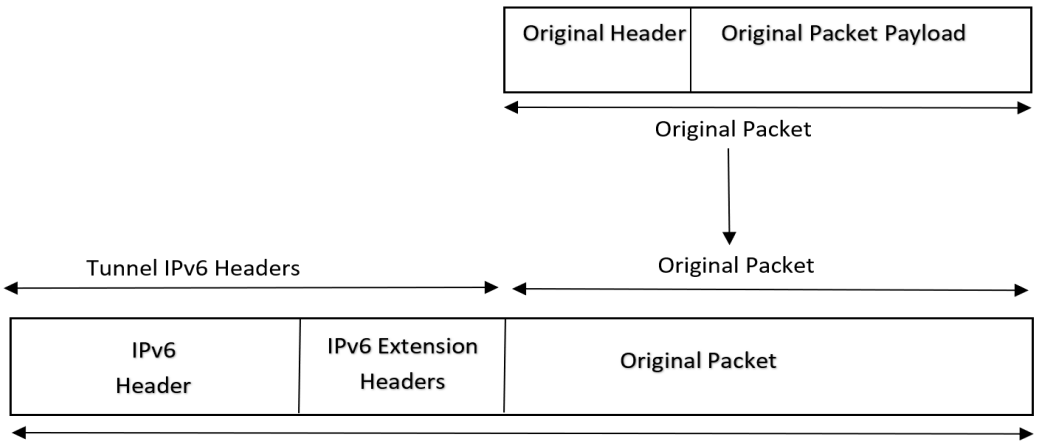

Figure 4. 4in6 tunnel encapsulation overview [26].

The process begins by grabbing the original IPv4 packet and prepending it to another IPv6 header (tunnel header) and sometimes the extension header, as it is an optional feature, and then sending this encapsulated packet through the softwire tunnel to the other tunnel end-point node, where the packet will be decapsulated and the original packet will be extracted and forwarded according to its destination and routing table.

For a better understanding of 4in6 tunnel components, the below tunnel terminologies are clarified [26]:

- Virtual link: Refers to the IPv6 tunnel itself;
- Original packet: The IPv4 packet that is supposed to be sent over the link, which consists of two components: packet header and payload;
- Tunnel IPv6 header: Refers to the IPv6 tunnel header without the payload;
- Tunnel IPv6 packet: Refers to the whole IPv6 tunnel packet (tunnel IPv6 header + original packet);
- Tunnel entry-point node: The node responsible for initiating the encapsulation process, also called the "encapsulator node". It is the source of the IPv6 tunnel packet;
- Tunnel exit-point node: The node responsible for receiving the encapsulated packet and then decapsulating it, extracting the original IPv4 packet components (header + payload)

and then forwarding the IPv4 packet to its destination, also called the "decapsulator node". It is the destination of the IPv6 tunnel packet;
- TTL (Time to Live) behavior: During the packet-forwarding process of the tunnel IPv6 packet, the IPv4 header's (the original packet header) TTL value decreases by 1.

Note: Both end-nodes are capable of encapsulation/decapsulation; the naming refers only to the function of the process initiators. DS-Lite has an advantage when it comes to DoS attacks from random public IPv4 addresses. This is due to the following process:

- AFTR receives an encapsulated packet, then decapsulates it;
- If the extracted packet payload has a source IPv4 address that is not private RFC 1918, it will be immediately dropped;
- This built-in security feature gives DS-Lite the ability to stop DoS attacks stemming from unauthorized IPv4 addresses [12].

DS-Lite, however, has a drawback because it uses stateful translation on the AFTR side, which makes the machine liable to DoS attacks and limits its scalability in case of expansion or the need in bigger projects, especially when there are numerous amounts of B4s routers within the infrastructure [17].

5. STRIDE Methodology

STRIDE stands for Spoofing, Tampering, Repudiation, Information Disclosure, and Elevation of Privilege. The method was explained in detail in [37], which summarized the general attacks that any network system may be vulnerable to. Below is a brief description of each attack:

- Spoofing: The claim to be someone you are not or the act of some malicious user who pretends to be, e.g., a trusted website [37];
- Tampering: The changing in the actual data flow between two nodes [37];
- Repudiation: The ability of the sender to deny the fact that he did send a specific packet or sign a specific document [37];
- Information Disclosure: The access to confidential information that one should not have, such as the TTL value of a packet, or confidential data such as online banking credentials or any other login credentials [37];
- Denial of Service: The process of overwhelming a specific server or network connection with a huge number of useless queries or data in order to block the legitimate ones from receiving a reply or even being processed [37]. In other words, DoS makes the targeted machine unavailable for its original purpose;
- Elevation of Privileges: The access of a specific user to a certain level of sensitive data which might be a strictly confidential file that is meant to be for upper management or the root user, for instance [37].

According to [37], the best method to test the system's vulnerabilities is to build a DFD (data flow diagram) for the system and apply the STRIDE method to it. Table 3 shows the different potential security threats for each DFD element.

Table 3. Vulnerability of different DFD elements to different threats [1].

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

6. Applying STRIDE to DS-Lite

The first step in applying the STRIDE method is to build the DFD of the examined system and specify the potential attacking points at each element. Therefore, the DFD for

DS-Lite was built as shown in Figure 5, which shows the spots (1–11). The security analysis is divided into several groups such as the traffic between the client and B4, then between B4 and the AFTR, etc.

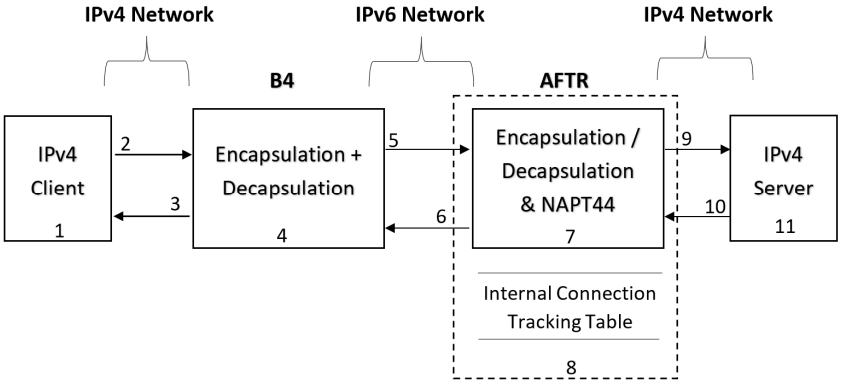

Figure 5. Data flow diagram of DS-Lite.

6.1. IPv4 Client

1. Spoofing: A malicious user might spoof the IP address of the client and attack the B4 router by sending many useless requests to the B4 and fully utilizing its computation power;
2. Repudiation: The request initiator might deny the fact that he made the request in the first place. By doing so, a malicious user might attack the B4 router and then deny the fact that he was the one who sent this specific request such as an echo request, DNS resolution request, etc.

6.2. Data Flow from the Client to the B4 Router

1. Tampering: An adversary might tamper (change or modify) the content of the flowing packets such as IP address, source port number, etc. The attacker might also diverge the packet toward a fraudulent server which might cause an FoS (failure of service) attack, which is the prevention of the legitimate requester from receiving a response [1];
2. Information Disclosure: An attacker might get hold of confidential information such as the TTL value of a packet, e.g., some sensitive information sent by the client as plain text or the browsing habits of the client [1];
3. Denial of Service: An attacker sends too many useless requests to the B4 router and blocks the legitimate requester from having his query processed.

6.3. Data Flow from B4 Router to IPv4 Client

1. Tampering: In this case, the potential victim is the client, who is vulnerable to an attack at the application level, such as sending a TCP RST signal that will end the already established TCP connection [6] and also a de-authentication attack [38] that results in disconnecting the device from the router's Wi-Fi, in addition to a plain-text injection attack [38] that allows the B4 router to send misleading information to the client, such as false routing details;
2. Information Disclosure: An attacker gaining access to sensitive information in a malicious way, such as an important text sent back by the IPv4 server behind AFTR;
3. Denial of Service: Sending a high number of forged replies to overwhelm the client and prevent the client from sending more requests to the B4.

6.4. B4 Router

1. Spoofing: Another machine presents itself as the legitimate B4 router and starts communicating with the rest of the DS-Lite network elements, which puts everyone that deals with this machine at high risk, such as a malicious device initiating a

communication with the AFTR router and presenting itself as the B4 router, which leads to whole traffic (from AFTR side) pouring in the wrong direction by changing the destination IP address of the packet, which results in the original sender not receiving a response to his request. Another consequence could be when the malicious server starts recording the data passing through and using it for illegal activities such as blackmailing;

2. Tampering: Modifying the current source and destination address of the 4in6 traffic that is being processed within the B4 router in order to shift the traffic and prevent it from being sent to the real AFTR, forwarding it to a malicious server. Another possibility could be tampering with the IP address/port pair of the sender, which eventually results in a different entry in the AFTR translation table, and therefore, the packet reply will not return to the original sender;
3. Repudiation: In this case, when the B4 is spoofed, the malicious user will deny the fact that he sent a specific packet even though he did. The obvious solution to this issue is proper logging [6];
4. Information Disclosure: Attacker gaining access to sensitive data within the router such as the client's source IP address, TTL value, browsing data, etc.;
5. Denial of Service: Flooding the AFTR router with unnecessary requests in order to overwhelm the AFTR and stop it from processing any further incoming packets (see Section 6.2, item 3);
6. Elevation of Privileges: The attacker gains access to high-level (privileged) data, which mainly occurs due to an inside job [1], when an employee is implicit in the act.

6.5. Data Flow from B4 to AFTR

1. Tampering: 4in6 traffic might be altered and the source IP address or port number is liable to be modified by the attacker;
2. Information Disclosure: The access to confidential information in many ways (see Section 6.2, item 2);
3. Denial of Service: The flood of useless queries exhausts the resources of the AFTR and hinders its main job (see Section 6.2, item 3).

6.6. Data Flow from AFTR to B4

1. Tampering: Modifying the traffic details before it reaches the B4 router, such as the source IP address or port number of the 4in6 flowing traffic;
2. Information Disclosure: The unauthorized access to sensitive data (see Section 6.2, item 2);
3. Denial of Service: Flooding the B4 router with useless traffic (see Section 6.2, item 3).

6.7. AFTR Router

1. Spoofing: Attacker impersonates the AFTR and starts communicating with the rest of the machines around him, such as the B4 router and IPv4 server, which might lead to exchanging sensitive data with the wrong person (see Section 6.4, item 1);
2. Tampering: Changing the content of the 4in6 packet payload (see Section 6.4, item 2);
3. Repudiation: Hiding the packet sender identity (see Section 6.4, item 3);
4. Information Disclosure: The possibility of an attacker gaining access to confidential data (see Section 6.4, item 4);
5. Denial of Service: Various types of attacks can be described in this section, such as exhausting the AFTR with too many useless requests and hindering the process of encapsulating the desired packet;
6. Elevation of Privileges: The act of unauthorized access to a very confidential data center of a specific folder in a malicious way (see Section 6.4, item 6).

6.8. Internal Connection Tracking Table of the AFTR

1. Tampering: An attacker can manipulate the connection tracking table of the AFTR router that saves the incoming 4in6 traffic packet with its source address and port number and maps it with a destination IPv4 device according to the stateful NAT rule within the router (see Table 1). So, the manipulation can cause a faulty destination address and disturb the encapsulation process and result in a packet loss;
2. Denial of Service: Flooding the connection tracking table with too many unnecessary entries (false connections) might overwhelm the table beyond its capabilities and cause it to flush its data or saturate the AFTR itself, which means the AFTR will no longer be able to process any incoming packet, or it will at least lose some legitimate connections.

6.9. Data Flow from AFTR to IPv4 Server

1. Tampering: The destination IP of the traffic might be altered and redirected to a malicious server instead of being directed to the legitimate IPv4 server;
2. Information Disclosure: Unauthorized access to sensitive data of the IPv4 traffic;
3. Denial of Service: Overwhelming the host with too many requests (see Section 6.3, item 3).

6.10. Data Flow from IPv4 Server to AFTR

1. Tampering: The IPv4 packets might be altered in terms of the IP source address or port number, which will definitely change the corresponding 4in6 traffic details;
2. Information Disclosure: Unauthorized access to IPv4 traffic before it reaches the AFTR router, which might leak the confidential browsing habits of the client;
3. Denial of Service: Exhausting the AFTR and its main encapsulation function (see Section 6.2, item 3).

6.11. IPv4 Server

1. Spoofing: An attacker impersonates the IPv4 server and initiates communication with AFTR and the rest of DS-Lite topology;
2. Repudiation: The host (IPv4 sever) might hide his identity, perform all sorts of malicious acts, and then deny any responsibility for them. For example, in the previous point of spoofing, the attacker might spoof the IPv4 server, perform an attack against AFTR, and then deny the fact that he initiated the communication in the first place.

7. DS-Lite Testbed

7.1. Testbed Topology and Specifications

As shown in Figure 6, the testbed consisted of five machines (IPv4 client, B4, AFTR, IPv4 server, and the attacker). They were all based on VMware workstation VMs and built upon CentOS-7 images. Every machine had the following specifications:

- RAM: 3 GB;
- Hard disk: 20 GB;
- CPU: 1 core.

The host PC that hosted the whole testbed had the following specifications:

- OS: Windows 10 Pro;
- RAM: 16 GB;
- Hard disk: 1 TB;
- CPU: Intel-Core i7, 4 physical cores.

The IPIP6 tunnel was built between B4 and AFTR, which took care of encapsulating the IPv4 packet inside the IPv6 packet and then decapsulating it. For the NAT process on the AFTR side, the `iptables` rule was configured to masquerade the source IP address to

the AFTR's network interface (ens35). As for the tunnel between B4 and AFTR, a shell script was executed on the B4 machine, which consists of the following commands:

```
ip link add name ipip6 type ip6tnl local 2001:db8:0:1::2\
remote 2001:db8:0:1::1 mode any dev ens39
ip link set dev ipip6 up
ip route add 198.51.100.0/24 dev ipip6
```

It is noted that an important simplification is contained in the above setup compared to Table 1: the Softwire-ID is missing from the connection tracking table. It was observed that the experiments were not influenced by this change, but rather the set-up of the testbed was simplified: `iptables` could be used instead of a real AFTR implementation.

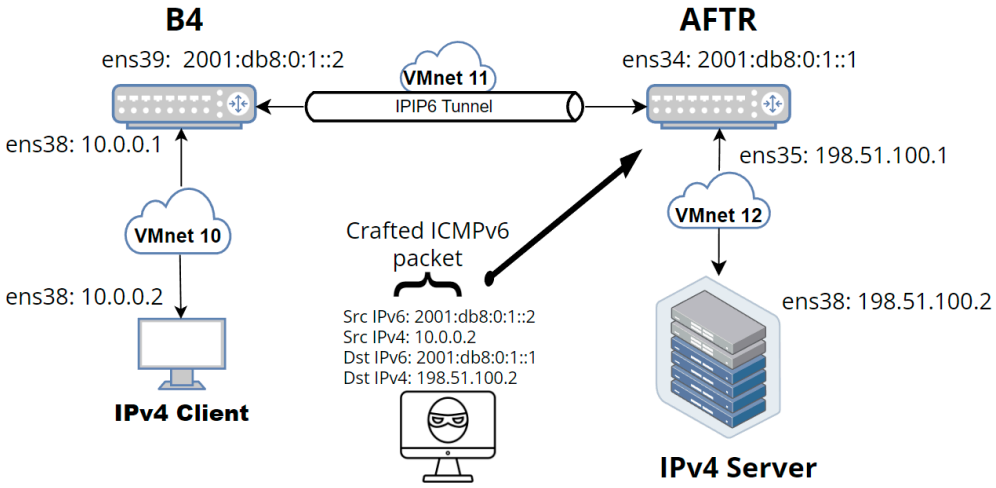

Figure 6. DS-Lite testbed (attack Scenario 1).

7.2. Attack Scenarios

7.2.1. Spoofing Attack from within VMnet-11 (an Inside Job)

The proposed attack shown in Figure 6 is spoofing the ingress tunnel endpoint (ens39), which is the softwire tunnel interface of the B4 router, and then sniffing/logging traffic or performing MitM attack against the egress tunnel endpoint (AFTR).

The attack was based on the "thc-ipv6" toolkit [39], which has several tools within it. One powerful tool was chosen to perform the attack (four2six). It manually creates a crafted packet of IPv4 inside IPv6 (encapsulated packet) and sends it over the channel to the victim (AFTR).

This crafted packet is actually a spoofed one, imitating that it was sent by the B4 router, having the same specifications as a genuine B4-originated packet:

- IPv6 Src: 2001:db8:0:1::2;
- IPv6 Dst: 2001:db8:0:1::1;
- IPv4 Src: 10.0.0.2;
- IPv4 Dst: 198.51.100.2.

In fact, the spoofed attack went through AFTR and then to the end receiver (IPv4 server), and a series of ICMP echo requests and replies had been exchanged between the attacker and victim. In more detail, what happened is:

- The AFTR received the malicious packet;
- The AFTR saw the packet as a 4in6 packet, and therefore decapsulated the packet;
- The AFTR then forwarded the extracted IPv4 packet with the source address of 198.51.100.1 towards the IPv4 server (198.51.100.2);
- IPv4 server sent ICMPv4 echo reply back;
- The AFTR saw the later packet and encapsulated it in an IPv6 packet;
- The echo reply from the AFTR, which is an IPv6 packet with an IPv4 packet in its payload, was sent back to its destination IPv6 of address 2001:db8:0:1::2;

- Figure 7 shows three different terminals: the top one is the incoming packets at the attacker interface, the one in the middle is the B4 incoming traffic from the AFTR side, and at the bottom is the incoming traffic at the IPv4 client;
- B4 received the IPv6 packet carrying an IPv4 packet with ICMP echo reply because it is the legitimate owner of the 2001:db8:0:1::1 IPv6 destination address;
- The attacker saw the same reply because its NIC runs in promiscuous mode;
- The interesting thing here is illustrated by the IPv4 client receiving an ICMP4 echo reply for a request that he did not request in the first place, which was issued by the attacker initially;
- Echo replies in Figure 7 are circled in red.

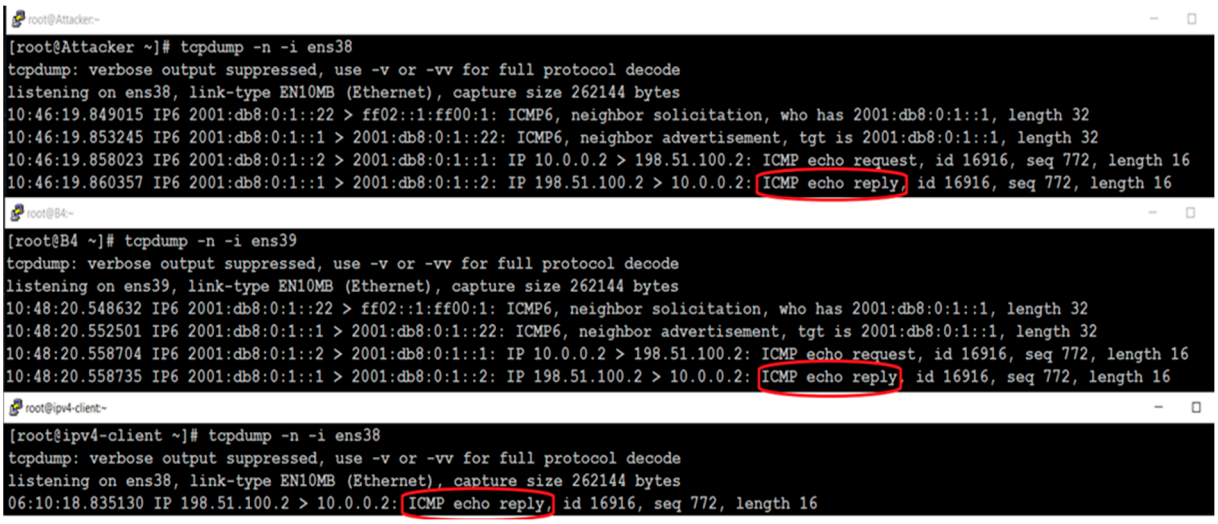

Figure 7. Spoofed packet echo reply.

7.2.2. Attack from within VMnet-10 (Compromised B4 Network)

It is also worth mentioning that B4 by design is also capable of routing native IPv6 packets, which takes us to Figure 8, where the IPv6 address was added to interface ens38 at the B4 machine in order to process and forward incoming IPv6 packets.

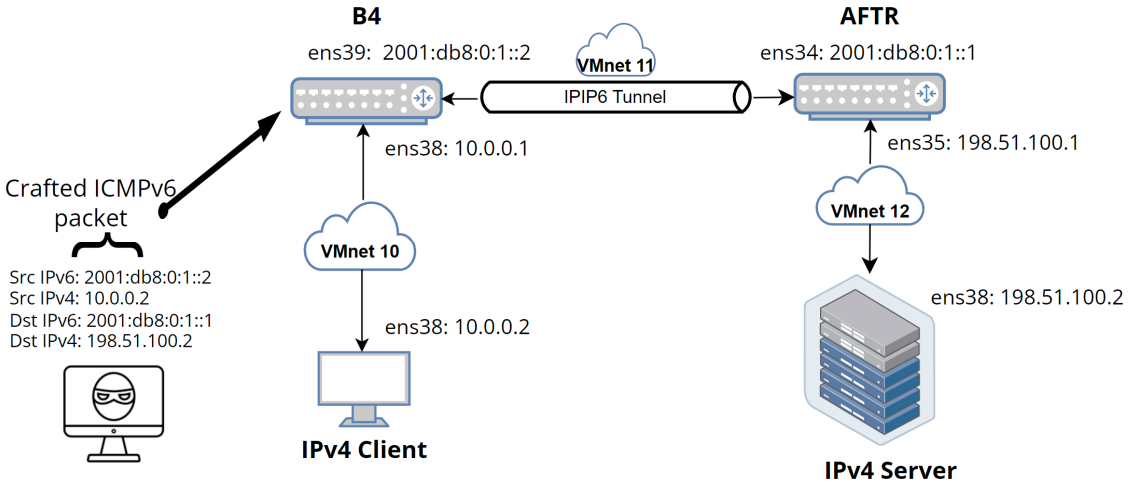

Figure 8. DS-Lite testbed (attack Scenario 2).

A real-life spoofing scenario could also be stopped before reaching the internal network of AFTR (VMnet-11 in the current example). Therefore, another attacking scenario was proposed in Figure 8, where the attacker is not sitting directly in the AFTR network, but outside it. The idea behind such an attack is that an attacker has access to the home router

network, which gives him direct contact with the CPE. This access allows the attacker to send a crafted packet with a spoofed IPv6 address of a B4 router or any other IPv6 address from the internal network of AFTR (VMnet-11 in the current example).

Fortunately for the user, the attack did not go through—the B4 router was always dropping any incoming packet from the attacker-2 machine that has an IP address (a spoofed one) from the pool of (VMnet-11). The reason for that is simple: there was no route configured on the B4 router to forward the crafted IPv6 packet to the AFTR router. The tunnel was hardcoded in the B4 machine by creating an ipip6 tunnel interface to forward any IPv4 packet (heading towards 198.51.100.0/24 network) to the ipip6 tunnel interface which has been configured. However, the attacker packet has no route to the tunnel interface, and that is why it was dropped.

7.2.3. Source Port Exhaustion Attack from within VMnet-10

At first, the attack was conducted using the same platform, a regular specs Windows-based computer; however, the results were not stable and not reproducible due to limited cores on the host PC. Therefore, the decision was made to use the resources of NICT StarBED, Japan. A "P" series node [40] was used, which is a Dell PowerEdge 430 server with the following specifications: two Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.1 GHz CPUs having 16 cores each and 348 GB 2400 MHz DDR4 SDRAM.

Windows 10 Pro operating system was installed, and the same process was repeated by building the testbed using VMware workstation player and CentOS-7-based virtual machines. The specifications for each machine were as follows:

- RAM: 4 GB;
- Cores: 4 cores, except for AFTR having 6 cores;
- Disk: 20 GB.

Before disclosing the attack details, it is important to explain the mathematical sense behind it.

In the networking world, ports are counted based on a 16-bit standard. The 16 bits were chosen when the TCP and UDP standards were designed [41]. As a result, the maximum number of ports equals 2^{16} = 65,536. This number applies to TCP and UDP ports. In this experiment, the focus was on the UDP ports and how to exhaust them by sending too many queries (UDP requests) in a short period of time. However, not all ports in the range are usable for establishing new connections, because ports between 1 and 1024 are called well-known ports and they are reserved for specific functions such as FTP, HTTP, DNS, etc.

Therefore, the total range of ports that the NAPT (Network Address and Port Translation) device can use as UDP source port numbers equal 65,536 − 1024 = 64,512 ports.

Figure 9 shows the attacking Scenario 3 of sending too many AAAA queries from "IPv4 Client-1" and "IPv4 Client-2" machines. In order for the attack to function, a few files must be correctly configured:

- `/sys/module/nf_conntrack/parameters/hashsize`: The hash size is actually the size of the hash table storing the lists of conntrack entries [42], which is better as a base of 2, which means that hash size could be 2^{14} = 16,384;
- `/proc/sys/net/netfilter/nf_conntrack_max`: It represents the maximum number of allowed conntrack entries that netfilter can keep running simultaneously. It was set to hashize × 8 = 16,384 × 8 = 131,072. This number will have a significant impact when the attack is run later so that the NAT table does not get full easily;
- `/proc/sys/net/netfilter/nf_conntrack_udp_timeout`: This timeout means that after 30 s, the already used UDP ports are ready to be re-assigned by the kernel, and the default value of 30 s was left as it is;
- `/etc/security/limits.conf`: This file controls the number of maximum open files at the same time, where the below line was added: `root hard nofile 1000000.`

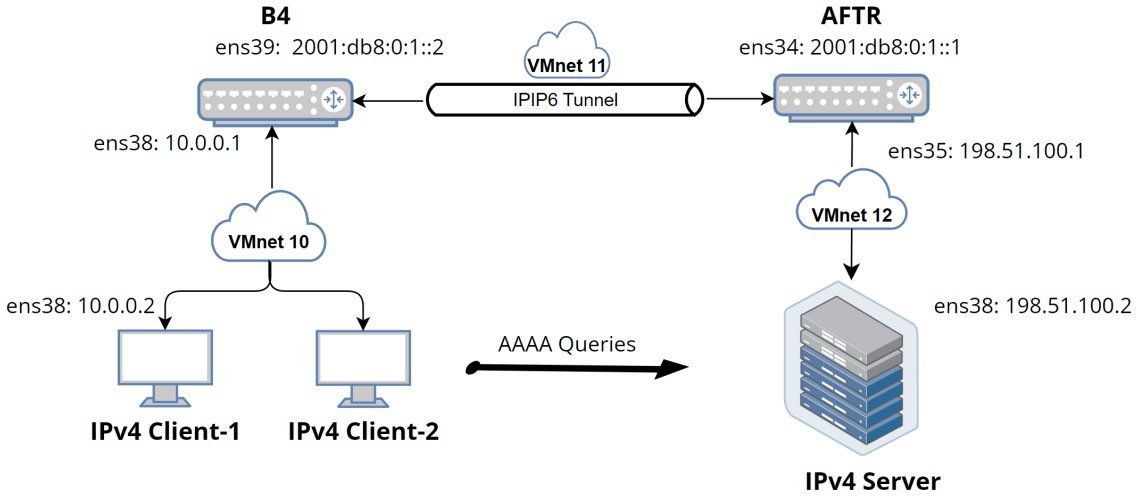

Figure 9. Attacking Scenario 3 (AAAA queries).

To perform this kind of attack, another tool can be used, where a huge number of DNS queries can be sent to the target machine (AFTR), which is called "dns64perf++" [43] (described in [44]). When all parameters (on the attacking machine side) are set correctly (number of requests, the delay between requests, etc.), the AFTR pool of source ports will be exhausted, and the machine will not be able to process incoming packets anymore until the UDP timeout (port allocation timeout in AFTR machine) is up. The below command was used on each of the attacking clients:

```
/dns64perf++v4 198.51.100.2 53 0.0.0.0/5 60000 1 1 60000 400000 0.1
```

This command sends 60,000 queries from each client. For two clients, this number reaches 120,000 queries, which is still under 131,072 (`nf_conntrack_max`). As a result, the NAT table will not be overfilled with packets that exceed its capacity. Another interesting fact in the command is the delay of 400,000 nanoseconds between queries, resulting in 2500 queries per second, which means that for two clients, a total of 5000 queries per second can be sent.

The attack was executed by an automated script from client 1, carrying out the following:

- Accesses AFTR remotely (via ssh) and runs tshark (traffic monitoring software) and captures the traffic at the ens35 interface;
- Runs the attack from IPv4-client-1 (the same machine);
- Accesses IPv4 client-2 remotely (via ssh) and runs the same attack with the same parameters;
- Waits until attacks are finished on both clients;
- Accesses AFTR remotely (via ssh) and stops tshark;
- Sends the tshark results as a pcap file to the host machine, so it can be read in the Wireshark software.

Figure 10 shows the exhaustion of the source ports clearly to prove that a total of 64,512 sent queries must be visible, which comes from subtracting the well-known 1024 ports from the total of 65,536 ports. Figure 10 shows that AFTR processed only 64,536 packets at the ens35 interface. After digging deeper, more packets embedded within Wireshark results were found, such as 20 ICMP and 4 ARP packets. As a result, the remaining packets are $64{,}536 - 20 - 4 = 64{,}512$, which is exactly the anticipated number.

No.	Time	Src. IP	Dst. IP	Protocol	Src. port	Dst. port	Info.
64534	14.336138714	198.51.100.1	198.51.100.2	DNS	38628	53	Standard query 0x8131 AAAA 000-000-129-049
64535	14.339424635	198.51.100.1	198.51.100.2	DNS	38624	53	Standard query 0x8139 AAAA 000-000-129-057
64536	14.386624374	198.51.100.1	198.51.100.2	DNS	38625	53	Standard query 0x81b0 AAAA 000-000-129-176

Figure 10. Last lines for Wireshark capture at ens35 on AFTR.

In principle, 60,000 queries were sent from each client (120,000 in total), and AFTR was able to process no more than 64,512 packets and ran out of ports in around 14 s. This failure of service is supposed to last until the UDP session timeout at AFTR is over and new source ports are available to be reallocated. Moreover, Figure 11 shows the interaction between all nodes of DS-Lite topology, where the port exhaustion process at AFTR is clearly illustrated with a sequence of packets and when exactly the pool of ports is exhausted.

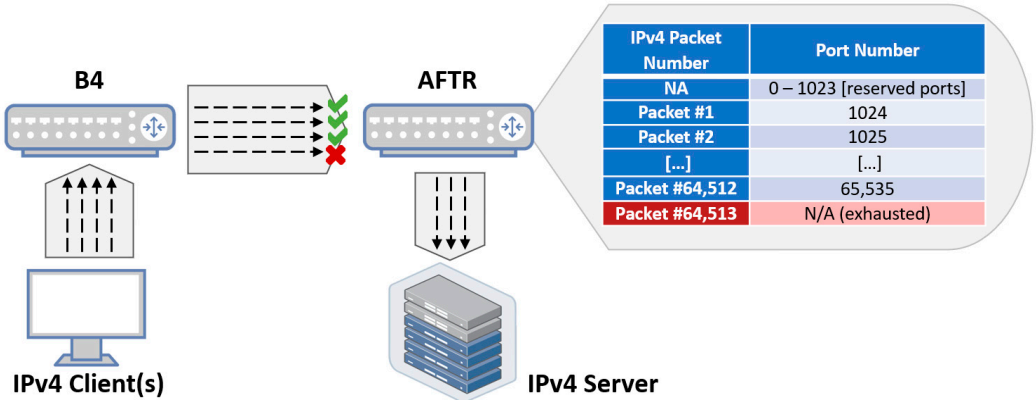

Figure 11. Illustration of port exhaustion at the AFTR.

7.3. Mitigation Method

Several IDS and IPS (intrusion detection and prevention systems) were tested in order to identify and drop the spoofed packet from the legitimate one. The process proved to be quite hard to implement, especially when the attacker machine is located inside the access network of the ISP (VMnet-11). Therefore, a workaround had been put in place and iptables rules were emplaced to perform the job.

7.3.1. First Attack Mitigation

An `ip6tables` rule was configured at AFTR in order to allow the incoming traffic at the ens34 interface of the AFTR only under the condition that the incoming packet's source MAC address belongs to the B4 ens39 interface (see Figure 6).

```
ip6tables -A INPUT -i ens34 -s 2001:db8:0:1::2 -m mac !  -mac-source 00:0c:29:67:14:14 -j DROP
```

On the other hand, such a mitigation method will not be effective if the attacker clones the "`00:0c:29:67:14:14`" MAC address to bypass the `iptables` filtering rule.

7.3.2. Second Attack Mitigation

Despite the fact that this attack did not go through and the spoofed packet had been dropped, a proposed solution is presented here in case of building the topology using another machine (different Linux distribution for instance), where such an attack could be prevented at the B4 router side with the following rules:

```
iptables -A FORWARD -i ens38 -s 10.0.0.0/24 -j ACCEPT
iptables -A FORWARD -i ens38 -j DROP
```

The first rule accepts packets with source addresses from the private IPv4 address network (10.0.0.0/24), while the second one drops every packet that approaches the ens38 interface and needs to be forwarded. These two rules take care of a malicious packet as it reaches the B4 side and before it heads toward the AFTR (see Figure 8).

However, restricting access to the private IP address network of VMnet-10 (10.0.0.0/24) will not fully protect from DoS attack, because the attack might be originated from a machine that resides within the network of the IPv4 client (an inside job).

7.3.3. Third Attack Mitigation Method

One of the ways to mitigate this attack or at least reduce its impact is by adding two (or more) IP addresses to the AFTR ens35 interface (see Figure 9), where the whole calculation will be double (or triple, depending on how many IP addresses were added) due to the fact that extended NAT works by binding the IP/port pair together. The fact that ens35 has two IP addresses now gives the AFTR more flexibility by having one additional pool of exactly the same port numbers of 64,512 ports, but they will be coupled with another source IP address. In case two IP addresses were not enough to mitigate the attack, the system administrator can easily add a third, fourth, or more as needed. On the practical side, two extra IP addresses were added to the ens35 interface on AFTR. As a result, the AFTR was able to process all incoming 120,000 packets, unlike last time, when it stopped after 64,512 packets.

Another mitigation method is possible through the rate-limiting process, which limits the number of packets per second that can be sent and received by the NIC (network interface card) [45]; it also enables the administrator to assign bandwidth restrictions to specific traffic such as ICMP, UDP, etc. Some research works [46,47] have proposed the rate-limiting mitigation mechanism after noticing the difference and the asymmetry between incoming and outgoing traffic in the designated network.

7.4. Summary

As the testbed for DS-Lite investigated in a topic that has rarely been discussed and analyzed, valuable results were collected based on the attacking scenarios, which can be used to enhance the security around DS-Lite infrastructure's vulnerable spots and, therefore, provide a more secure network infrastructure. Therefore, researchers are encouraged to invest more effort in analyzing the potential security threats that face IPv6 transition technologies, especially DS-Lite. It is recommended that the focus be placed on DoS attacks and source IP address spoofing, as it was found through analysis that these attacks are the most common. As for mitigation methods, a sophisticated IDS and IPS tool is highly recommended, such as SNORT [48] or Suricata [49].

8. Conclusions

In this paper, we have demonstrated the significance of transition technologies, specifically through the tunneling method, and how practical they can be. Our testbed effectively simulated a DS-Lite topology and uncovered its security vulnerabilities. The findings show that the IPv4 client did not need a public IPv4 address to communicate with an IPv4 server. On the other hand, every element within the DS-Lite topology is vulnerable to various types of attacks, including DoS, tampering, and spoofing. As such, further analyses and addressing vulnerabilities are crucial for the successful implementation of IPv6 transition technologies that use tunneling. Our work has benefited from the "thc-ipv6" toolkit, which provided us with a range of tools and attacking possibilities. These tools can be used in future research to better understand the weaknesses and threats associated with DS-Lite and other transition technologies.

Overall, our study highlights the importance of thorough analysis and vulnerability testing when implementing IPv6 transition technologies that use tunneling. Our findings can benefit network administrators, policy makers, and researchers who seek to enhance the security of their networks and prevent attacks.

Author Contributions: All authors contributed to the study's conception and design. Material preparation and testbed building by A.A.-A., with valuable support from G.L. The first draft of the manuscript was written by A.A.-A., and all authors commented on previous versions of the manuscript. Conceptualization: G.L.; methodology: A.A.-A.; formal analysis and investigation: A.A.-A.; writing—original draft preparation: A.A.-A.; writing—review and editing: G.L. and A.A.-A.; resources: A.A.-A. and G.L.; supervision: G.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The Public GitHub repository of the first author contains several automation scripts to build every machine that was used to build the topology: https://github.com/ameen-mcmxc/DS-Lite_Test_Bed (accessed on 16 March 2021). The "pcap" file generated by Wireshark software captures the traffic at the ens35 interface of the AFTR machine in case of attack scenario 3. The results of the attack were published in the same public GitHub repository file directory: https://github.com/ameen-mcmxc/DS-Lite_Test_Bed/blob/main/DNS_Perf_Results.pcap (accessed on 12 June 2022).

Acknowledgments: In part, the resources of NICT StarBED, Japan, were used for conducting the experiment. The authors thank Shuuhei Takimoto for the opportunity to use these resources. The authors thank Sándor Répás, András Gerendás, and Omar D'yab for their reading and commenting on the manuscript. The authors also would like to acknowledge the preprint version of this work, which is available at [50]. This paper is based on material previously published in the preprint, and we thank the previous reviewers for their valuable feedback.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lencse, G.; Kadobayashi, Y. Methodology for the identification of potential security issues of different IPv6 transition technologies: Threat analysis of DNS64 and stateful NAT64. *Comput. Secur.* **2018**, *77*, 397–411. [CrossRef]
2. Al-Azzawi, A. Towards the Security Analysis of the Five Most Prominent IPv4aaS Technologies. *Acta Tech. Jaurinensis* **2020**, *13*, 85–98. [CrossRef]
3. Bagnulo, M.; Sullivan, A.; Matthews, P.; Beijnum, I. DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers. IETF RFC 6147. 2011. Available online: https://www.rfc-editor.org/info/rfc6147 (accessed on 18 May 2023).
4. Bagnulo, M.; Matthews, P.; Beijnum, I. Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers. IETF RFC 6146. 2011. Available online: https://www.rfc-editor.org/info/rfc6146 (accessed on 12 January 2021).
5. Mawatari, M.; Kawashima, M.; Byrne, C. 464XLAT: Combination of stateful and stateless translation. IETF RFC 6877. 2013. Available online: https://www.rfc-editor.org/info/rfc6877 (accessed on 16 January 2021).
6. Al-Azzawi, A.; Lencse, G. Towards the Identification of the Possible Security Issues of the 464XLAT IPv6 Transition Technology. In Proceedings of the 2020 43rd International Conference on Telecommunications and Signal Processing (TSP), Milan, Italy, 7–9 July 2020. [CrossRef]
7. Al-Azzawi, A.; Lencse, G. Testbed for the Security Analysis of the 464XLAT IPv6 Transition Technology in a Virtual Environment. In Proceedings of the 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech, 26–28 July 2021. [CrossRef]
8. Al-Azzawi, A.; Lencse, G. Identification of the Possible Security Issues of the 464XLAT IPv6 Transition Technology. *Infocommun. J.* **2021**, *13*, 10–18. [CrossRef]
9. Lencse, G.; Kadobayashi, Y. Comprehensive survey of IPv6 transition technologies: A subjective classification for security analysis. *IEICE Trans. Commun.* **2019**, *102*, 2021–2035. [CrossRef]
10. Lee, Y.; Maglione, R.; Williams, C.; Jacquenet, C.; Boucadair, M. Deployment Considerations for Dual-Stack Lite. IETF RFC 6908. 2013. Available online: https://www.rfc-editor.org/info/rfc6908 (accessed on 9 March 2021).
11. Fu, Y.; Jiang, S.; Dong, J.; Chen, Y. Dual-Stack Lite (DS-Lite) Management Information Base (MIB) for Address Family Transition Routers (AFTRs). IETF RFC 7870. 2016. Available online: https://www.rfc-editor.org/info/rfc7870 (accessed on 11 March 2021).
12. Durand, A.; Droms, R.; Woodyatt, J.; Lee, Y. Dual-Stack Lite Broadband Deployments following IPv4 Exhaustion. IETF RFC 6333. 2011. Available online: https://www.rfc-editor.org/info/rfc6333 (accessed on 19 March 2021).
13. Hankins, D.; Mrugalski, T. Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite. IETF RFC 6334. 2011. Available online: https://www.rfc-editor.org/info/rfc6334 (accessed on 12 March 2021).
14. Boucadair, M.; Jacquenet, C.; Sivakumar, S. A YANG Data Model for Dual-Stack Lite (DS-Lite). IETF RFC 8513. 2019. Available online: https://www.rfc-editor.org/info/rfc8513 (accessed on 18 May 2023).
15. Chen, M.; Yi, M.; Huang, M.; Huang, G.; Ren, Y.; Liu, A. A novel deep policy gradient action quantization for trusted collaborative computation in intelligent vehicle networks. *Expert Syst. Appl.* **2023**, *221*, 119743. [CrossRef]
16. Zhang, R.; Li, Z.; Xiong, N.N.; Zhang, S.; Liu, A. TDTA: A truth detection based task assignment scheme for mobile crowdsourced Industrial Internet of Things. *Inf. Sci.* **2022**, *610*, 246–265. [CrossRef]
17. Lencse, G.; Palet Martinez, J.; Howard, L.; Patterson, R.; Farrer, I. Pros and cons of IPv6 transition technologies for IPv4aaS. Internet Draft. 2022. Available online: https://datatracker.ietf.org/doc/html/draft-ietf-v6ops-transition-comparison-04 (accessed on 18 May 2023).
18. Scott, O. The PPTP VPN Protocol: Is It Safe? Infosec Resources. 2019. Available online: https://resources.infosecinstitute.com/topic/the-pptp-vpn-protocol-is-it-safe (accessed on 11 April 2021).

19. Hamzeh, K.; Pall, G.; Verthein, W.; Taarud, J.; Little, W.; Zorn, G. Point-to-Point Tunneling Protocol (PPTP). IETF RFC 2637. 2019. Available online: https://www.rfc-editor.org/info/rfc2637 (accessed on 5 April 2021).
20. Schneier, B. Cryptanalysis of Microsoft's point-to-point tunneling protocol (PPTP). In Proceedings of the 5th ACM Conference on Computer and Communications Security, San Francisco, CA, USA, 2–5 November 1998; pp. 132–141.
21. Zorn, G.; Cobb, S. Microsoft PPP CHAP Extensions. IETF RFC 2433. 1998. Available online: https://www.rfc-editor.org/info/rfc2433 (accessed on 9 April 2021).
22. Zorn, G. Microsoft PPP CHAP Extensions, version 2. IETF RFC 2759. 2000. Available online: https://www.rfc-editor.org/info/rfc2759 (accessed on 1 March 2021).
23. Krishnan, S.; Thaler, D.; Hoagland, J. Security Concerns with IP Tunneling. IETF RFC 6169. 2011. Available online: https://www.rfc-editor.org/info/rfc6169 (accessed on 18 May 2023).
24. Carpenter, B.; Jung, C. Transmission of IPv6 over IPv4 Domains without Explicit Tunnels. IETF RFC 2529. 1999. Available online: https://www.rfc-editor.org/info/rfc2529 (accessed on 9 March 2021).
25. Nordmark, E.; Inc, R.G. Basic Transition Mechanisms for IPv6 Hosts and Routers. IETF RFC 4213. 2005. Available online: https://www.rfc-editor.org/info/rfc4213 (accessed on 4 May 2021).
26. Conta, A.; Deering, S. Generic Packet Tunneling in IPv6 Specification. IETF RFC 2473. 1998. Available online: https://www.rfc-editor.org/info/rfc2473 (accessed on 21 April 2021).
27. Savola, P.; Patel, C. Security Considerations for 6to4. IETF RFC 3964. 2004. Available online: https://www.rfc-editor.org/info/rfc3964 (accessed on 6 January 2021).
28. Abdulla, S.A. Survey of security issues in IPv4 to IPv6 tunnel transition mechanisms. *Int. J. Secur. Netw.* **2017**, *12*, 83–102. [CrossRef]
29. Hogg, S. IPv6: Dual Stack Where You Can; Tunnel Where You Must. Networkworld. 2017. Available online: https://www.networkworld.com/article/2285078/ipv6-dual-stack-where-you-can-tunnel-where-you-must.html (accessed on 8 April 2021).
30. Yang, D.; Song, X.; Guo, Q. Security on IPv6. In Proceedings of the 2nd IEEE International Conference on Advanced Computer Control, Shenyang, China, 27–29 March 2010; pp. 323–326.
31. Mi, W. The Applicability and Security Analysis of IPv6 Tunnel Transition Mechanisms. In Proceedings of the International Conference on Algorithms and Architectures for Parallel Processing, Dalian, China, 24–27 August 2014; pp. 560–570.
32. Hei, Y.; Katsuno, S.; Ano, S. An implementation and evaluation of IPv6 end-to-end secure communication system for closed members. In Proceedings of the International Symposium on Applications and the Internet Workshops (SAINTW'06), Phoenix, AZ, USA, 27 January 2006.
33. Amin, S.A.; Choong, S.H. On IPv6 Traceback. In Proceedings of the International Conference on Advanced Communication Technology (ICACT2006), Phoenix Park, Republic of Korea, 20–22 February 2006.
34. Xie, L.; Bi, J.; Wu, J. An authentication based source address spoofing prevention method deployed in IPv6 edge network. In Proceedings of the International Conference on Computational Science, Kuala Lumpur, Malaysia, 26–29 August 2007; pp. 801–808.
35. Lee, W.J.; Heo, S.Y.; Byun, T.Y.; Sohn, Y.H.; Han, K.J. A secure packet filtering mechanism for tunneling over Internet. In Proceedings of the International Conference on Embedded Software and Systems, Daegu, Republic of Korea, 14–16 May 2007; pp. 641–652.
36. Engelhardt, J.; Bouliane, N. Writing Netfilter Modules. July 2012. Available online: http://inai.de/documents/Netfilter_Modules.pdf (accessed on 7 March 2021).
37. Shostack, A. *Threat Modeling: Designing for Security*; John Wiley & Sons, Inc.: Indianapolis, IN, USA, 2014.
38. Kristiyanto, Y.; Ernastuti, E. Analysis of Deauthentication Attack on IEEE 802.11 Connectivity Based on IoT Technology Using External Penetration Test. *CommIT (Commun. Inf. Technol.) J.* **2020**, *14*, 45–51. [CrossRef]
39. van Hauser Heuse, M. thc-ipv6 (Version 3.8) [Computer Software]. 2020. Available online: https://github.com/vanhauser-thc/thc-ipv6 (accessed on 9 April 2021).
40. Making a synthesis emulation in IOT ERA possible Starbed5 Project Website. StarBED5 Project website /StarBED Equipment/. (n.d.). Available online: https://starbed.nict.go.jp/en/equipment/ (accessed on 20 March 2021).
41. Reed, D.P. User Datagram Protocol. IETF RFC 768. 1980. Available online: https://www.rfc-editor.org/info/rfc768 (accessed on 16 April 2021).
42. Netfilter Conntrack Performance Tweaking v0.8. Available online: https://wiki.khnet.info/index.php/Conntrack_tuning (accessed on 11 January 2021).
43. Bakai, D. DNS64perf++ Program. Available online: https://github.com/bakaid/dns64perfpp (accessed on 14 March 2021).
44. Lencse, G.; Bakai, D. Design and implementation of a test program for benchmarking DNS64 servers. *IEICE Trans. Commun.* **2017**, *100*, 948–954. [CrossRef]
45. Noormohammadpour, M.; Raghavendra, C.S. Datacenter traffic control: Understanding techniques and tradeoffs. *IEEE Commun. Surv. Tutor.* **2017**, *20*, 1492–1525. [CrossRef]
46. Gil, T.M.; Poletto, M. MULTOPS: A Data-Structure for Bandwidth Attack Detection. In Proceedings of the 10th USENIX Security Symposium (USENIX Security 01), Washington, DC, USA, 13–17 August 2001.
47. Mahajan, R.; Bellovin, S.M.; Floyd, S.; Ioannidis, J.; Paxson, V.; Shenker, S. Controlling high bandwidth aggregates in the network. *ACM SIGCOMM Comput. Commun. Rev.* **2002**, *32*, 62–73. [CrossRef]
48. Caswell, B.; Beale, J. *Snort 2.1 Intrusion Detection*; Elsevier Inc.: Amsterdam, The Netherlands, 2004.

49. OISF, Open Source Network Analysis and Threat Detection Software. 2020. Available online: https://suricata.io/ (accessed on 7 May 2021).
50. Al-Azzawi, A.; Lencse, G. The Possible Security Issues of the DS-Lite IPv6 Transition Technology, 26 August 2022, PREPRINT (Version 1). Available online: https://doi.org/10.21203/rs.3.rs-1972342/v1 (accessed on 9 May 2021).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.