Routing Loop Attacks using IPv6 Tunnels

Gabi Nakibly Michael Arov
National EW Research & Simulation Center
Rafael – Advanced Defense Systems
Haifa, Israel
{gabin,marov}@rafael.co.il

***Abstract*—IPv6 is the future network layer protocol for the Internet. Since it is not compatible with its predecessor, some interoperability mechanisms were designed. An important category of these mechanisms is automatic tunnels, which enable IPv6 communication over an IPv4 network without prior configuration. This category includes ISATAP, 6to4 and Teredo. We present a novel class of attacks that exploit vulnerabilities in these tunnels. These attacks take advantage of inconsistencies between a tunnel's overlay IPv6 routing state and the native IPv6 routing state. The attacks form routing loops which can be abused as a vehicle for traffic amplification to facilitate DoS attacks. We exhibit five attacks of this class. One of the presented attacks can DoS a Teredo server using a single packet. The exploited vulnerabilities are embedded in the design of the tunnels; hence any implementation of these tunnels may be vulnerable. In particular, the attacks were tested against the ISATAP, 6to4 and Teredo implementations of Windows Vista and Windows Server 2008 R2.**

I. Introduction

The sixth version of the Internet Protocol [1] (IPv6) is the future network layer protocol for the Internet. However, IPv6 is not backward compatible with IPv4: a host or a router supporting IPv4 can not process an IPv6 packet. Since the complete migration of the Internet to IPv6 is expected to take several years, if not decades, interoperability mechanisms that will enable the co-existence of IPv4 and IPv6 are required. One such mechanism is *tunneling*. Tunnels enable two IPv6 nodes to communicate over an IPv4-only network.

In general, tunnels operate as follows. Each tunnel has at least two end points. Each end point must be able to process both IPv4 and IPv6 packets and must possess an IPv4 address. To deliver an IPv6 packet over the tunnel, the ingress end point encapsulates the packet with an IPv4 header[1]. The source IPv4 address is that of the ingress end point and the destination IPv4 address is that of the intended egress end point. Consequently, each tunnel end point must have a routing table that associates each IPv6 destination address with an appropriate next-hop IPv4 address. The packet is then handled by the IPv4-only network as a normal IPv4 packet. When it reaches the egress end point, it strips the IPv4 header and continues to process the original IPv6 packet. The detailed operation of tunnels can be found in [2].

A tunnel in which the end points' routing tables need to be explicitly configured is called a *configured tunnel*. Tunnels of this type do not scale well, since every end point must be reconfigured as peers join or leave the tunnel. To alleviate this scalability problem, another type of tunnels was introduced – *automatic tunnels*. In automatic tunnels the egress entity's IPv4 address is computationally derived from the destination IPv6 address. This feature eliminates the need to keep an explicit routing table at the tunnel's end points. In particular, the end points do not have to be updated as peers join and leave the tunnel. In fact, the end points of an automatic tunnel do not know which other end points are currently part of the tunnel. However, all end points operate on the implicit assumption that once a packet arrives at the tunnel, its destination indeed is part of the tunnel. The primary purpose of this work is to show that this assumption poses a significant security vulnerability. We present a novel class of routing loop attacks that exploit this inconsistency in the routing state. This class is exemplified with five attacks.

The paper considers the three most prominent automatic tunnels to date: ISATAP [3], 6to4 [4], and Teredo [5]. These tunnels are supported by most major operating systems. They are the primary vehicles today for delivering IPv6 connectivity, even more than a native IPv6 access [6]. It is important to note that the attacks exploit design vulnerabilities in each of these tunnels, not implementation bugs. Consequently, *any IPv6 implementations may be vulnerable*. To validate the attacks, we have tested them against two Windows platforms – Windows Vista and Windows Server 2008 R2.

Although we think that the five attack examples have significant merit on their own, we believe that the main contribution of this paper is the introduction of a novel general class of routing loop attacks. This class of attacks gives a new insight into the secure design and deployment of automatic tunnels.

The rest of the paper is organized as follows. Section II gives a brief overview of the three automatic tunneling mechanisms considered in this work: ISATAP, 6to4 and Teredo. Section III surveys previous work on IPv6 tunnel vulnerabilities. Sections IV and V present the routing loop attacks and possible mitigation measures, respectively. Section VI concludes the paper.

[1] The Protocol field in the IPv4 header has the decimal value of 41, indicating that IPv6 header follows.

II. IPv6 Tunnels

In this section we give a brief overview of the three automatic tunnels considered in this paper: ISATAP, 6to4, and Teredo. These tunnels are complementary, rather than alternative, as they are designed for different network scenarios.

A. ISATAP

ISATAP – Intra-Site Automatic Tunneling Protocol [3] – is primarily designed to transport IPv6 packets between nodes in an IPv4 enterprise network. One of those nodes is a router which also has a native IPv6 interface. The router forwards IPv6 packets into or out of the tunnel. A node that belongs to an ISATAP tunnel has to know the IPv4 address of the router. If the IPv4 interface of a node has the address IP^4, the corresponding ISATAP interface is assigned a 64-bit ID having one of the following two formats: 0200:5EFE:IP^4 or 0000:5EFE:IP^4. The first one is used if IP^4 is non-private and the second one otherwise. Using this interface ID, a link-local address is constructed. The node probes the ISATAP router using the Neighbor Discovery Protocol [7], in order to discover the global prefix of the tunnel and to construct a global IPv6 address.

For each ISATAP interface on a node a set of locators is configured. A locator is a mapping of a node's IPv4 address to its associated interface. Only if a packet's IPv4 destination address and arrival interface match a locator configured for the ISATAP interface is the packet processed by that interface.

To send an IPv6 packet destined outside of the tunnel, the packet has to be encapsulated with an IPv4 header whose destination address is the router of the tunnel. If the packet is destined inside the tunnel, the IPv4 destination will be the 32 rightmost bits of the IPv6 destination address. In both cases the IPv4 source address is the IPv4 address of the encapsulator. At the egress end point the node first determines whether the packet matches a locator of the ISATAP interface. If there is a match, it verifies that one of the following two conditions holds:

1) the source IPv6 address corresponds to the source IPv4 address;
2) the source IPv4 address is the IPv4 address of the ISATAP router in the tunnel.

The first condition holds when the packet's source is part of the same ISATAP tunnel. The second one holds if the packet originates from outside of the tunnel.

B. 6to4

The 6to4 mechanism [4] is designed to transport IPv6 packets between IPv6 clouds or sites connected by the IPv4 Internet. It is assumed that each IPv6 site has an edge router with an IPv4 interface on the Internet side. The IPv4 address of that interface determines the IPv6 prefix of the entire site. If this address is IP^4, the 6to4 prefix of the site is 2002:IP^4/48. An edge router forwards IPv6 packets into and out of the 6to4 tunnel on behalf of the nodes in its site. An edge router that wishes to forward an IPv6 packet on the 6to4 tunnel to another site will encapsulate the packet with an IPv4 header having a destination address derived from the IPv6 destination address. The source address will be the IPv4 address of the ingress edge router. Before decapsulating the IPv4 header, the egress edge router verifies that if the source address is a 6to4 address, it corresponds to the IPv4 source address.

If the destination of the IPv6 packet is not a 6to4 address (does not have a 2002::/16 prefix) but a native IPv6 address, the edge router encapsulates and forwards the packet to a special router called a 6to4 relay. A 6to4 relay is connected to the IPv6 native Internet as well as to the IPv4 Internet. It forwards IPv6 packets into and out of the 6to4 tunnel on behalf of the nodes in the native IPv6 Internet. The 6to4 relays in the Internet have a predetermined anycast address: 192.88.99.0 [8]. A relay router advertises a route to 2002::/16 into the native IPv6 exterior routing domain. Hence, packets from the IPv6 native Internet destined to 6to4 sites will be forwarded to it. The relay encapsulates and forwards the packets on the 6to4 tunnel using the 6to4 destination address as described above. Before forwarding a packet to the native IPv6 network, the relay verifies that the packet's 6to4 source address corresponds to its IPv4 source address.

C. Teredo

The ISATAP and 6to4 tunnels encapsulate IPv6 packets with an IPv4 header. However, since most NATs cannot handle IP-in-IP packets, these mechanisms cannot work in the presence of a NAT. Hence a third mechanism was designed – Teredo [5]. Teredo enables nodes located behind one or more IPv4 NATs to obtain IPv6 connectivity by tunneling packets over UDP. A Teredo node performs a qualification procedure by interacting with an entity called a Teredo server located outside the NATs. Using this procedure, the node determines its external IPv4 address and UDP port assigned to it by the NATs. Each Teredo node is assigned an IPv6 Teredo address with the following format:

2001:0:<Teredo server>:<flags>
:<obfuscated external port>:<obfuscated external IP>,

where the <Teredo server> is the IPv4 address of the Teredo server, <flags> are various administrative flags not further discussed in this paper, <obfuscated external port> is the 1's complement of the external UDP port assigned to the Teredo node, and <obfuscated external IP> is the 1's complement of the external IPv4 address assigned to the Teredo node.

A Teredo node that wishes to send an IPv6 packet to another Teredo node behind a cone NAT (see section 5 in [9]) encapsulates the packet with UDP and IPv4 headers, where the destination address and port will be derived from the destination's Teredo address. The source address and port will be the originator's internal address and port. When a destination Teredo node is behind a restricted NAT (see section 5 in [9]), a packet from another node will be dropped by the NAT unless the destination node first sends out a packet to that node. For this purpose the originator node sends an empty packet called a bubble to the destination Teredo node through its Teredo server. The bubble's IPv4 destination address is the server's address and the UDP destination port is 3544. The server receives the bubble and forwards it to the destination node using its Teredo address. This bubble passes the restricted NAT because a destination node previously sent a packet to its server during the initial qualification procedure. The destination node in turn will respond by sending its own bubble to the originator of the first one. The purpose of the second bubble is to punch a hole in the destination's restricted NAT. Once the second bubble is accepted, a packet exchange can proceed.

III. Related Work

There is a large volume of work on the security benefits and shortfalls of IPv6, e.g., [10], [11], [12], [13]. However, only few works specifically address the security implication of IPv6 tunnels in general and IPv6 automatic tunnels in particular.

In [14], various security considerations of 6to4 are discussed. It details attacks that can be mounted either on nodes participating in a 6to4 tunnel or on nodes outside of it. The main vulnerability discussed in [14] is that IPv6 source address spoofing can be more easily carried out by directing the packet through a 6to4 router or relay. In this case the spoofed packet is more difficult to trace. Ref. [14] also reemphasizes known security checks that needs to be carried out before encapsulating or decapsulating an IPv6 packet with an IPv4 header.

In [15] the authors discuss security concerns for IP tunnels in general. They first call attention to the possibility that tunneled traffic can bypass network-based security measures if they are not configured correctly. The authors also note the difficulty of identifying and filtering such traffic. They also point out the increased exposure of tunnel end points to attacks if the tunnel creates an opening in the NAT or if the tunnel's address is more predictable than a native one.

In [16], an amplification attack has been demonstrated using routing header of type 0. This header sets intermediate way points for the packet. The attack forced a packet to loop between two routers by setting their addresses alternately many times in the routing header. This attack led to the deprecation of this type of routing header [17]. The effects of this attack are similar to those of the attacks described in this paper.

IV. Routing Loop Attacks

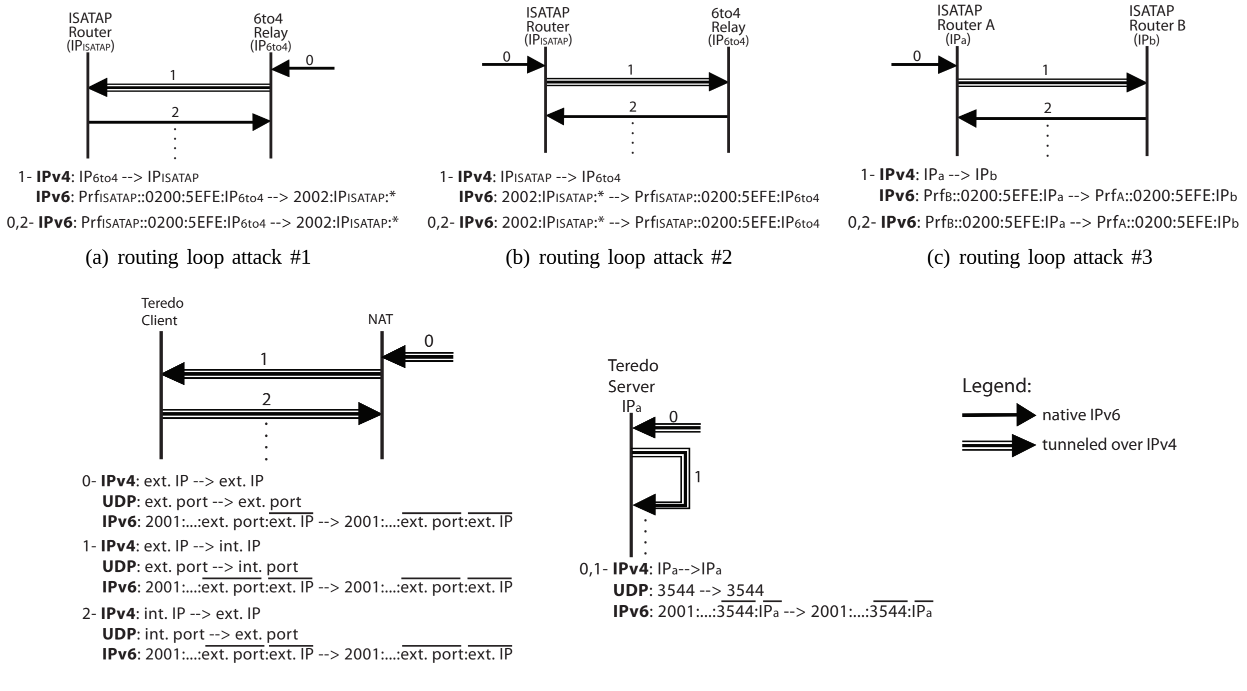
We now present the new class of attacks while exemplifying it with five routing loop attacks. Attacks in this class take advantage of inconsistencies between a tunnel's overlay IPv6 routing state and the native IPv6 routing state. More specifically, they exploit the fact that each end point in an automatic tunnel is ignorant of the other nodes that are currently participating in the tunnel. The attacker exploits this by crafting a packet which is routed over a tunnel to a node that is not participating in that tunnel. This node forwards the packet out of the tunnel to a native IPv6 network. In that network, the packet is routed back to the ingress point that forwards it back into the tunnel. Consequently, the packet will loop in and out of the tunnel. We shall refer to the nodes that forward the packet in and out of the tunnel as the victims of the attack.

A loop terminates only when the Hop Limit [1] field in the IPv6 header of the packet is zeroed out. The maximum value that can be assigned to this field is 255. Note that when the packet is tunneled over IPv4 routers, the Hop Limit does not decrease. Every attack packet will traverse each hop along the loop $255/N$ times, where N is the number of IPv6 routers on the loop. As a result, the loops can be used as traffic amplification tools with a ratio of $255/N$. The number of IPv6 routers on the loop is determined by the type of attack and by the positions of the two victims. The closer the two victims are, the larger the amplification ratio will be.

All the attacks described here were tested and verified against implementations of the tunnels in Windows Vista and Windows Server 2008 R2. For each attack we describe its applicability in practical network settings. In particular, we note that all attacks are initiated with a packet having a spoofed source address. As such they might be foiled by proper egress filtering measures deployed close to the attacker's location.

A. Attack #1: 6to4 Relay to ISATAP Router

The two victims of this attack are a 6to4 relay and an ISATAP router. Let IP_{ISATAP} and IP_{6to4} denote the IPv4 address of the ISATAP router and the 6to4 relay, respectively. Let Prf_{ISATAP} denote the IPv6 64-bit prefix of the ISATAP tunnel. The attack is depicted in Figure 1(a). It is initiated by sending an IPv6 packet (packet 0 in Fig. 1(a)) to a 6to4 destination address with an embedded router address of IP_{ISATAP}, i.e., the destination address begins with 2002:IP_{ISATAP}::/48. The source address of the packet is an ISATAP address with Prf_{ISATAP} as the prefix and IP_{6to4} as the embedded IPv4 address. As the destination address is 6to4, the packet will be routed over the IPv6 network to the closest 6to4 relay. The relay receives the packet through its IPv6 interface and processes it as a normal IPv6 packet that needs to be delivered to the appropriate 6to4 site. Hence, the packet is forwarded over the relay's IPv4 interface with an IPv4 header having a destination address derived from the IPv6 destination, i.e., IP_{ISATAP}. The source address is the address of the 6to4 relay, IP_{6to4}. The packet (packet 1 in Fig. 1(a)) is routed over the IPv4 network to the ISATAP router. The router receives


ISATAP Router (IPISATAP)
6to4 Relay (IP6to4)
0
1
2
1- IPv4: IP6to4 --> IPISATAP
IPv6: PrfISATAP::0200:5EFE:IP6to4 --> 2002:IPISATAP:*
0,2- IPv6: PrfISATAP::0200:5EFE:IP6to4 --> 2002:IPISATAP:*
ISATAP Router (IPISATAP)
6to4 Relay (IP6to4)
0
1
2
1- IPv4: IPISATAP --> IP6to4
IPv6: 2002:IPISATAP:* --> PrfISATAP::0200:5EFE:IP6to4
0,2- IPv6: 2002:IPISATAP:* --> PrfISATAP::0200:5EFE:IP6to4
ISATAP Router A (IPa)
ISATAP Router B (IPb)
0
1
2
1- IPv4: IPa --> IPb
IPv6: PrfB::0200:5EFE:IPa --> PrfA::0200:5EFE:IPb
0,2- IPv6: PrfB::0200:5EFE:IPa --> PrfA::0200:5EFE:IPb
Teredo Client
NAT
0
1
2
0- IPv4: ext. IP --> ext. IP
UDP: ext. port --> ext. port
IPv6: 2001:...:ext. port:ext. IP --> 2001:...:ext. port:ext. IP
1- IPv4: ext. IP --> int. IP
UDP: ext. port --> int. port
IPv6: 2001:...:ext. port:ext. IP --> 2001:...:ext. port:ext. IP
2- IPv4: int. IP --> ext. IP
UDP: int. port --> ext. port
IPv6: 2001:...:ext. port:ext. IP --> 2001:...:ext. port:ext. IP
Teredo Server IPa
0
1
0,1- IPv4: IPa-->IPa
UDP: 3544 --> 3544
IPv6: 2001:...:3544:IPa --> 2001:...:3544:IPa
Legend:
native IPv6
tunneled over IPv4

(a) routing loop attack #1

(b) routing loop attack #2

(c) routing loop attack #3

(d) routing loop attack #4

(e) routing loop attack #5

Fig. 1. Illustrations of the various routing loop attacks

the packet on its IPv4 interface. It processes the packet as a regular IPv4 packet that originates from one of the end points of the ISATAP tunnel. Since the IPv4 source address corresponds to the IPv6 source address, the packet will be decapsulated. Since the packet's IPv6 destination is outside the ISATAP tunnel, the packet will be forwarded onto the native IPv6 interface. The forwarded packet (packet 2 in Fig. 1(a)) is identical to the original attack packet. Hence, it will be routed back to the closest 6to4 relay, in which the loop will start again.

The loop will stop once the packet traverses 255 hops on the native IPv6 network. Note that only the part of the loop between the ISATAP router and the 6to4 relay traverses an IPv6 network. The opposite direction goes over a 6to4 tunnel over an IPv4 network in which the Hop Limit does not decrease.

Applicability – There are some preconditions for the attack to succeed. First, IP_{ISATAP} must be a non-private address so the tunneled packet can be routed over the IPv4 Internet. Second, there must not be ingress filtering of protocol-41 packets at the ISATAP's site border. Third, the tunneled packet must arrive at the ISATAP router via an interface associated with one of the locators of the ISATAP interface. This condition is hardest to fulfill when the ISATAP router sits at the site's border and only its internal interface is associated with the ISATAP locator. However, if the site has more than one entry point the attacker may choose a victim 6to4 relay in a location so that packets sent by it will enter the ISATAP site not through the victim ISATAP router.

B. Attack #2: ISATAP Router to 6to4 Relay

The two victims in this attack are again a 6to4 relay and an ISATAP router, but here they have swapped roles. This time the ISATATP router accepts the attack packet and forwards it on its ISATAP tunnel to the 6to4 relay, which decapsulates it and forwards it back to the ISATAP router on the IPv6 network. Let IP_{ISATAP}, IP_{6to4} and Prf_{ISATAP} be the same as above. The attack is depicted in Figure 1(b). This attack is initiated by sending an IPv6 packet (packet 0 in Fig. 1(b)) with a destination ISATAP address having Prf_{ISATAP} as the prefix and IP_{6to4} as the embedded IPv4 address. The source address of the packet is a 6to4 address with a router having the IP_{ISATAP} address , i.e., the destination address begins with 2002:IP_{ISATAP}::/48. The packet will be routed over the IPv6 network to the ISATAP router. The router receives the packet through its IPv6 interface and processes it as a normal IPv6 packet that needs to be delivered to the appropriate end point in the ISATAP tunnel. Hence, the packet is forwarded over the router's IPv4 interface with an IPv4 encapsulation having a destination address derived from the IPv6 destination , i.e., IP_{6to4}. The source address is the address of the ISATAP router, IP_{ISATAP}. The packet (packet 1 in Fig. 1(b)) is routed over the IPv4 network to the 6to4 relay. The relay receives the packet on its IPv4 interface. It processes the packet as a normal IPv4 packet that originates from one of the end points of the 6to4 tunnel. Since the IPv4 source address corresponds to the IPv6 source address, the packet will be admitted and decapsulated. Since the packet's IPv6 destination is outside the 6to4 tunnel, the packet will be forwarded out on the native IPv6 interface. The forwarded packet (packet 2 in Fig. 1(b)) is identical to the original attack packet. Hence, it will be routed back to the ISATAP router, in which the loop will start again.

In this attack the Hop Limit field will decrease only when the packet traverses the IPv6 network from the 6to4 relay to the ISATAP router. In the opposite direction the packet goes over an ISATAP tunnel over an IPv4 network, in which the Hop Limit does not decrease.

Applicability – First IP_{ISATAP} must be a non-private address so it can be processed by the 6to4 relay. Second, if Neighbor Unreachability Detection [7] is employed by the ISATAP router, it will discover that the 6to4 relay is not part of its ISATAP link. However, this will not stop the loop. The entry that corresponds to the 6to4 relay in the Neighbor Cache will indeed be removed. However, upon receipt of the looped attack packet the entry is recreate and the packet is forwarded immediately as address resolution is not needed in an ISATAP link. The NUD procedure will only enhance the attack by repeatedly sending probe packets to the 6to4 relay.

C. Attack #3: ISATAP Router to ISATAP Router

The two victims in this attack are two ISATAP routers – router A and router B – having addresses IP_a and IP_b, respectively. Let Prf_A and Prf_B be the prefixes of the ISATAP tunnels of router A and router B, respectively. Note that the two routers do not participate in the same ISATAP tunnel. However, they may reside at the same or different sites. The attack is depicted in Figure 1(c). It is initiated by sending an IPv6 packet (packet 0 in Fig. 1(c)) with a destination ISATAP address having Prf_A as the prefix and IP_b as the embedded IPv4 address. The source address of the packet is an ISATAP address having Prf_B as the prefix and IP_a as the embedded IPv4 address. The packet will be routed over the IPv6 network to router A. The router receives the packet through its IPv6 interface and processes it as a normal IPv6 packet that needs to be delivered to the appropriate end point of its ISATAP tunnel. The fact that the source address is also an ISATAP address does not matter here; the important thing is that the packet originated outside of the tunnel A. Hence, the packet is forwarded over the router's IPv4 interface with an IPv4 encapsulation having a destination address derived from the IPv6 destination , i.e., IP_b. The source address is the address of the router A, IP_a. The packet (marked with 1 in Fig. 1(c)) is routed over the IPv4 network to router B. The router receives the packet on its IPv4 interface. It processes the packet as a regular IPv4 packet that originates from one of the end points of its tunnel. Since the IPv4 source address corresponds to the IPv6 source address, the packet will be decapsulated. The packet's IPv6 destination is outside of router B's tunnel; hence the packet is forwarded out onto the IPv6 interface. The forwarded packet (packet 2 in Fig. 1(c)) is identical to the original attack packet. Hence, it will be routed back to router A, in which the loop will start again.

In this attack the Hop Limit field will decrease only when the packet traverses the IPv6 network from router B to router A. In the opposite direction the packet goes over an ISATAP tunnel over an IPv4 network, in which the Hop Limit does not decrease.

Applicability – The preconditions of the first attack apply here as well, unless the routers reside at the same site. In that case their addresses may be private.

D. Attack #4: Teredo Client to NAT

This attack exploits a Teredo tunnel. The two victims are a forwarding node that employs Teredo for its own IPv6 connectivity and its closest NAT. Such a forwarding node may be a router, a firewall, a Mobile IP home agent etc. We assume that the NAT is of type cone and it supports hair-pin routing with source address translation. These two assumptions are based on two requirements, REQ-8 and REQ-9, included in a Best Current Practice published by the IETF [18]. The attack is depicted in Figure 1(d). It is initiated by sending an IPv6 packet over the Teredo tunnel (packet 0 in Fig. 1(d)). The packet's destination IPv4 address and UDP port are the same as the source IPv4 address and UDP port. They are equal to the external IPv4 address and UDP port of the client. The IPv6 destination and source addresses are Teredo addresses, denoted by IP^d_{Teredo} and IP^s_{Teredo}, respectively, where the fields <obfuscated external port> and <obfuscated external IP> in both addresses are identical and equal to the 1's complement of the Teredo client's external port and address, respectively. The fields <Teredo server> or <flags> in those addresses should be different. Although their values are not important, they must not be equal to the respective fields in the client's Teredo address. Consequently, IP^d_{Teredo} and IP^s_{Teredo} are not equal to the client's Teredo address.

Having a state associated with the client following the initial qualification procedure and being of type cone, the NAT will not filter the attack packet and will pass it to the internal network while translating the destination IPv4 address and UDP port to the internal address and port of the client (packet 1 in Fig. 1(d)). The packet reaches the client over its IPv4 interface. The IPv4 source address and port of the packet correspond to the IPv6 source Teredo address; hence the client will admit the packet and remove the IPv4 and UDP headers. Since IP^d_{Teredo} is not the address of the client and the client is in forwarding mode, the client forwards the packet back to the network through its Teredo interface (packet 2 in Fig. 1(d)). The packet is encapsulated again with IPv4 and UDP headers, while the destination address and port are derived from IP^d_{Teredo}. Namely, they are equal to the client's external address and port. The source address and port are the client's internal address and port. Since the NAT is assumed to support hair-pin routing, when the packet reaches the NAT it will be routed back to the internal network. The destination address and port will be translated to the client's internal address and port. Since the NAT supports source address translation, the source address and port will be translated to the client's external address and port. The resulting packet is identical to the previous packet (packet 1 in Fig. 1(d)). Hence, it will be routed back to the client, in which the loop will start again.

In this attack the Hop Limit field will decrease only when the packet traverses the Teredo client. Only then is the packet handled by an IPv6 stack. In all the other hops on the loop, including the NAT, only IPv4 processing takes place.

Applicability – We note that in some network cases proper ingress filtering measures at the site, such as reverse path forwarding [19], may prevent the initial attack packet from entering the site.

E. Attack #5: Teredo Server

This attack differs from the attacks above. First, it engages with only one victim, a Teredo server. Second, the loop is not formed by forwarding the same IPv6 packet over and over, but by creating a new packet over and over again. Hence, the lifetime of the loop is infinite and not limited by the Hop Limit field. These two differences make this attack the most violent of all the attacks described in this paper. Executing the attack on a victim will result in an immediate exhaustion of the victim's CPU resources and will bring it to a crawl.

The attack loop is formed by tricking a Teredo server to produce a bubble destined to itself upon receipt of another bubble. The attack is depicted in Figure 1(e). It is initiated by sending a bubble over the Teredo tunnel to the server (packet 0 in Fig. 1(e)). The bubble's destination IPv4 address and port are identical to its source IPv4 address and port. They are equal to the IPv4 address of the server and 3544, respectively. The IPv6 destination and source addresses are two distinct Teredo addresses, in both of which the fields <obfuscated external port> and <obfuscated external IP> are identical and equal to the 1's complement of the server's IP and port (3544). The server receives and processes the packet as a normal Teredo bubble. In particular, it verifies that the source IPv4 address and port correspond to the source IPv6 Teredo address. The server then creates a new bubble (packet 1 in Fig. 1(e)) destined to the IPv4 address and port as derived from the IPv6 destination address. The Teredo specification does not define a check to prevent this (see section 5.3.1. in [5]). Hence, the bubble will be destined to the server's IPv4 address and to port 3544. Since the new bubble is identical to the previous one, the loop starts again indefinitely.

Applicability – The initial attack packet has identical IPv4 source and destination addresses. Some operating systems, e.g. Linux, will automatically drop such packets upon arrival. Hence, a Teredo server deployed on such OSes is not vulnerable.

V. Mitigation Measures

We suggest some simple security measures to mitigate the attacks. These measures are to be applied at the potential victims. Common to all the attacks is that the victims admit and forward a packet which is eventually routed back to them. The proposed security measures are aimed at recognizing such packets and discarding them. Before a node forwards a packet, it must check its destination address to verify that there is no chance the packet will eventually loop back to it. To this end a node must be aware of any automatic tunneling mechanism – even those it does not employ – that might be used to loop the packet back to it as demonstrated in the previous section. In particular, the following conditions must hold before forwarding a packet:

1) If the destination address is an ISTATAP address, its last four octets must not be equal to an IPv4 address of one of the node's interfaces.
2) If the destination address is a 6to4 address, its 3-6 octets must not be equal to an IPv4 address of one of the node's interfaces.
3) If the destination address is a Teredo address, the field <obfuscated external IP> must not be equal to the 1's complement of an IPv4 address of one of the node's interfaces or to an IPv4 address which is mapped to that node by a NAT[2].

All these checks should be applied in every IPv6 node that might forward packets and is participating in at least one of these tunnels. For example, an ISTATP router that does not participate in a 6to4 or Teredo tunnel must still exercise all three checks. This implies that for any new automatic tunneling mechanisms that will be designed in the future, a corresponding security check should be added.

VI. Conclusions

In this paper we present a novel class of routing loop attacks that exploit the design of IPv6 automatic tunnels. Five attacks of this class which abuse ISATAP, 6to4, and Teredo are exhibited. The attacks exploit the inconsistencies between a tunnel's overlay IPv6 routing state and the native IPv6 routing state. Consequently, a carefully constructed packet will loop. In the first four attacks the loop is bounded by the Hop Limit field in the IPv6 header. However, the last attack is infinite since it causes a Teredo server to produce a new bubble packet on every loop. It is our opinion that further research may unearth more routing loop attacks of this class.

The proposed mitigation measures for such attacks are relatively simple however they require knowledge of other tunneling mechanisms that may not be employed by the defending node.

[2] Note that the externally mapped address is known to a Teredo client.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable suggestions on and corrections of this paper.

REFERENCES

[1] S. Deering and R. Hinden, “Internet Protocol, version 6 (IPv6) specification,” IETF RFC 2460, December 1998.

[2] E. Nordmark and R. Gilligan, “Basic transition mechanisms for IPv6 hosts and routers,” IETF RFC 4213, October 2005.

[3] F. Templin, T. Gleeson, and D. Thaler, “Intra-site automatic tunnel addressing protocol (ISATAP),” IETF RFC 5214, March 2008.

[4] B. Carpenter and K. Moore, “Connection of IPv6 domains via IPv4 clouds,” IETF RFC 3056, February 2001.

[5] C. Huitema, “Teredo: Tunneling IPv6 over UDP through network address translations (NATs),” IETF RFC 4380, February 2006.

[6] S. H. Gunderson, “Global IPv6 statistics - measuring the current state of IPv6 for ordinary users,” RIPE 57, October 2008.

[7] T. Narten *et al.*, “Neighbor discovery for IP version 6 (IPv6),” IETF RFC 4861, September 2007.

[8] C. Huitema, “An anycast prefix for 6to4 relay routers,” IETF RFC 3068, June 2001.

[9] J. Rosenberg *et al.*, “STUN - simple traversal of user datagram protocol (UDP) through network address translators (NATs),” IETF RFC 3489, March 2003.

[10] S. Convery and D.Miller, “IPv6 and IPv4 threat comparison and best-practice evaluation,” http://seanconvery. com/v6-v4-threats.pdf, March 2004.

[11] S. M. Bellovin, A. Keromytis, and B. Cheswick, “Worm propagation strategies in an IPv6 internet,” *;login:*, pp. 70–76, February 2006.

[12] P. M. Gleitz and S. M. Bellovin, “Transient addressing for related processes: Improved firewalling by using IPv6 and multiple addresses per host,” in *Proceedings of the Eleventh Usenix Security Symposium*, August 2001.

[13] M. Kaeo *et al.*, “IPv6 security technology paper,” *North American IPv6 Task Force (NAv6TF) Technology Report*, July 2006.

[14] P. Savola and C. Patel, “Security considerations for 6to4,” IETF RFC 3964, December 2004.

[15] J. Hoagland, S. Krishnan, and D. Thaler, “Security concerns with IP tunneling,” IETF Internet Draft draft-ietf-v6ops-tunnel-security-concerns-01, October 2008.

[16] P. Biondi and A. Ebalard, “Ipv6 routing header security,” in *CanSecWest Security Conference*, April 2007.

[17] J. Abley, P. Savola, and G. Neville-Neil, “Deprecation of type 0 routing headers in IPv6,” IETF RFC 5095, December 2007.

[18] F. Audet *et al.*, “Network address translation (NAT) behavioral requirements for unicast UDP,” IETF RFC 4787 (BCP 127), January 2007.

[19] F. Baker and P. Savola, “Ingress filtering for multihomed networks,” IETF RFC 3704, March 2004.