International Journal of Research in Advent Technology, Vol.2, No.5, May 2014
E-ISSN: 2321-9637

Protocol Transitioning through Tunneling- Worth or Risk

Ms.Khushboo.S.Sathawane[1], Prof.Sangram S.Dandge[2],
M.E.(II Year)- Department of Computer Science and Engg. P.R.M.I.T.R, Badnera-Amravati[1,]
Assistant Professor- Department of Computer Science and Engg. P.R.M.I.T.R, Badnera-Amravati [2]
Email: Khushboo.sathawane@gmail.com[1] , sangramdandge@gmail.com[2]

Abstract- The future network layer protocol for the Internet is IPv6 . 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.

Teredo is a service that enables hosts located behind one or more IPv4 NATs to obtain IPv6 connectivity by tunneling packets over IPv4 UDP. In this paper,we explain how IPv6 candidates located behind NATs can enlist the help of "Teredo servers" and "Teredo relays" to learn their "global addresses" and to obtain connectivity, and how clients, servers and relays can be organized in Teredo networks.This paper studies the security implications of Teredo. However, by tunneling IPv6 traffic over IPv4 UDP through the NAT and directly to the end node, Teredo raises some security concerns. Primary concerns include bypassing security controls, reducing defense in depth, and allowing unsolicited traffic. 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. 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 this paper we are Investigating whether this protocol transitioning mechanism is a worth or a risk.

Index Terms- IPV6 ;Teredo; Tunneling;, NAT;

1. INTRODUCTION

IPv6 is the next version of the Internet Protocol, and many hosts and networks are being upgraded to support this version and take advantage of its features. A part of the Internet that is expected to lag behind in IPv6 availability are the IPv4 Network Address Translation (NAT) devices used in many household and organizational networks. They are only infrequently updated or replaced, especially on small networks such as those found in residences. 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.[20] 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 header1. 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]. 1The Protocol field in the IPv4 header has the decimal value of 41,indicating that IPv6 header follows. 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 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.

The paper considers the three most prominent automatic tunnels to date: ISATAP [3], 6to4 [4], and Teredo [5]. However, transition mechanisms that tunnel IPv6 directly over IPv4, such as the Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) and 6to4, do not typically work through NATs.

2. PRELIMINARIES

2.1 What's New in IPV6

The IPv6 protocol brings a variety of new mechanisms to improve Internet reliability [7]. These include the following:

- Significantly large address space
- Simplified network management using stateless auto-configuration of nodes
- Routing efficiency due to use of fixed-length headers
- Reduction in network processor overhead due to reduced fragmentation
- Improved security (IPsec is built-in)
- Well-defined flow labels for Quality of Service
- End-to-end address transparency

2.2 NAT: Network Address Translation

Basic Network Address Translation (Basic NAT) is a method by which IP addresses are mapped from one group to another, transparent to end users. Network Address Port Translation, or NAPT, is a method by which many network addresses and their TCP/UDP ports are translated into a single network address and its TCP/UDP ports. Together, these two operations, referred to as traditional NAT[8], provide a mechanism to connect a realm with private addresses to an external realm with globally unique registered addresses. The need for IP Address translation arises when a network's internal IP addresses cannot be used outside the network either for privacy reasons or because they are invalid for use outside the network. Network topology outside a local domain can change in many ways. Customers may change providers, company backbones may be reorganized, or providers may merge or split. Whenever external topology changes with time, address assignment for nodes within the local domain must also change to reflect the external changes. Changes of this type can be hidden from users within the domain by centralizing changes to a single address translation router. Basic Address Translation allows hosts in a private network to transparently access the external network and enable access to selected local hosts from the outside. It is mandatory that all requests and responses pertaining to a session be routed via the same NAT router. One way to ascertain this would be to have NAT based on a border router that is unique to a stub domain, where all IP packets either originated from the domain or are destined for the domain. There are other ways to ensure this with multiple NAT devices.The NAT solution has the disadvantage of taking away the endto-end significance of an IP address, and making up for this with an increased state in the network. As a result, with a NAT device enroute, end-to-end IP network level security assured by IPSec cannot be assumed to apply to end hosts. The advantage of this approach, however, is that it can be installed without changes to hosts or routers. Protocol Structure NAT is a procedure, not a structured protocol.

2.3. Tunneling Mechanism

To minimize any dependencies during the transition, all the routers in the path between two IPv6 nodes do not need to support IPv6. This mechanism is called tunneling. Basically, IPv6 packets are placed inside IPv4 packets, which are routed through the IPv4 routers. The following figure illustrates the tunneling mechanism through routers (R) using IPv4[8].Different uses of tunneling in the transition are:

- Configured tunnels between two routers (as in the figure 1.)
- Automatic tunnels that terminate at the dual hosts

A configured tunnel is currently used in the Internet for other purposes, for example, the MBONE (the IPv4 multicast backbone). Operationally, it consists of configuring two routers to have a virtual point-to-point link between them over the IPv4 network. This kind of tunnel is likely to be used on some parts of the Internet for the foreseeable future.

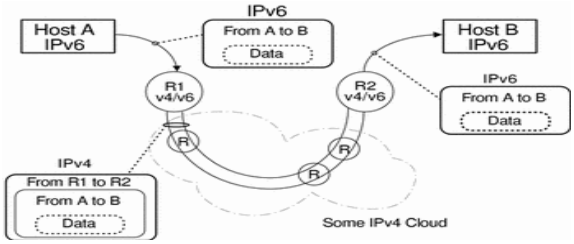


Figure 1 Tunneling Mechanism

3. RELATED WORKS

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.

3.1 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 IP4, the corresponding ISATAP interface is assigned a 64-bit ID having one of the following two formats: 0200:5EFE:IP4 or 0000:5EFE:IP4. The first one is used if IP4 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 [10], 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. 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.

3.2 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 IP4, the 6to4 prefix of the site is 2002:IP4/48. 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.

3.3 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.

4. THE TEREDO

Microsoft is making a strong push for IPv6, and in response has developed a transition mechanism to address this issue. Fortunately, the mechanism was routed through IETF channels, and the IETF has published RFC 4380 as a standards-track individual submission. Originally the protocol was called Shipworm (after a species of mollusk that digs holes in ship hulls, analogous to what the protocol does with NAT devices). But the protocol has been renamed Teredo, after a common genera of shipworms (perhaps to avoid any negative connotation).

Teredo is already in use on the Internet. It is available in Windows Vista and Longhorn, where it is enabled by default. Teredo is also available in Windows XP SP2 and Windows 2003 SP1, although disabled by default. At least one third-party implementation of Teredo is available for UNIX and Mac® OS X.

For an IPv6-capable node behind an IPv4 NAT, the barrier to sending and receiving packets from IPv peers is that at least a portion of the network between the IPv6-capable node and the peer does not support IPv6. This includes at least the NAT. To resolve the problem, Teredo establishes an open-ended tunnel from the client, through the NAT, to a dual-stacked node on the Internet. IPv6 packets are tunneled through a single User Datagram Protocol (UDP) port on the NAT.

The use of Teredo has important security implications, and these implications are discussed in this paper. Little published research exists on this topic, other than the "Security Considerations" section of the Teredo RFC itself. John Spence of Command Information includes a brief mention of Teredo in the "IPv6 Security and Security Update,"[3] and suggests disabling it since it "defeats IPv4 NAT."

4.1.Teredo Tunnels

In the Teredo case the tunneling is UDP, so all IPv6 Teredo packets are composed of an IPv4 packet header and a UDP transport header, followed by the IPv6 packet as the UDP payload[12]. Teredo uses a combination of ICMPv6 [13] message exchanges to set up a connection and tunneled packets encapsulated using an outer IPv4 header and a UDP header, and it contains the IPv6 packet as a UDP payload.

The exact nature of the packet exchange in setting up a Teredo connection depends on the nature of the

NAT device that sits in front of the Teredo client. Figure 2 shows an example packet exchange that Teredo uses when the client is behind a Restricted NAT.

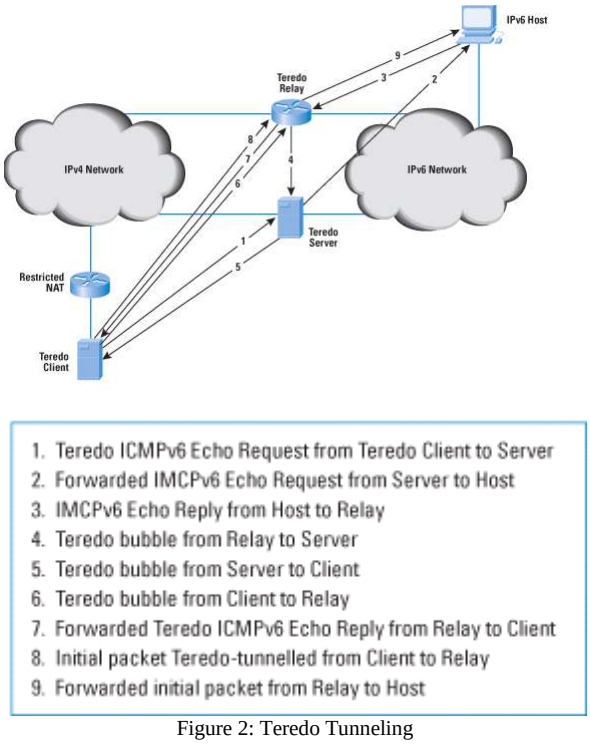


Figure 2: Teredo Tunneling

5. WORKING OF TEREDO

Teredo works by tunneling IPv6 over an IPv4 UDP port for at least the portion of the network that is Pv4 only. Teredo has a high degree of automatic tunnel setup.

5.1 Teredo components

The Teredo framework consists of three basic components: clients, relays, and servers. Teredo clients are nodes seeking to use Teredo to reach a peer on the IPv6 Internet. For example, a node may need to reach an IPv6-only server. Clients are dual-stack (IPv4 and IPv6) nodes that are "trapped" behind one or more IPv4 NATs. Teredo clients always send and receive Teredo IPv6 traffic tunneled in UDP over IPv4 (see Figure 3).

In this paper, ports refer specifically to IPv4 UDP ports unless otherwise noted.

Teredo relays serve as routers to bridge the IPv4 and IPv6 Internets for Teredo nodes. IPv6 native packets are encapsulated for transmission over the IPv4 Internet (including the client); when packets are received from the IPv4 Internet, they are decapsulated into native IPv6 packets for the IPv6 Internet.

Teredo servers help clients set up tunnels to IPv6 nodes, determining their Teredo address and whether their NAT is compatible with Teredo. Like relays, Teredo servers sit on both the IPv4 and IPv6 Internets, but do not serve as a general relay. Teredo servers pass along packets to and from the client, but only messages that pertain to the functioning of the Teredo protocol; they do not pass along data packets.

The Teredo servers are generally statically configured on the client. For example, Windows nodes by default use "teredo.ipv6.microsoft.com" as their

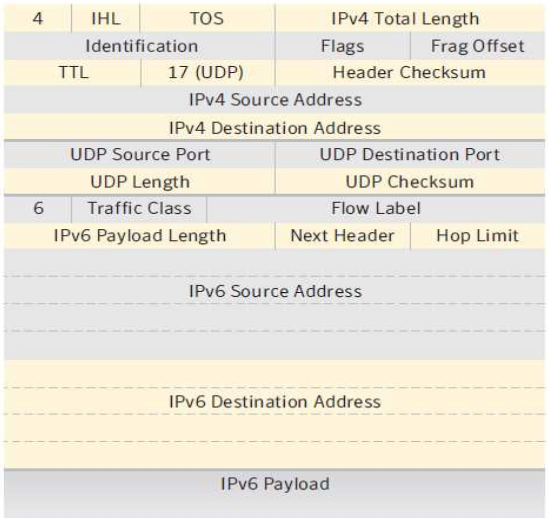


Figure 3. Teredo encapsulates IPv6 packets in UDP over IPv4 when packets are routed as IPv4.

server; this currently resolves to four servers (or at least four IPv4 addresses) that Microsoft maintains.

The standard port on which the Teredo servers listen is UDP port 3544. Both clients and relays can use any UDP port for their Teredo service, so their UDP service port could be ephemeral. Because the client is behind an IPv4 NAT, the external port number of its Teredo service is, in general, not the same as the local port that is listened on. However, the Teredo protocol tries to keep that external port number stable since it is the port to which the relays need to connect. Servers are specifically designed to be stateless, so a large number of clients can be accommodated.

5.2. Teredo setup

Before packets can be sent to and from remote IPv6 nodes, some tunnel setup communication occurs. The phases are as follows:

1. The client completes a qualification procedure (see "Qualification procedure" section) to establish aTeredo address.
2. The client determines which relay to use (see "Packet relaying and peer setup for non-Teredo peers" section) for a given IPv6 peer node. This phase may involve a procedure to set up the NAT for traffic from the relay ("Bubble packets and creating a NAT hole"

section).

3. .A packet is sent via a relay The first phase needs to be conducted only once (for each time Teredo is activated on the client). The next two phases are completed for each peer that was not recently used. After that setup, it is just a matter of sending the packet via the relay. The relaying and per-peer setup take a special form when the remote peer is also a Teredo IPv6 address ("Packet relaying and peer setup for Teredo peers" section). A special provision (outside the scope of this paper) allows IPv6 nodes behind the same NAT to find each other by using an optional local client discovery procedure

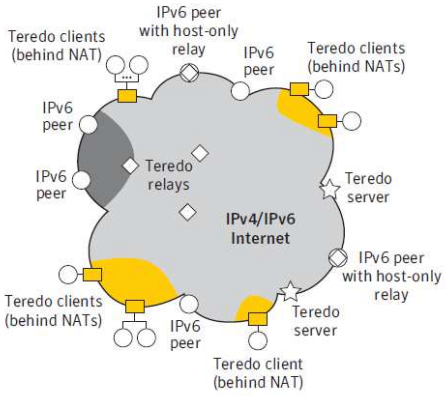


Figure 4. A Teredo microcosm, including key Teredo components, native IPv6 nodes, and IPv4 NATs. The cloud represents the Internet, where the yellow areas are IPv4 only, the dark gray area is IPv6 only, and the mixed gray area supports both. The interior of the cloud represents Internet routers and infrastructure.

5.3 Teredo addresses

Teredo clients (and only Teredo clients) receive a specially formatted IPv6 address called a Teredo address. Addresses contain enough information for a relay to reach a client (see Figure 5).

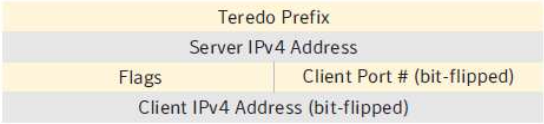


Figure 5. The format of a Teredo address. Like all IPv6 addresses, it is 128 bits (16 octets) long.

The prefix is standard for Teredo addresses; 2001:0000::/32 was recently assigned. You might see other prefixes, such as 3ffe:831f::/32, used in Teredo components that predate the current assignment.The second 32 bits of the address correspond to the IPv4 address of the client's Teredo server. This part of the address tells remote nodes which server is assisting the client with communication setup.The bottom 48 bits correspond to the client's external address and Teredo service port. This part of the address indicates to relays where to send packets destined directly for the client. To protect these two fields from any NAT translation, all of the bits in these fields are reversed. The flags field is 16 bits, but only 1 bit is assigned by the RFC. The top bit is the "cone bit." If set, the cone bit indicates that the node is behind a pure cone NAT; if unset, it indicates the node is behind a restricted NAT. The rest of the bits in the field should be set to 0.

5.4 Origin data

When a Teredo server sends an IPv6 packet to one of its clients on behalf of an IPv4 host, it adds additional data between the UDP encapsulation and the IPv6 packet. This is the origin data (see Figure 6) and reflects the IPv4 address and port number that it acts on behalf of. (The RFC calls this origin encapsulation.)

As in Teredo addresses, the port number and address have all their bits reversed. The client concludes that extra data is present, as the first nibble after the UDP header is 0 instead of 6 (the version number from the IPv6 header).

The qualification procedure determines if a client can use the Teredo service and establishes the Teredo address. For example, a client cannot use the Teredo service if it is behind a symmetric NAT.

Figure 6. The format of the origin data, which is located below the encapsulated IPv6 packet. Qualification procedure

A portion of the Neighbor Discovery Protocol (NDP, RFC 2461) is used, with the Teredo server acting as the router.

During qualification, the client sends Router Solicitations (RSs); the server then sends back Router Advertisements (RAs) plus an origin data block (see "Origin data" section) in response. Both the RA and RS messages are encapsulated ICMPv6 packets. Since the RA is sent in response to an RS from the client's Teredo service port, the origin data reveals to the client its external Teredo address and port number. That data becomes part of the client's Teredo address.

Qualification begins with the client sending an RS to the server with the cone bit set. Setting the cone bit means the client is trying to determine if it is behind a pure cone NAT. When it sees the cone bit is set, the server sends the RA from a different IPv4 address to the one that it received the packet on. If the client is indeed behind a pure cone NAT, the NAT passes the packet to the client. However, if the client is behind a restricted NAT, the

NAT will not pass the packet to the client because the source is not a previous destination.

If the client receives the RA, it knows it is behind a pure cone NAT and concludes qualification. The client forms a Teredo address with the cone bit on.

However, if the RA is not received, it could be due to packet loss. So after T seconds of waiting (default is 4 seconds), the client tries again, up to N times total (default is 3).If the client still doesn't receive the RA, it tentatively assumes it is behind a restricted NAT and sends the RS with the cone bit unset. Since the cone bit is off, the server responds from the same address as it received the RA from. If this attempt does not succeed after N times of waiting T seconds, the client gives up, assuming a server connectivity problem.

5.5 Secure qualification

Teredo provides an option for the qualification procedure to be "secured" by adding authentication data (the RFC calls this authentication encapsulation) between the UDP header and the origin data with the encapsulated packet. Without this data, the client would not know that the response is sent from the real server (versus having received a randomly sent RA). The authentication data takes the format shown in Figure 7.

Here, the client identifier and authentication value are optional and have their specific length indicated in one-octet fields. The nonce value is always present and always 8 octets in length; it is a random number chosen by the client and repeated by the server in the response.

This simple measure establishes (with high probability)

that if there is an attacker, it is at least on-path between the client and the server. Figure 8 shows the layout of the authentication data in this simple case.

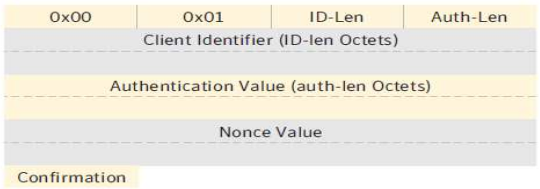


Figure 7. The general format of the authentication data. In secure qualification, this data is positioned after the UDP header.

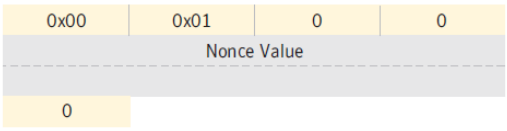


Figure 8. Authentication data at it simplest, when there is no client identifier or authentication value.

The authentication value (if present) is a keyed cryptographic hash of most of this header, the origin header, and the IPv6 packet. By default, the hash is based on HMAC and SHA1. This measure provides stronger protection against tampering and can help ensure that the server is the one intended. The RFC is not specific on the value of the client identifier, but it can relate to the authentication value. The confirmation byte is non-0 if the client should obtain a new key.

5.6 Bubble packets and creating a NAT hole

Teredo makes use of what the Teredo RFC refers to as bubble packets. These are simple IPv6 packets with no IP payload; that is, the IP payload length is 0, and the Next Header field has the value 59 (No Next Header).

These packets manipulate a NAT into allowing the real traffic. A typical use is when a relay needs to send a packet to a Teredo client, but the client is behind a restricted NAT (as evidenced by the cone bit being unset), and the relay is not a previous (recent) peer with that client. This circumstance prevents direct communication, so the following bubble-to-open procedure (see Figure 9) takes place:

1. The relay sends an encapsulated bubble packet to the Teredo client's server with the IPv6 destination set to the Teredo peer. The server address is extracted from the client's Teredo address.
2. The server passes the bubble along to the Teredo client, adding origin data (the IPv4 address and port of the relay).
3. The NAT receives the packet and passes it on to the client. The NAT allows this because the client and server communicate on a regular basis.
4. Upon receipt of the bubble, the client sends an encapsulated bubble to the address and port in the origin data (the relay).
5. The encapsulated bubble is received by the NAT and forwarded to the relay. The NAT now sees the relay as a recent peer and allows incoming packets from it.

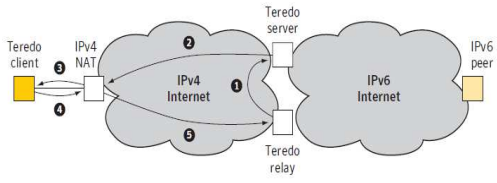


Figure 9. The bubble-to-open procedure opens a restricted NAT's port to a relay. To do this, the relay asks the server to ask the client to send it a bubble packet

Thus, Teredo provides an on-demand service that allows packets from arbitrary Internet hosts to be passed to the client. For a Teredo client's service port, the service makes a

restricted NAT resemble a pure cone NAT. This concept is explored further in "Teredo implications on ability to reach a host through a NAT" section.

In any case, the RFC requires rate limiting of the bubbles sent to a specific peer, to protect against flooding. A bubble SHOULD NOT be sent if one was sent in the last 2 seconds or if four were sent in the past 5 minutes without receiving any direct responses.

5.7 Teredo bubble packets format

A Teredo bubble packet is typically sent to create or maintain a NAT mapping and consists of an IPv6 header with no IPv6 payload. Figure 10 shows the Teredo bubble packet[1].

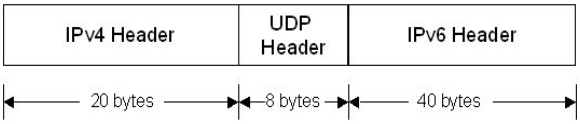


Figure 10 . Teredo bubble packet

In the IPv6 header, the Next Header field is set to 59, indicating that there is no payload present.

6. ROUTING LOOP ATTACKS

We now present the new class of attacks while exemplifying it with five routing loop attacks[14]. 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 [15] 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. 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.

6.1 Attack #1: 6to4 Relay to ISATAP Router

The two victims of this attack are a 6to4 relay and an ISATAP router. Let IPISATAP and IP6to4 denote the IPv4 address of the ISATAP router and the 6to4 relay, respectively. Let PrfISATAP 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. 11(a)) to a 6to4 destination address with an embedded router address of IPISATAP , i.e., the destination address begins with 2002:IPISATAP ::/48. The source address of the packet is an ISATAP address with PrfISATAP as the prefix and IP6to4 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., IPISATAP . The source address is the address of the 6to4 relay, IP6to4. The packet (packet 1 in Fig. 11(a)) is routed over the IPv4 network to the ISATAP router. 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 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. 11(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.

6.2 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 IPISATAP , IP6to4 and PrfISATAP be the same as above. The attack is depicted

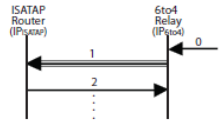
in Figure 11(b). This attack is initiated by sending an IPv6 packet (packet 0 in Fig. 11(b)) with a destination ISATAP address having PrfISATAP as the prefix and IP6to4 as the embedded IPv4 address. The source address of the packet is a 6to4 address with a router having the IPISATAP address , i.e., the destination address begins with 2002:IPISATAP ::/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., IP6to4. The source address is the address of the ISATAP router, IPISATAP . The packet (packet 1 in Fig. 11(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. 11(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.

6.3. Attack #3: ISATAP Router to ISATAP Router

The two victims in this attack are two ISATAP routers – router A and router B – having addresses IPa and IPb, respectively. Let PrfA and PrfB 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. 11(c)) with a destination ISATAP address having PrfA as the prefix and IPb as the embedded IPv4 address. The source address of the packet is an ISATAP address having PrfB as the prefix and IPa 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., IPb. The source address is the address of the router A, IPa. The packet (marked with 1 in Fig. 11(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. 11(c)) is identical to the original attack packet. Hence, it will be routed back to router A, in which the loop will start again.

6.4 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 [16]. The attack is depicted in Figure 11(d). It is initiated by sending an IPv6 packet over the Teredo tunnel (packet 0 in Fig. 11(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 IPdTeredo and IPsTeredo,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 Teredoclient's external port and address, respectively. Consequently, IPd Teredo and IPs Teredo are not equal to the client's Teredo address. Having a state associated with the client



(a) routing loop attack #1

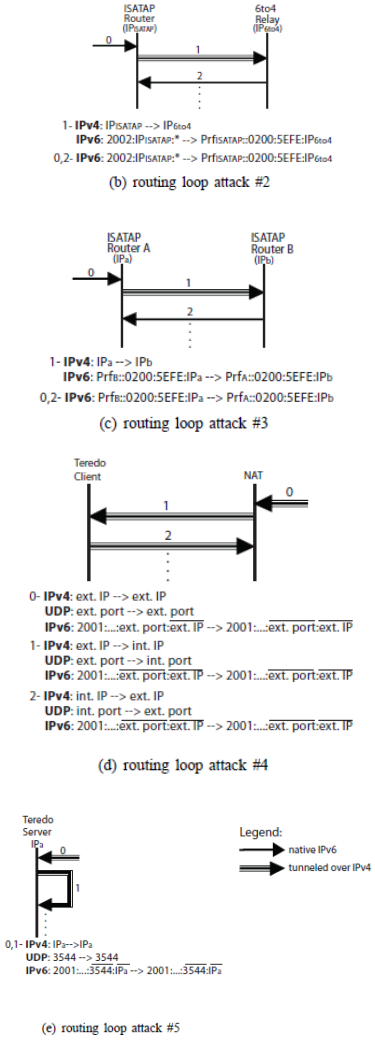


Figure 11 . Illustrations of the various routing loop attacks

following the initial qualification procedure and being of type cone, the NAT will not filter the attack packet and will pass it tothe internal network while translating the destination IPv4 address and UDP port to the internal address and port of the client (packet 1 in Fig. 11(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 IPd 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. 11(d)). The packet is encapsulated again with IPv4 and UDP headers, while the destination address and port are derived from IPdTeredo.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 willbe 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. 11(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 [17], may prevent the initial attack packet from entering the site.

6.5 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 11(e). It is initiated by sending a bubble over the Teredo tunnel to the server (packet 0 in Fig. 11(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. 11(e)) destined to the IPv4 address and port as derived from the IPv6 destination address. The Teredo specification does not define a check to preventthis (see section 5.3.1. in [18]). 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.

7. MITIGATION MEASURES

Some simple security measures could be opted 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 NAT2. 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.

8. CONCLUSIONS

In this paper, some of the significant security implications of the protocol are highlighted; that is, ways in which Teredo positively or negatively impacts the IPv4 and IPv6 portions of the Internet. Teredo provides a way for dual-stack nodes that do not have direct IPv6 connectivity (due to being located behind an IPv4 NAT) to communicate with remote IPv6 nodes. This approach promotes the earlier use of IPv6 for the large number of hosts "stuck" behind NATs. Servers see a client's intended IPv6 peers, so one should use only trusted servers; this is a concern mainly if the server setting is secretly switched to a malicious server

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.

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.

.

REFERENCES

[1] Microsoft. "Teredo Overview." microsoft.com. http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/teredo.mspx

[2] Denis-Courmont, Rémi. Miredo. http://www.simphalempin.com/dev/miredo/

[3] Spence, John. "IPv6 Security and Security Update." NAv6TF/ARIN XV IPv6 Conference, April2005:http://www.nav6tf.org/documents/ arin-nav6tf apr05/6.IPv6_Security_Update_JS.pdf

[4] Jennings, Cullen. "NAT Classification Results using STUN." Internet Draft draft-jennings-midcom- stun-results-00. February 2004: http://www.employees.org/~fluffy/ietf/ draft-jennings-midcom-stun-results 00.html

[5] Davies, E., S. Krishnan, and P. Savola. IPv6 Transition/Co-existence Security Considerations. Internet Draft draft-savola-v6ops- security-overview-04.txt (work in progress). March 2006: http://ietfreport.isoc.org/idref/draft-ietf-v6ops-security-overview/

[6] Symantec. Symantec Internet Security Threat Report: Trends for January 06–June 06. Symantec white paper, Volume X, Sept 2006: http://www.symantec.com/specprog/threatreport /ent-whitepaper_ symantec_internet_security_threat_report_x_ 09_2006.en-us.pdf

[7] Ensuring a Smooth Transition to Internet Protocol Version 6 (IPv6) http://www.brocade.com/downloads/documents/ white_papers/GA-WP-1574-00.pdf

[8] Sponsor Source NAT is defined by IETF (http://www.ietf.org) in RFC 3022. Reference http://www.javvin.com/protocol/rfc3022.pdf Traditional IP Network Address Translator (Traditional NAT)

[9] Ebook: TCP/IP Illustrated – Volume 1 by-Kevin R.Fall & W. Richard Stevens

[10] Weaver, N. "How Many Ways to 0wn the Internet?: Towards Viable Worm Defenses:

http://www.cs.berkeley.edu/~nweaver/wormdefense.ppt

[11] Seely, Donn. A Tour of the Worm In Proceedings of the 1989 Winter USENIX Technical Conference, January 1989: http://securitydigest.org/phage/resource/seely.pdf

[12] Shiang-Ming Huang, Quincy Wu, Yi-Bing Lin "Tunneling IPv6 through NAT with Teredo Mechanism" Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on (Volume:2)

[13] C. Huitema, "Teredo: Tunneling IPv6 over UDP through NATs", Internet draft, draft-huitema-v6ops-teredo-02.txt (Work In Progress), March 2004.

[14] Gabi Nakibly Michael Arov "Routing Loop Attacks using IPv6 Tunnels "-WOOT'09 Proceedings of the 3rd USENIX conference onOffensive technologies Pages 7-7 USENIX Association Berkeley, CA, USA ©2009

[15] T. Narten et al., "Neighbor discovery for IP version 6 (IPv6),"IETF RFC 4861, September 2007

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

[17] F. Baker and P. Savola, "Ingress filtering for multihomed networks," IETF RFC 3704, March 2004.

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

[19] Teredo Overview- Microsoft Corporation Published: January 2003

[20] Dr. James Hoagland –"The Teredo Protocol: Tunneling Past Network Security and Other Security Implications", Principal Security Researcher Symantec Advanced Threat Research