Internet Engineering Task Force
Internet Draft
Expiration Date: July 2003

P. Savola
CSC/FUNET

January 2003

Security Considerations for 6to4

draft-savola-v6ops-6to4-security-02.txt

Status of this Memo

Abstract

The IPv6 interim mechanism 6to4 (RFC3056) uses automatic IPv6-over-IPv4 tunneling to interconnect IPv6 networks. The architecture includes Relay Routers and Routers, which accept and decapsulate IPv4 traffic from anywhere. There aren't many constraints on the embedded IPv6 packets, or where IPv4 traffic will be automatically tunneled to. These could enable one to go around access controls, and more likely, being able to perform proxy Denial of Service attacks using 6to4 relays or routers as reflectors. Anyone is also capable of spoofing traffic from non-6to4 addresses, as if it was coming from a relay, to a 6to4 node. This document discusses these issues in more detail and tries to suggest enhancements to alleviate the problems.

Table of Contents

1. Introduction

The IPv6 interim mechanism "6to4" [6TO4] specifies automatic IPv6-over-IPv4 tunneling to interconnect isolated IPv6 clouds without explicit tunnels by embedding the tunnel IPv4 address in the IPv6 6to4 prefix.

One challenge with this mechanism is that all 6to4 routers must accept and decapsulate IPv4 packets from every other 6to4 router; there are no strict constraints on what the IPv6 packet may contain, which implies a trust relationship.

Another, bigger challenge is that to interconnect native IPv6 networks and 6to4 clouds, relay routers are used as bridges between these two clouds. Relay routers can be tricked by malicious parties to send IPv4 or IPv6 traffic anywhere the attacker wants. With source address spoofing, this could be called traffic "laundering" or a "proxy" denial-of-service attack. To some extent, these reflected attacks can also be launched off from any node at all.

Even worse, anyone can send tunneled traffic, spoofed to come from non-6to4 addresses to any 6to4 router, and the node does not have any means to ensure its correctness, but to assume it came from a legitimate Relay.

The 6to4 specification outlined quite a few security considerations, but it has been shown that in practice some of these have been

difficult to get implemented due to their abstract nature.

This draft analyses the 6to4 security issues in more detail and outlines some enhancements and caveats.

Sections 2-4 are more or less introductory in nature, rehashing how 6to4 should be used today based on the 6to4 specification, so that it is easier to understand how security could be affected. Section 5 provides a threat analysis for implementations that already implement most of the security checks. Section 6 introduces some filtering rules for 6to4 implementations, and section 7 discusses some additional problems which still need some consideration. Appendix A outlines a few possible trivial attack scenarios in the case that very little or no security has been implemented.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

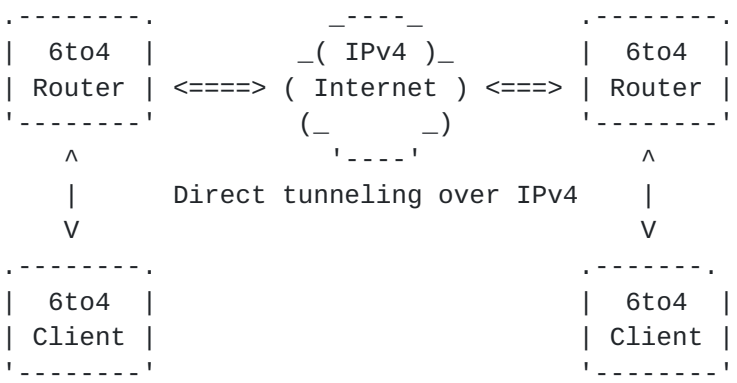
2. Different 6to4 Forwarding Scenarios

It should be noted that when communicating between 6to4 and native domains, the relays that will be used in the two directions are very likely different; routing is highly asymmetric. Because of this, it is not feasible to limit relays you accept traffic from with e.g. access lists.

The first three subsections introduce the most common form of 6to4 operation. Other models are presented in the fourth subsection.

2.1. From 6to4 to 6to4

6to4 domains always exchange 6to4 traffic directly via IPv4 tunneling; the endpoint address V4ADDR is derived from 6to4 prefix 2002:V4ADDR.

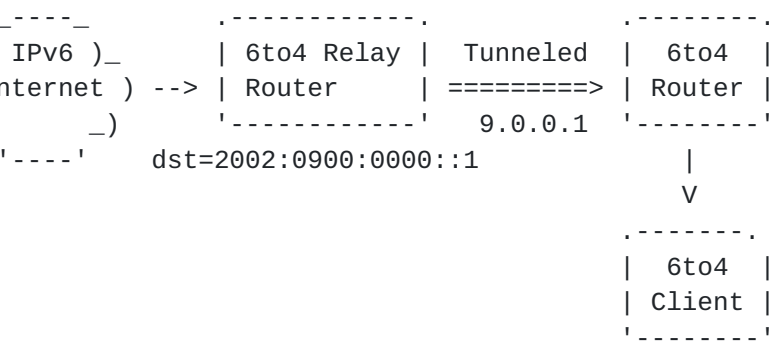


It is required that every 6to4 router considers every other 6to4 router it wants to talk to to be "on-link" (with IPv4 as the link-layer). If this is restricted by increasing the prefix length from 2002::/16, some traffic will be sent to the 6to4 Relay Router, which would forward it to other 6to4 Routers. However, if the original destination does not have equally long prefix, the traffic it tries to send back will be tunneled directly, and will be dropped.

Therefore, the restricted scenario with a longer prefix-length is not globally workable and will not be considered here.

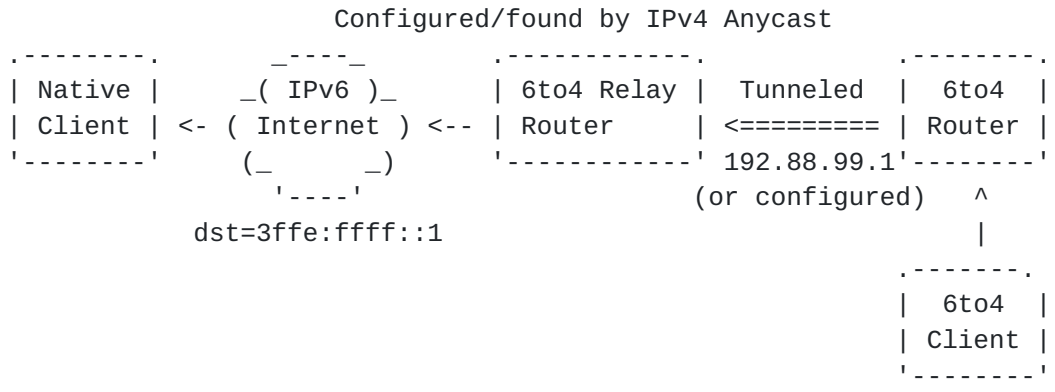
2.2. From Native to 6to4

When native domains send traffic to 6to4 address 2002:V4ADDR, it will be routed to the topologically nearest, advertising 6to4 Relay Router. Relay router will tunnel the traffic over IPv4 to the corresponding IPv4 address V4ADDR. (Note that IPv4 address 9.0.0.1 here is just an example of a global IPv4 address.)



2.3. From 6to4 to Native

6to4 domains send traffic to native domains by tunneling it over IPv4 to their configured 6to4 Relay Router, or the closest found using 6to4 IPv4 Anycast [6TO4ANY]. The relay will decapsulate the packet and forward it to native IPv6 Internet, the same way as any other IPv6 packet.



2.4. Other Models

These are more or less special cases of 6to4 operation; in later chapters, unless otherwise stated, only the most generally-used models (above) will be considered.

2.4.1. BGP Between 6to4 Routers and Relays

[6TO4, 5.2.2.2] presents a model where, instead of static configuration, BGP4+ is used between 6to4 Relay Routers and 6to4 Routers.

If the 6to4 router established a BGP session between all the possible 6to4 relays, the traffic from non-6to4 sites would always go through "home relay", and configuring "trusted relay" would not be a problem; an alternative would be to advertise the more specific 6to4 routes between 6to4 Relays, as described later in this memo.

This model is not known to be used at the time of writing; this is probably caused by the fact that parties that need 6to4 are those that are not able to run BGP; and setting up these sessions would be much more work for relay operators.

2.4.2. 6to4 as an Optimization Method

Some seem to use 6to4 as an IPv6 connectivity "optimization method"; that is, they have also non-6to4 addresses on their nodes and border routers, but also employ 6to4 to reach 6to4 sites.

This is typically done to be able to reach 6to4 destinations by direct tunneling and not having to use relays at all.

Some also publish both 6to4 and non-6to4 addresses in DNS to affect inbound connections; if the source host's default address selection [ADDRSEL] works properly, 6to4 sources will use 6to4 addresses to reach the site and non-6to4 nodes use non-6to4 addresses. If this

behaviour of foreign nodes can be assumed, the security threats to such sites can be significantly simplified.

2.4.3. 6to4 as Tunnel End-Point Addressing Mechanism

6to4 addresses can also be used only as an IPv6-in-IPv4 tunnel endpoint addressing and routing mechanism.

An example of this is interconnecting 10 branch offices where nodes use non-6to4 addresses. Only the offices' border routers need to be aware of 6to4, and use 6to4 addresses solely for addressing the tunnels between different branch offices. This assumes that all outgoing traffic from the main organization (but not between branch offices) uses one or more non-6to4 connections.

This is similar to the optimization model above, and can be used to make the addressing and routing easier.

3. Some Functionalities of 6to4

In this section, some, relatively obvious features of different 6to4 components are listed to better undestand what's the required behaviour.

3.1. Functions of Different 6to4 Network Components

- Non-6to4 (Native) Node

  If native IPv6 nodes want to communicate with 6to4 nodes, they send the traffic along normally. The traffic will reach the topologically closest, advertising 6to4 Relay Router, and will be tunneled to the destination 6to4 Router, which will pass it to the 6to4 node via normal forwarding process.

- 6to4 Host

  A host, usually autoconfigured, that has an address from a 6to4 prefix, but doesn't have a 6to4 pseudo-interface. It doesn't need to know anything about 6to4, and it acts like a normal IPv6 Host in every manner. Note that 6to4 Hosts can also be 6to4 Routers in some scenarios, but then 6to4 Router functionalities, below, apply.

- 6to4 Router

  Acts at the border of a 6to4 domain. It does not have a native, global IPv6 address. More specifically:

- provide "native-like" IPv6 connectivity to local clients and routers
- if packets are sent to foreign 6to4 addresses, tunnel them to the destination 6to4 router using IPv4
- if packets are sent to locally configured 6to4 addresses, forward them normally
- if packets are sent to non-6to4 addresses, tunnel them to the configured/closest-by-anycast 6to4 Relay Router, which will pass them on
- if packets are received from 6to4 addresses, decapsulate the IPv4 packets received from 6to4 routers
- if packets are received from non-6to4 addresses, decapsulate the IPv4 packets received from 6to4 Relay Router closest to the source (note: it is not easily distinguishable that the packet was really received from a Relay router, not from a spoofing third party.)

o 6to4 Relay Router

Acts as a relay between all 6to4 domains and native IPv6; more specifically:

- advertises the reachability of the 2002::/16 prefix to native IPv6 routing, thus receiving traffic to all 6to4 addresses from closest native IPv6 nodes
- (if implements RFC3068) advertise the reachability of IPv4 '6to4 Relay anycast prefix' (192.88.99.0/24) to IPv4 routing, thus receiving some tunneled traffic to native IPv6 nodes from 6to4 Routers
- if packets are received from 6to4 addresses through tunneling, decapsulate them and forwards them on using normal IPv6 routing
- if packets are received through normal IPv6 routing from native addresses, and are destined for 2002::/16, tunnel them to the corresponding 6to4 Router

3.2. Non-functions of Different 6to4 Network Components

What should not happen; this forms a basis for the security checks. The lists are not exhaustive.

o 6to4 Router or Relay
- use private, broadcast or reserved IPv4 addresses in tunnels, or the matching 6to4 prefixes
- receive traffic from 6to4 Routers where the IPv4 tunnel source address does not match the 6to4 prefix

- receive traffic where the destination IPv6 address is not a global address; in particular, e.g. link-local addresses, mapped addresses and such should not be used

o 6to4 Router
- send traffic to other 6to4 domains through 6to4 Relay Router or via some third party 6to4 Router
- receive traffic from other 6to4 domains via a 6to4 Relay Router
- receive traffic to other than to your own 6to4 prefix(es)

o 6to4 Relay Router
- receive traffic from 6to4 to 6to4

4. Special Processing of 6to4 Packets

One could summarize the special processing of 6to4 as follows:

o Relay Router
1. incoming from native, tunneled to 6to4
2. tunneled from 6to4, going to native

o Router
1. tunneled from relay, source is native
2. tunneled to relay, destination is native
3. tunneled directly, destination is 6to4

4.1. Encapsulating IPv6 Packets into IPv4

IPv6 packets are encapsulated into IPv4 in three scenarios:

1. 6to4 Router sends packets to other 6to4 Routers (2002::/16 destination)
2. 6to4 Router sends packets to its configured/nearest-by-anycast 6to4 Relay Router (non-2002::/16 destination)
3. 6to4 Relay Router sends packets from native IPv6 sources to 6to4 Routers (2002::/16 destination)

4.2. Decapsulating IPv4 Packets into IPv6

IPv6 packets are decapsulated from IPv4 in three scenarios:

1. 6to4 Router receives packets from other 6to4 Routers (2002::/16 source)
2. 6to4 Router receives packets from a node, supposedly 6to4 Relay Router closest to the source (non-2002::/16 source)
3. 6to4 Relay Router receives packets from 6to4 Routers, to be sent to native IPv6 destinations (2002::/16 source)

5. Threat Analysis

The 6to4 threat analysis presented here focuses on 6to4 implementations which have implemented most if not all security checks listed in [6TO4]; in particular, checks that always match 2002:V4ADDR and V4ADDR must be implemented. Many implementations are known to be problematic at least in some cases.

The appendix lists some additional trivial threats which are applicable if no or only little security is implemented.

The IPv4 address blocks 8.0.0.0/24 and 9.0.0.0/24 are only used for demonstrative purposes, representing global IPv4 addresses.

5.1. Threats Related to Any 6to4 Node

Any 6to4 node can be made to participate in a DoS attack listed in 5.2.2 below, being used as "dst". The threat will be discussed there.

5.2. Threats Related to 6to4 Routers

Note that in this memo, a loose interpretation of "6to4 router" is used; it is also used to refer to those 6to4 nodes which have the 6to4 pseudo-interface.

There are two main classes of threats; attacks against the 6to4 pseudo-interface and attacks relying on being able to abuse the fact that it is difficult for 6to4 routers to tell whether packets from non-6to4 nodes come from legitimate relays.

5.2.1. Attacks Against the 6to4 Pseudo-Interface

Unless the 6to4 pseudo-interface has been sufficiently protected, it's possible to remotely attack against the pseudo-interface with tunneled link-local packets, just as if it were in a local network. Threats to Neighbor Discovery are listed in [SEND].

The potential effects of an attack can be mitigated if the interface is insulated from the other interfaces (e.g. a separate neighbor cache). In practise, this is not the case.

The attack can be eliminated by restricting the use of 6to4 pseudo-interface: if any packet with scope smaller than global is received, it must be silently discarded. (Site-local might be allowed in some restricted scenarios.) This may conflict with future uses of [6TO4, 3.1].

5.2.1.1. Comparison to Situation without 6to4

Even though rather simply fixable, this attack is not new as such; the same is possible using automatic tunneling [MECH] or configured tunneling (if one is able to spoof source IPv4 address to that of the tunnel end-point).

However, as automatic tunneling is being deprecated, the worst problem comes from 6to4; any open decapsulation is bad.

5.2.2. Relay Spoofing, DoS against IPv6 Nodes

6to4 Routers receive packets from non-6to4 source addresses through Relay Routers, as described in section 2.2 and 4.2 point 2.

In the general case, the 6to4 router cannot distinguish Relay routers from malicious nodes sending IPv4-encapsulated IPv6 traffic directly to the 6to4 router.

This makes two kinds of attacks possible:

- unidirectional source address spoofing
- Denial-of-Service attack reflection against native IPv6 nodes

In both scenarios, the attacker sends IPv4-encapsulated IPv6 packets to the 6to4 router with contents like:

```
src = 3ffe:1122:3344::1 (forged)
dst = 2002:0900:0002::1
src_v4 = 8.0.0.1
dst_v4 = 9.0.0.2 (matching dst)
```

Now the 6to4 router receives these packets from 8.0.0.1, decapsulates them, discarding the IPv4 header containing the source address 8.0.0.1 and processes them as normal ("dst" has been guessed or obtained using one of a number of techniques).

In the first scenario, it is assumed that "src" is somehow specially trusted (at least to some extent) more than some other packets. This kind of weak trust based on IP addresses is commonplace. This enables unidirectional (as the replies will be lost) source address spoofing, but may be enough for e.g. exploiting some remote vulnerabilities in connectionless protocol server applications.

In the second scenario, if "dst" exists, the replies (e.g. TCP SYN ACK, TCP RST, ICMP Echo Reply, input sent to UDP echo service, etc.) are sent to the victim "src", above. All the traces from the original attacker src_v4 have been discarded. These return packets

will go through a relay.

These attacks are similar to ones shown in [RHHASEC].

5.2.2.1. Comparison to Situation without 6to4

The unidirectional spoofing attack exists without 6to4 too, but it requires the attacker is able to spoof IPv6 source addresses. With 6to4, one is able to launch this attack without any spoofing at all. A restriction is that the source address cannot be spoofed to belong to the destination site as only non-6to4 addresses can be spoofed (assuming correct implementations). Blindly trusting source address of packets coming from the Internet without other precautions is very bad practise, though -- and this attack would typically be useful only for spoofing destination site -- which is not possible, and can be protected against with egress filtering.

The Denial-of-Service attack is also not really new; the only twists come from the fact that traces of an attack are more easily lost and that attacker does not really have to even spoof his IPv4 address to be able to reasonably discreetly launch the attack.

However, it can be argued that this DoS attack is not critical because:

- 6to4 as an enabling mechanism does not provide any possibility for packet multiplication, attacks are generally 1:1
- victim receives packets as replies from "dst": unless echo service (e.g. UDP port 7) is used, the attacker has reasonably little control on the content of packets; for example, pure "SYN" TCP packets often used for flooding cannot be sent.
- attack packets pass through choke point(s), namely (one or more) 6to4 relays; in addition to physical limitations, these could implement some form 6to4-site-specific traffic limiting
- one has to know a valid destination address (however, this is easily guessable or deducible with e.g. an ICMP echo request with a limited Hop Count).

The attack can be launched from hosts whose connection is ingress-filtered, too. So, this enables a way to launch attacks which hide the source address even when it was supposed to be prevented (for IPv4).

However, often this is not a real factor, as usually the attackers are just zombies and real attackers may not even care if the unspoofed source address is discovered.

This is one of the most serious threats.

5.3. Threats Related to 6to4 Relays

6to4 Relays are also subject to attacks, but usually in a different role than 6to4 Routers; usually Relays' function is anonymization of the attack and losing trails, not reflection -- as properly implemented relays should be resistant to reflection.

6to4 Relays have only one significant security check they must perform for general safety: when decapsulating IPv4 packets, check that 2002:V4ADDR and V4ADDR match. If this is not done, several threats become more serious.

Also, it is assumed here that the Relay checks that it will not relay packets between 6to4 addresses. In particular, packets decapsulated from 6to4 routers cannot be encapsulated again towards 6to4 routers, as descibed in rules in section 6.

5.3.1. Attacks Against the 6to4 Pseudo-Interface

The threats are the same as against 6to4 routers.

5.3.2. Spoofing, DoS against IPv6 Nodes

If one cannot assume that IPv6 source addresses are ingress-filtered, the same threats as listed in 5.2.2 apply.

The difference here is that a native IPv6 node spoofs the source IPv6 addresses, and the relay router provides a layer of indirection and loses the trails.

5.3.3. Participating in DoS attacks against IPv4

An attack specific to 6to4 Relays is similar to 6to4 Routers, but against IPv4; the attacker sends IPv6-native packets with an IPv4 address he wants to bomb as the 6to4 destination address, like:

```
src = 3ffe:1122:3344::1 (spoofed or real attacker)
dst = 2002:0900:0002::1 (victim)
```

Now the 6to4 relay receives these packets, and encapsulates in IPv4 packets that are sent towards 9.0.0.2; the destination may not have a faintest idea of what IPv6 is, but is bombed with packets coming from the relay's IPv4 address, including IPv6 packets as the payload.

5.3.3.1. Comparison to Situation without 6to4

Slightly different arguments apply; below are reasons why this can be considered not too serious an attack:

- o 6to4 as an enabling mechanism does not provide any possibility for packet multiplication, attacks are generally 1:1
- o victim receives packets as sent by the source; if the victim is an IPv4-only node, it just sees "protocol 41" packets which are not typically dangerous and easily filtered
- o 6to4 relay's source IPv4 address is used in packets, and tracing is easier
- o source IPv6 address (spoofed or real) is in the packets which may make manual tracing easier
- o attack packets pass through choke point(s), namely (one or more) 6to4 relays; in addition to physical limitations, these could implement some form 6to4-site-specific traffic limiting

And as counter-arguments, some more serious aspects of it are:

- o victim receives packets as sent by the source; if the victim is 6to4 node, IPv6 packet can include almost anything; however if IPv6 source addess is not spoofed, this attack is nothing new.
- o the relays can be chosen by the attacker, so if there are a large number of relays, he can pick ones that are known best suited for the attack (e.g. bad security policy, ones using 192.88.99.1 as source for more difficult tracing, etc.)

5.3.4. Using Any IPv6 Node for Reflection

Any IPv6 node will respond to IPv6 packets destined to the node with source address belonging to 2002::/16.

This attack is applicable if:

- o the relay chosen by the attacker does not check that IPv4 source and 2002::/16 source address match, or
- o the attacker's IPv6 connection is not ingress-filtered (and it was really a non-6to4 source)

The IPv6 source address being forged, the node will participate as an unwilling intermediary to an attack through a 6to4 relay against any IPv4 node (or 6to4 node), like:

```
src = 2002:0900:0002::1 (forged, target of the attack)
dst = 3ffe:1122:3344::1 (intermediary node)
```

This is not new: similar attack with any other spoofed source address is possible if ingress filtering is not enabled. The only difference here is that when attacking IPv4 nodes, the relay's IPv4 source address is seen as the immediate source of the attack. Closer inspection (after packet reflection) reveals the IPv6 datagram with (possibly spoofed) 2002::/16 destination address.

5.3.4.1. Comparison to Situation without 6to4

This attack is a reflected variation of the attack above; the implications are similar to those in section 5.2.2.1:

- o A non-compliant 6to4 implementation or IPv6 source address spoofing is required
- o 6to4 as an enabling mechanism does not provide any possibility for packet multiplication, attacks are generally 1:1
- o victim receives packets as replies from "dst": unless echo service (e.g. UDP port 7) is used, the attacker has reasonably little control on the content of packets; for example, pure "SYN" TCP packets often used for flooding cannot be sent.
- o attack packets pass through choke point(s), namely (one or more) 6to4 relays; in addition to physical limitations, these could implement some form 6to4-site-specific traffic limiting

5.3.5. IPv4 Local Directed Broadcast Attacks

This threat is applicable if the relay does not check whether the IPv4 address it tries to send encapsulated IPv6 packets to is a local broadcast address. This threat is mentioned in [6TO4]. The packet could be sent as follows:

```
src = 3ffe:ffff:5678::aaaa (may be forged)
dst = 2002:0900:00ff::bbbb
```

9.0.0.255 is assumed the the relay's broadcast address. After receiving the packet natively, if the relay doesn't check the destination address for subnet broadcast, it would send the encapsulated IP-IP packet to 9.0.0.255. This would be received by all nodes in the subnet, and the responses would be directed to the relay.

5.3.5.1. Comparison to Situation without 6to4

This is a special form of "directed broadcast" attack which cannot be protected against except by the mentioned check. However, there is a

significant difference: the reply packets are sent back to the relay. Therefore only the non-compliant device can suffer from this; the rest of the Internet cannot be affected.

5.3.6. Theft of Service

The administrators of 6to4 Relay Routers often want to use some policy to limit the use of relay.

The policy control is usually done by applying some restrictions in where the routing information for 2002::/16 and/or 192.88.99.0/24 (if [6TO4ANY] is used) will spread.

Some users may be able to use the service regardless of these controls using:

- configuring the address of the relay using its IPv4 address instead of 192.88.99.1
- using Routing Header to route IPv6 packets to reach some 6to4 Relay (some other routing tricks like neighbors setting static routes are also possible)

The former can be protected against using configured access lists in the relay; this is only feasible if the number of IP networks the relay is supposed to serve is relatively low. Another possible way to mitigate this is to filter out arriving tunneled packets with protocol 41 (IPv6) which do not have the the 192.88.99.1 as the destination address.

The latter has similar issues: the IPv6 source address could be checked so the packets to the relay only come from the valid IPv6 addresses which are able to reach the relay anyway.

Of these, except in really restricted scenarios, only the first may be of some interest, and the mitigating the problem by access list is rather straightforward.

As this threat does not have implications on other than the organization providing Relay, it is not further analysed.

5.4. Possible Threat Mitigation Methods

This section gives a rough idea of mechanisms thought to mitigate the threats.

5.4.1. 6to4 Decapsulation Cache

6to4 decapsulators (routers, relays) could keep a least recently used (LRU) header cache of possibly a few hundred entries of recently seen packets for tracing purposes.

The problem here is how that kind of data could be extracted -- by third parties that need it -- in timely fashion. Many implementations are, of course, already able to perform something like this by e.g. manually set logging access lists.

5.4.2. Rate-limiting at 6to4 Routers/Relays

TBD.

5.4.3. An Application of iTrace Model

6to4 decapsulators (or some of them) could send out some specific packets probabilitically as a way ensure that reflectors cannot be used to lose trails of an attack. This could either be a simplification or an extension of e.g. [ITRACE] model, depending on how fast its specification goes.

The most important place for this would be at 6to4 Routers, to counter the reflection attack descibed in 5.2.2. If so, the check could be placed at the decapsulation phase where packets have a 6to4 destination address but the source is non-6to4.

5.5. Summary

It would be useful to try to characterize the different threats by comparing the severity of the threat to:

1. IPv4 networks today, where in many cases (even most), source address spoofing is possible and there are no easy ways to trace attacks

2. Hypothetical IPv4 networks -- the case if ingress filtering would be deployed everywhere

3. Hypothetical IPv6 networks -- the case if ingress filtering would be deployed everywhere in current and future IPv6 networks

However, this would be very difficult as it is not easy to assign severity values to all the features 6to4 adds and try to decide whether it's more serious or not.

5.5.1. Summary of the Threats

Below is the summary of the threats discussed above. Threat in 5.1 was merged with 5.2.2 as the effects are the same but from a different perspective.

Type	Sec	Characterization	Using	Against	Fix	I-F	Comp
Othr	5.2.1	Pseudo-Interface	Rtr/Rly	itself	yes	N/A	3
Othr	5.3.5	Local Direct. Bcast	Rly	itself	yes	N/A	3
Othr	5.3.6	Theft of Service	Rly	itself	yes	N/A	-
Spf	5.2.2	Relay Spoofing	Rtr	ownsite	y?	-	same
Dir	5.3.3	DoS against IPv4	Rly	any v4	?	6	1,2
Refl	5.2.2	Refl. off any 6to4	Rtr/Any	non6to4	?	-	2
Refl	5.3.2	Refl. off any 6to4	R*/Any	non6to4	?	6	2
Refl	5.3.4	Refl. off any IPv6	Rly/Any	any v4	1/2	4+6	1,2

The table is sorted by threat type. Possibilities are spoofing, direct attack, attack by reflection (ie. final attack consists of some response packets) and other.

Threats when realize (ab)use some IPv6 nodes: possibilities are either 6to4 Routers (Rtr), 6to4 Relays (Rly) or any IPv6 nodes or any 6to4 nodes (Any). "R*" means that both Relays and Routers are used.

The final target of the attack is descibed in "Against"; it can be node(s) or network itself, the site itself which could prevent the attack, any IPv4 node or any non-6to4 IPv6 node (non6to4).

If a fix for the problem is apparent, it is mentioned in the Fix field.

If it can be assumed that either complete Internet-wide IPv4 or IPv6 ingress filtering would (more or less) fix or significantly alleviate the problem, the fixing version of ingress filtering is noted in I-F column. The notable case is 5.3.4 where both v4/v6 ingress filtering is needed -- but if the half of the readily-available fix is done, IPv6 ingress filtering is enough. The other notable case is threat 5.2.2, which cannot be disabled by ingress filtering.

The last field "Comp" tries to compare the threats to their IPv4 equivalents, using:

1. cannot control packets significantly, ie. a weak attack
2. can be mitigated significantly by adding some kind of tracing
3. some new form of attack

5.5.2. Generic Notes about Threats

Note: TBD.

- correct and fully-implemented base security features are a pre-requisite for reasonably safe operation
- being able to spoof IPv4 or IPv6 packets enables one to launch similar or more powerful attacks even currently
- some 6to4 attacks provide an additional layer of indirection, which may or may not be useful
- 6to4 as an enabling mechanism does not provide any possibility for packet multiplication which would affect global Internet, attacks are generally 1:1
- typically the reflected packets have restricted content, limiting the usability in an attack
- attacks typically have either 6to4 relay router's address or some other information which could be used in manual tracing
- attack packets pass through choke point(s), namely (one or more) 6to4 relays; in addition to physical limitations, these could implement some form 6to4-site-specific traffic limiting
- attacks could in theory be traceable using an extension of [ITRACE] or [REVITRACE], but as those haven't been specified, much less used, the point seems rather academic yet

When considering motives for DoS attacks and how to protect against them (and considering the cost, and whether the protection actually buys you anything), the following should not be forgotten:

- IPv4 and IPv6 ingress filtering are not likely to be commonplace for a long time; until it is, you cannot really depend on it
- the real attacker (launching a DoS or DDoS) may not really even care whether some zombie nodes get found out
- techniques to trace DoS attacks are still in infancy (or not even there) yet; due to time anything takes to get deployed, it is not clear whether tracing mechanisms even for basic DoS attack mechanisms would get reasonably widely deployed before it was time to (more or less) retire 6to4
- DoS attacks are something that, in practise, operational people have to be able to deal with anyway

6. Solutions

6.1. Generic Approach

6.1.1. Encapsulating IPv6 into IPv4

```
    src and dst MUST pass ipv6-sanity checks, else drop (defined below)
    if src=2002
        src MUST match src_v4
            [ the scenario: 4.1. case 1. or 2. ]
        if dst=2002
            dst_v4 SHOULD NOT be assigned to the router (avoid
misconfigurations)
                [ the scenario: 4.1. case 1. ]
        fi
    elif dst=2002
        dst_v4 MAY have to match one of ipv4 equiv. of 6to4 prefixes masked by
a
           user-specified prefix length (restricting who can use the relay)
            [ the scenario: 4.1. case 3. ]
    else
        drop
            [ the scenario: we somehow got a native-native ipv6 packet ]
    fi
    accept
```

6.1.2. Decapsulating IPv4 into IPv6

```
    src_v4 and dst_v4 MUST pass ipv4-sanity checks, else drop (defined below)
    src and dst MUST pass ipv6-sanity checks, else drop (defined below)
    if dst=2002
        dst MUST match dst_v4
            [ the scenario: 4.2. case 1. or 2. ]
            if src=2002
                src MUST match src_v4
                dst_v4 SHOULD be assigned to the router (see notes below)
                    [ the scenario: 4.2. case 1. ]
            fi
    elif src=2002
        src MUST match src_v4
        dst_v4 SHOULD be assigned to the router (see notes below)
        src_v4 MAY have to match one of ipv4 equiv. of 6to4 prefixes masked by
a
              user-specified prefix length (restricting who can use the relay)
            [ the scenario: 4.2. case 3. ]
    else
        drop
            [ the scenario: we somehow got a native-native ipv6 packet ]
```

```
    fi
    accept
```

6.1.3. IPv4 and IPv6 Sanity Checks

6.1.3.1. IPv4

IPv4 address MUST be a global unicast address, as required by the 6to4 specification. The disallowed addresses include those defined in [RFC1812], and others widely used and known not to be global. These are:

- 0.0.0.0/8 (the system has no address assigned yet)
- 10.0.0.0/8 (private)
- 127.0.0.0/8 (loopback)
- 172.16.0.0/12 (private)
- 192.168.0.0/16 (private)
- 169.254.0.0/16 (IANA Assigned DHCP link-local)
- 224.0.0.0/4 (multicast)
- 255.0.0.0/8 (broadcast)

In addition it MUST be checked that the address is not any of the system's broadcast addresses. This is especially important if the implementation is made so that it can receive and process encapsulated IPv4 packets arriving at its broadcast addresses, or try to send encapsulated IPv4 packets to one of its broadcast addresses.

6.1.3.2. IPv6

IPv6 address MUST NOT be:

- 0::/16 (compatible, mapped addresses, loopback, unspecified, ...)
- fe80::/10 (link-local)
- fec0::/10 (site-local)
- ff02::/16 (link-local multicast)

Other multicast could also be considered for filtering.

In addition, it MUST be checked that equivalent 2002:V4ADDR checks, where V4ADDR is any of the above IPv4 addresses, will not be passed.

6.1.3.3. Optional Ingress Filtering

In addition, the implementation may perform some form of ingress filtering (e.g. Unicast Reverse Path Forwarding checks). For example, if the 6to4 Router has multiple interfaces, of which some are "internal", receiving either IPv4 or IPv6 packets with source address belonging to any of these internal networks from the Internet might be disallowed.

If these checks are implemented, it is RECOMMENDED that they default to disabled.

6.1.3.4. Notes About the Checks

The rule 'dst_v4 SHOULD be assigned to the router' is not needed if the implementation is made in such a way that it only accepts and processes encapsulated IPv4 packets arriving on unicast IPv4 addresses, and that if destination address is known to be a local broadcast address, not try to encapsulate and send packets to it.

Some checks, especially the IPv4/IPv6 Sanity Checks, could be at least partially implementable with system-level access lists, if one would like to avoid placing too many restrictions in the 6to4 implementation itself. This depends on how many hooks for ACL's are in place. In practice it seems like this could not be done effectively enough unless the access list mechanism is able to parse the encapsulated packets within IP-IP.

6.2. Simplified Approach

This makes some assumptions about the implementation as pointed above to simplify the above rules.

6.2.1. Encapsulating IPv6 into IPv4

```
src and dst MUST pass ipv6-sanity checks, else drop
if src=2002
    src MUST match src_v4
elif dst=2002
    (accept)
else
    drop
fi
accept
```

6.2.2. Decapsulating IPv4 into IPv6

```
src_v4 and dst_v4 MUST pass ipv4-sanity checks, else drop
src and dst MUST pass ipv6-sanity checks, else drop
if dst=2002
    dst MUST match dst_v4
        if src=2002
            src MUST match src_v4
        fi
elif src=2002
    src MUST match src_v4
else
```

```
        drop
    fi
    accept
```

7. Problems

7.1. Implementation Considerations with Automatic Tunnels

There is a problem with multiple transition mechanisms if strict security checks are implemented. This may vary a bit from implementation to implementation.

Consider three mechanisms using automatic tunneling: 6to4, ISATAP [ISATAP] and Automatic Tunneling using Compatible Addresses [MECH]. All of these use IP-IP (protocol 41) [IPIP] IPv4 encapsulation with, more or less, a pseudo-interface.

When a router, which has any two of these enabled, receives an IPv4 encapsulated IPv6 packet:

```
src_v4 = 10.0.0.1
dst_v4 = 20.20.20.20
src = 3ffe:ffff::1
dst = 2002:1010:1010::2
```

what can it do? How should it decide which transitionary mechanism this belongs to; there is no "transitionary mechanism number" in IPv6 or IPv4 header to signify this. (This can also be viewed as a flexibility benefit.)

Without any kind of security checks (in any of implemented methods) these often just "work" as the mechanisms aren't differentiated but handled in "one big lump".

Configured tunneling [MECH] does not suffer from this as it is point-to-point, and based on src/dst pairs of both IPv4 and IPv6 addresses it can be deduced which logical tunnel interface is in the question.

Solutions for this include not using more than one automatic tunneling mechanisms in the same system or binding different mechanisms to different IPv4 addresses.

7.2. Reduced Flexibility

There is a worry about too strict rules limiting the (future) flexibility of 6to4. If later, for some reason, one would want to introduce new revolutionary ways to use 6to4, strict checking in all relevant nodes might prevent it, as new updated version would have to be deployed everywhere before the new method could be used.

On the other hand, one could argue that 6to4 has always been intended as an intermediate mechanism, and that future flexibility should not be critical. However, it is difficult to predict how long the intermediate period will be.

7.3. Anyone Pretending to Be a Relay Router

6to4 Routers receive traffic from non-6to4 ("native") sources via 6to4 Relays. 6to4 Routers have no way of matching IPv4 source address of the relay with non-6to4 IPv6 address of the source. In consequence, anyone can spoof any non-6to4 IPv6 address he wants by sending traffic, encapsulated, directly to 6to4 Routers. This is analyzed in more detail in the Threat Analysis section, above.

Of course, as the source IPv4 address may be logged, many may spoof their IPv4 source address, but the ability to do so is not be required: it is unlikely that source IPv4 (but rather, the spoofed IPv6 address) will be logged anywhere -- this would be equivalent to logging the MAC-address of IP packets.

Unfortunately, this problem is very difficult to solve properly. There have been two rough ideas:

- Every 6to4 Relay MUST configure and use "192.88.99.1" as the source address of packets that are encapsulated towards 6to4 Routers.

- Every 6to4 Relay MUST participate in an eBGP multi-hop peering mesh (which can be hierarchical): there more specific routes will be advertised.

It should be noted that if IPv6 operators do not implement ingress filtering for IPv6, so that spoofing IPv6 is not more difficult than spoofing IPv4, these problems have only little impact on the overall security of 6to4 nodes.

The first has since then been rejected: the difference in the difficulty of spoofing an address and spoofing it to be 192.88.99.1 does not seem to justify the mechanism. A tentative analysis for the second is given below.

7.3.1. Limited Distribution of More Specific Routes

If 6to4 prefixes more specific than 2002::/16 could be advertised, the traffic model between native<->6to4 and 6to4<-> could be changed so that only one Relay would always be used, most often the one closest to the 6to4 Router.

This model was rejected in the base specification, as IPv6 routing table was not to be polluted by 6to4 prefixes derived of IPv4 prefixes.

However, the problem can be avoided with some effort: creating a separate, possibly hierarchical based on IPv6 connections, peering mesh for 6to4 Relay routers. This could be done by forming eBGP [BGP] multi-hop peerings between Relays, and advertising more specific routes (e.g. the same superblocks of IPv4 addresses one expects to service) to all the other Routers.

In that way, the global routing table would not be impacted at all.

This seems to have at least three potential issues:

- Every Relay should be part of this (if one wants to have some extra safety that the addresses haven't been spoofed)
- The route from a native source takes the path to the first Relay, and there (possibly through other Relays) to the last Relay to tunnel the packet to the 6to4 Router; this adds at least some latency
- The mechanism of redistributing IPv4 6to4 client addresses to other relays as 6to4 prefixes needs work

Of these, only the last requires more discussion. It could be argued that this advertising should either be manually configured once (ie. relatively static prefixes, or generated from IPv4 route-objects in RADB etc.) or generated automatically (e.g. when a 6to4 Router first sends a "triggering" packet through the Relay). Of course, this data could even be derived in some form from the IPv4 routing tables. Further analysis on this is TBD if necessary.

This method seems to be the only one where strong cryptography-based mechanisms to be sure about the 6to4 Router - 6to4 Relay -relationship could be doable; otherwise, some sort of infrastructure (e.g. small-scale PKI) would have to be established which would have to include all the possible 6to4 Relays in the Internet.

8. Security Considerations

This draft discusses security considerations.

Even if proper checks are implemented, there are significant security threats ranging from DoS proxy attacks to spoofing and attacks against 6to4 pseudo-interface. These threats are analyzed in section 5.

As can be seen, there are mainly three classes of potential problem sources:

- 6to4 routers not being able to identify whether relays are legitimate
- wrong or impartially implemented 6to4 Routers
- relays performing packet laundering

The first is the toughest problem, still under research. The second can be fixed by ensuring the correctness of implementations; this is important. Thr third is also a difficult, but a fairly restricted problem as relays are limited in number.

These are analyzed in detail in Threat Analysis section, above.

9. Acknowledgements

Some issues were first brought up in [TRANSAB] and Alain Durand introduced one specific problem at IETF51 in August 2001 (though there was some discussion on the list prior to that); these gave the author the push to start looking into the details of securing 6to4.

Alexey Kuznetsov brought up the implementation problem with IPv6 martian checks. Christian Huitema formulated the rules that rely on Relays using only anycast. Keith Moore brought up the point about reduced flexibility. Brian Carpenter, Tony Hain and Vladislav Yasevich are acknowledged for lengthy discussions. Alain Durand reminded of relay spoofing problems. Brian Carpenter reminded of the BGP-based 6to4 router model. Christian Huitema gave a push to a more complete threat analysis.

In the latter phase, especially discussions with Christian Huitema, Brian Carpenter and Alain Durand were helpful when improving the document.

10. References

10.1. Normative References

[6TO4] Carpenter, B. and Moore K., "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

[6TO4ANY] Huitema, C., "An Anycast Prefix for 6to4 Relay Routers", RFC 3068, June 2001.

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., de Groot, G. and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[ADDRSEL] Draves, R., "Default Address Selection for IPv6", draft-ietf-ipv6-default-addr-select-09.txt (work-in-progress), August 2002.

[BGP] Rekhter, Y., Li, T., "A Border Gateway Protocol 4", RFC1771, March 1995.

[IPIP] Simpson, W., "IP in IP Tunneling", RFC 1853, October 1995.

[ISATAP] Templin, F. et al, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", draft-ietf-ngtrans-isatap-06.txt (work-in-progress), October 2002.

[ITRACE] Bellovin, S., Leech, M., Taylor, T., "ICMP Traceback Messages", draft-ietf-itrace-01.txt (work in progress).

[MECH] Gilligan, R., and Nordmark, E. "Transition Mechanisms for IPv6 Hosts and Routers", RFC 2893, August 2000.

[REVITRACE] Barros, C., "A Proposal for ICMP Traceback Messages", http://www.research.att.com/lists/ietf-itrace/2000/09/msg00044.html.

[RHHASEC] Savola, P., "Security of IPv6 Routing Header and Home Address Options", draft-savola-ipv6-rh-ha-security-03.txt (work-in-progress), December 2002.

[SEND] Nikander, P. (Ed.), "IPv6 Neighbor Discovery trust models and threats", draft-ietf-send-psreq-00.txt (work-in-progress), October 2002.

[TRANSAB] Hagino, J., "Possible abuse against IPv6 transition technologies", draft-itojun-ipv6-transition-abuse-01.txt (expired, work-in-progress), July 2000.

Author's Address

Pekka Savola
CSC/FUNET
Espoo, Finland
EMail: psavola@funet.fi

A. Some Trivial Attack Scenarios Outlined

Here, a few trivial attack scenarios are outlined -- ones that are prevented by implementing checks listed in [6TO4] or in this memo.

When two 6to4 Routers send traffic to each others' domains, packet sent by RA to RB is like (note: addresses from 8.0.0.0/24 are just examples of global IPv4 addresses):

```
src = 2002:0800:0001::aaaa
dst = 2002:0800:0002::bbbb
src_v4 = 8.0.0.1 (added when encapsulated to IPv4)
dst_v4 = 8.0.0.2 (added when encapsulated to IPv4)
```

When the packet is received by IPv4 stack on RB, it will be decapsulated so that only src and dst remain, as originally sent by RA:

```
src = 2002:0800:0001::aaaa
dst = 2002:0800:0002::bbbb
```

As every other node is just one hop away (IPv6-wise) and the link-layer (IPv4) addresses are lost, this may open a lot of possibilities for misuse.

As an example, unidirectional IPv6 spoofing is made trivial because nobody can check (without delving into IP-IP packets) whether the encapsulated IPv6 addresses were authentic (With native IPv6, this can be done by e.g. RPF-like mechanisms or access lists in upstream routers).

```
src = 2002:1234:5678::aaaa (forged)
dst = 2002:0800:0002::bbbb
src_v4 = 8.0.0.1 (added when encapsulated to IPv4)
dst_v4 = 8.0.0.2 (added when encapsulated to IPv4)
```

A similar attack with "src" being native address is possible even with the security checks, by having the sender node pretend to be a 6to4 Relay router.

More worries come in to the picture if e.g.

```
src = ::ffff:[some trusted IPv4 in a private network]
src/dst = ::ffff:127.0.0.1
src/dst = ::1
src/dst = ...
```

Some implementations might have been careful enough to have designed the stack to as to avoid the incoming (or reply) packets going to IPv4 packet processing through special addresses (e.g. IPv4-mapped addresses), but who can say for all ...