Security Vulnerabilities of IPv6 Tunnels

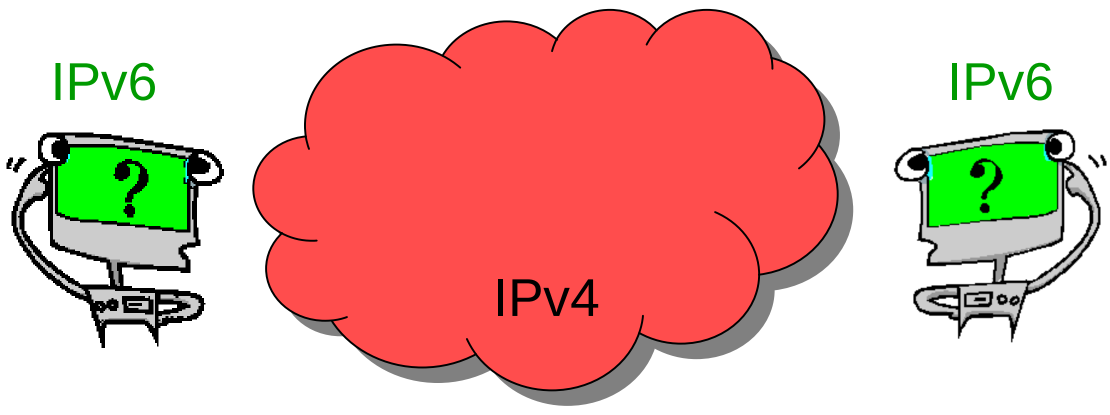
This article talks about novel security vulnerabilities of IPv6 tunnels – an important type of migration mechanisms from IPv4 to IPv6 implemented by all major operating systems and routers. The vulnerabilities allow an attacker to form routing loops which can easily produce DoS attacks. I will describe the principles of the attack and then show how to actually implement it. The vulnerabilities were first presented at Usenix W00T '09 in a paper I co-authored with Michael Arov.

IPv6 background

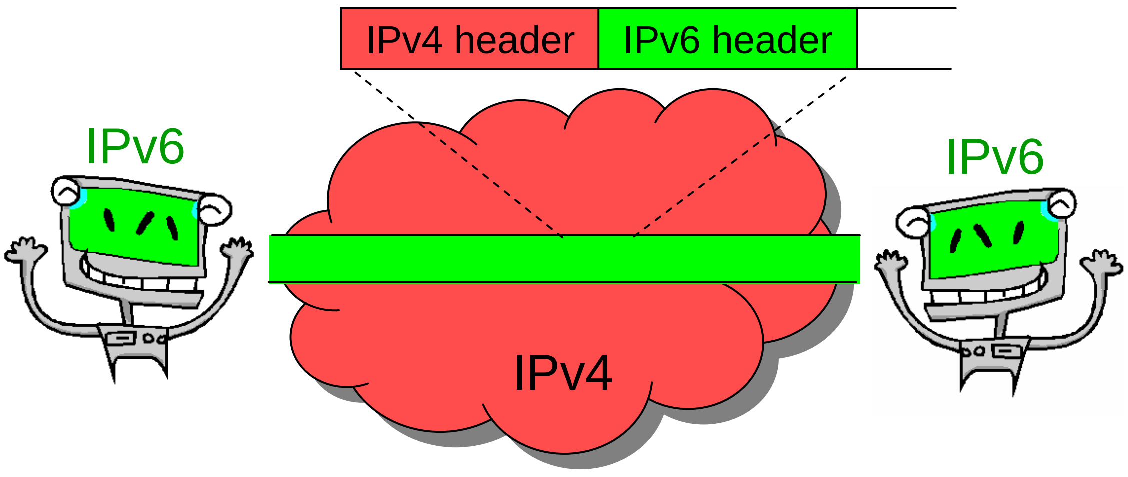
IPv6 is the next generation version of the omnipresent Internet Protocol. It was designed to answer the future exhaustion of the IPv4 address pool. IPv4 address space is 32 bits which translates to just above 4 billion addresses. IPv6 address space is 128 bits. This roughly translates to the entire address space of the IPv4 for every square nanometer on planet earth (a square nanometer is about the size of a molecule!). The IPv6 specification has been released almost 13 years ago, however it has seen little adoption so far since the IPv4 address space hasn't depleted yet. But this is about to change very soon. ISPs in Asia-Pacific are already unable to request new IPv4 addresses for their customers. In other regions of the world IPv4 address pools are expected to drain within a year or so. So IPv6 is definitely here to stay.

Migration to IPv6 using tunnels

As one might imagine the migration to IPv6 will be very gradual. Some estimate that IPv4 and IPv6 will co-exist on the Internet for decades. This issue presents a problem since the two versions are not compatible: a host or a router supporting IPv4 cannot process an IPv6 packet. So how IPv6 hosts can talk over a network that hasn't been migrated yet?



To solve this problem a few interoperability mechanisms has been devised by the IETF. They are called *Tunnels*. In general, tunnels operate as follows. Each tunnel has at least two end points. The ingress end point of the tunnel encapsulates the packet with an IPv4 header. The source IPv4 address is that of the ingress end point and the destination IPv4 address is that of the intended egress end point. 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.

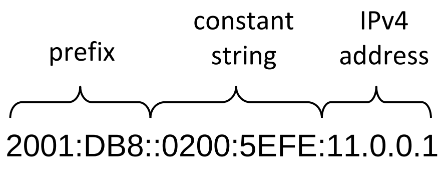


However, there is a problem with tunnels. Each end point must know it's peer's IPv4 address before sending a packet over the tunnel. This is usually accomplished by pre-configuring every end-point with the IPv4 addresses of all the other end points in the tunnel – an administrative nightmare. To alleviate this scalability problem, another type of tunnels was introduced -- *automatic tunnels*. In automatic tunnels an end point's IPv4 address is computationally derived from the destination IPv6 address. This feature eliminates the need to keep an explicit configuration at the tunnel's end points of the IPv4 addresses. In fact, the end points of an automatic tunnel do not know which other end points are currently part of the tunnel. However, all end points operate on the implicit assumption that once a packet arrives at the tunnel, its destination indeed is part of the tunnel. We next show that an attacker can exploit this lack of knowledge.

One example of an automatic tunnel is ISATAP. ISATAP is implemented in all recent version of Microsoft, Linux, and in some versions of Cisco IOS. Every end point of an ISATAP tunnel has an IPv6 address of the following format:

<*tunnel prefix*>:0200:5EFE:<*IPv4 address*>

First, there is a tunnel prefix of 64 bits. This is common to all tunnel end points. Then, there is a constant 32 bit string. And finally, the IPv4 address of the end point. Here is an example:

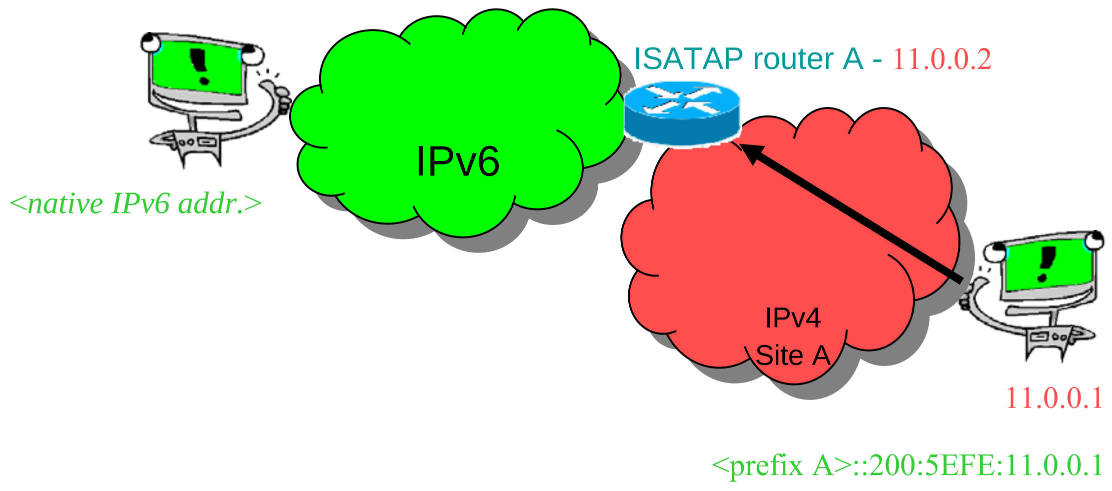


Let's see how ISATAP works. At least one of the end points of an ISATAP tunnel is a router which also has a native IPv6 interface. The router forwards IPv6 packets into or out of the tunnel. A node that belongs to an ISATAP tunnel has to know the IPv4 address of the router. In the illustration below we denote the prefix of the ISATAP tunnel as <prefix A>. 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. In the illustration below this is 11.0.0.2. The IPv4 source address is the IPv4 address of the source (11.0.0.1). At the router, it first verifies that the source IPv6 address corresponds to the source IPv4 address. If this is true, then the router decapsulates the IPv4 header and sends the packet out the IPv6 interface towards the final destination. When an IPv6 packet makes its way back to the ISATAP end point, the packet first reaches the ISATAP router. The router encapsulates the packet with an IPv4 header having a destination that is the lowest 32 bits of the IPv6 destination address (i.e. 11.0.0.1) and sends it out its IPv4 interface. There it reaches the ISATAP end point.

11.0.0.1 <prefix A>::200:5EFE:11.0.0.1

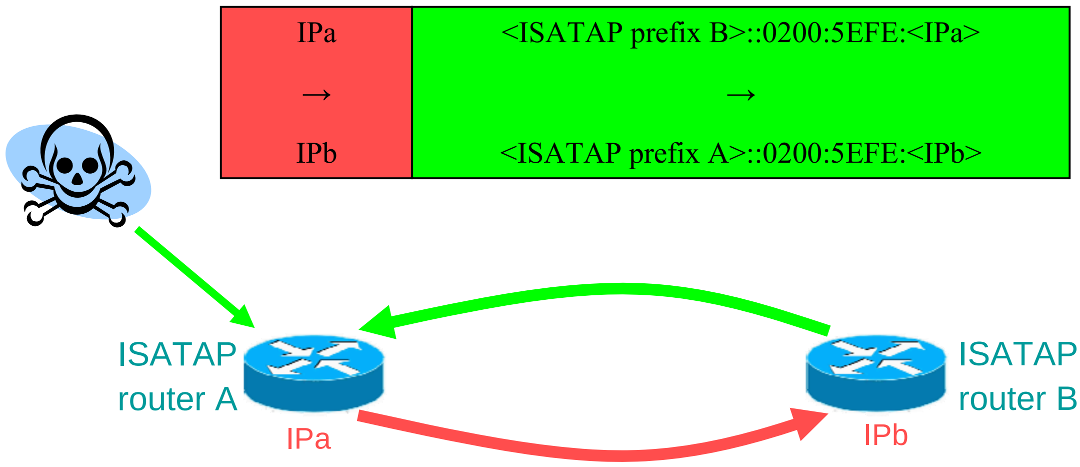
→ →

11.0.0.2 <native IPv6 addr.>



Note that when it forwards the packet into or out of the IPv4 network the router does not really know whether there is an actual ISATAP clinet with that IPV4/IPv6 address. Let's see how an attacker can exploit this.

The attack in Theory


IPa
→
IPb
<ISATAP prefix B>::0200:5EFE:<IPa>
→
<ISATAP prefix A>::0200:5EFE:<IPb>
ISATAP router A
IPa
ISATAP router B
IPb

In the following I shall explain how an attacker can make a specially drafted packet to traverse back and forth from one ISATAP router to another ISATAP router. This enables the attacker to easily overload both router and any other router on the loop between them.

The two victims of this attack are the ISATAP routers - A and B - that service two different ISATAP tunnel prefixes. The IPv4 address of routers A and B are IPa and IPb, respectively. We assume that the two routers both have two interfaces one towards an IPv6 network and another towards an IPv4 network. The attacker starts by sending the above IPv6 packet (the green portion) in the IPv6 network. The destination address has the prefix of router A and the IPv4 address of router B, whereas the source address has the prefix of router B and the IPv4 address of router A. Router A gets the packet through its IPv6 interface and treats it as a regular IPv6 packet that needs to be delivered to an ISATAP client on its tunnel which have IPb as a IPv4 address. So router A encapsulates it accordingly. Namely, the IPv4 destination is IPb and the IPv4 source is IPa. The packet is sent over the IPv4 network. Router B receives the packet through its IPv4 interface. Router B treats the packet as a regular IPv4 packet that originates from one of the end points of its tunnel, since the IPv4 and IPv6 source addresses of the packet correspond (i.e. the IPv4 address is embedded in the IPv6 address).

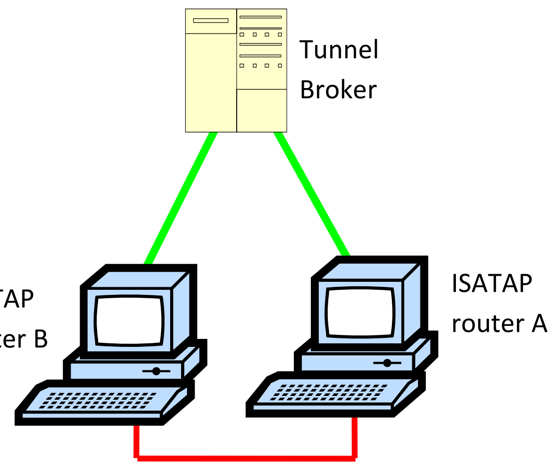
Since the packet is destined outside the tunnel of router B (the IPv6 destination address has prefix of tunnel A). The packet will be sent out on the native IPv6 interface. The native IPv6 routing will forward the packet back to router A. There the loop will start again. Router A will think that the packet is destined to an end point in its tunnel and will encapsulate the packet with an IPv4 header and send it back to the IPv4 network.

This way the packet will loop back and forth between the two routers. The loop will stop once the Hop Limit of the packet will reach 0 (every IPv6 router on the loop decrements it by 1). Since the maximum initial value that can be set by the attacker of the Hop Limit field is 255 the packet will traverse 255 hops on the native IPv6 network before stopping. Now, imagine that the attacker continuously sends such packets. You can see how easily one can overflow the routers.

Note that this is just one example on the attack. Pretty much the same attack can be launched on routers of other automatic tunnels such 6to4 or 6rd or a combination there of.

The attack in practice

To test the attack in practice you need to setup two ISATAP routers each having two interfaces: one connected to an IPv4 network and another connected to an IPv6 network. In the following figure a possible sample network layout is depicted. Two computers will act as two ISATAP routers. The two computers will be connected together on a LAN. This will act as the IPv4 network (marked in red). The ISATAP tunnel interfaces will point to this IPv4 network. In the following I shall assume that the subnet of this IPv4 network is 11.11.11.0/24, and that the computers are assigned the addresses 11.11.11.11 and 11.11.11.12. Moreover, each computer will need an interface to an IPv6 network. Assuming you do not have such an IPv6 network ready to use, this can be achieved by setting up a manual tunnel (marked in green) to a tunnel broker that will provide us with an IPv6 connectivity. The interface to the manual tunnel will be the desired IPv6 interface on each ISATAP router.



A tunnel broker is a gateway on the Internet that allows us to set up a manual tunnel to it on which IPv6 packets can be carried over IPv4 header (much like an ISATAP tunnel). The gateway then does the IPv6 routing for us. There are a few free tunnel broker providers on the Internet; choose any one you like. After registering with the provider you can ask for an IPv6 tunnel. A unique IPv6 address will be assigned to your computer. Since this tunnel, as opposed to ISATAP, is a configured one you need to explicitly configure your computer. Fortunately, some tunnel broker providers will give you the specific commands by which your computer can be configured. Since the gateway on the provider's end needs to be configured too you need to give them your local IPv4 address.

Here are sample commands to configure your local computer on Linux (I assume your local IP address is 11.11.11.11):

```
modprobe ipv6
ip tunnel add tb-ipv6 mode sit remote xx.xx.xx.xx local 11.11.11.11 ttl 255
ip link set tb-ipv6 up
ip addr add 2001:db8:1f06:d2f::2/64 dev tb-ipv6
ip route add ::/0 dev tb-ipv6
```

The above commands do the following in the order of the lines from top to bottom:

1) Make sure the ipv6 module is loaded.

2) Add a tunnel interface (IPv6 encapsulated in IPv4) with the appropriate IPv4 address as the end points. You need to replace xx.xx.xx.xx with the IPv4 address of the tunnel broker. This should be given to you by the provider. The new interface is called “tb-ipv6”

3) Enable the new interface.

4) Set the IPv6 address assigned to your computer by the provider to the tunnel interface (Here I assume the assigned IPv6 address is `2001:db8:1f06:d2f::2`).

5) Add a default route. By default, every IPv6 packet will be routed out this interface.

You should set up two **<u>different</u>** tunnels for both your computers.

The next step is to set up an ISATAP router on both computers. Here is a sample configuration for Linux:

```
ip tunnel add is0 mode isatap local 11.11.11.11 ttl 64
ip link set is0 up
ip addr add 2001:db8:1f06::200:5efe:b0b:b0b/64 dev is0
```

The above commands do the following in the order of the lines from top to bottom:

1) Add an ISATAP tunnel interface over the appropriate local IPv4 address. The new interface is called “is0”

2) Enable the new interface.

4) Assign an IPv6 ISATAP address to the new interface. Note that the lower 32 bits correspond to the local IPv4 address, and that the upper 64 bits correspond to the IPv6 prefix assigned to you by the tunnel broker provider.

You should configure ISATAP accordingly on both computers. If you use

Windows please advice the following page.

Now, the fun part; we should craft the attack packet and send it from one of the computers. I like to use a tool called Scapy to craft my IP packets. Scapy is based on the Python programming language and framework. In the following I assume we have the following configuration (the differences are marked in bold):

ISATAP router A:

```
IPv4 address: 11.11.11.11
IPv6 address: 2001:db8:1f06::200:5efe:b0b:b0b
```

ISATAP router B:

```
IPv4 address: 11.11.11.12
IPv6 address: 2001:db8:1f07::200:5efe:b0b:b0c
```

Here is the command that generates and sends the packet in Scapy:

```
Attack_Packet = IPv6(dst="2001:db8:1f07::200:5efe:b0b:b0b",
src="2001:db8:1f06::200:5efe:b0b:b0c")/ICMPv6EchoRequest()
send(Attack_Packet)
```

This packet can be sent from either computer. Let's assume we send it from router A. Since the prefix of the IPv6 destination address does not belong to router A then the packet is routed to the IPv6 network over the IPv6 tunnel broker interface to the tunnel broker. The tunnel broker routes the packet to router B. Since the prefix of the IPv6 destination address is the prefix of router B's ISATAP tunnel it will route it through its ISATAP interface, namely over its IPv4 interface while the packet is encapsulated with a destination address of 11.11.11.11 (this is the lower 32 bits of the IPv6 destination address). Router A will receive the packet, and decapsulates the IPv4 header. Since the IPv6 destination address is not that of router A it will again route it over the IPv6 interface, i.e. the tunnel broker interface. This loop continues until the Hop Limit field in the IPv6 header is zeroed out. If the initial Hop Limit value is 255, then the packet will loop 85

(=256/3) times before it is dropped. This is true since the loop contains 3 IPv6 routers (ISATAP routers A and B, and the tunnel broker) each decrements the Hop Limit by 1.

Mitigations

An Internet draft (draft-ietf-v6ops-tunnel-loops - soon to become RFC 6324) has been co-authored by Fred Templin and I to document possible ways to mitigate this routing loop attack. None of the proposed mitigation measures are ideal. However, the most reliable and simple measures proposed in the draft are operational ones. In these mitigation measures the network administrator explicitly removes the possibility for a routing loop. Here are some of the operational mitigations we propose in the draft:

Avoiding multiple tunnels

This measure mitigates the attack by simply allowing for only a single IPv6 tunnel to operate in a bounded IPv4 network. In the example above this mitigation measure will stop the attack by simply filtering all IPv4 packets (or IPv4 packet which encapsulate IPv6 packets) between the two ISATAP routers (A and B). This way no loop can be formed between the two routers. This approach has the advantage that it avoids the attack altogether. However, it requires careful configuration management which may not be tenable in large and/or unbounded IPv4 networks.

A Single Border Router

It is reasonable to assume that a tunnel router shall accept or forward tunneled packets only over its tunnel interface. It is also reasonable to assume that a tunnel router shall accept or forward IPv6 packets only over its IPv6 interface. If these two interfaces are physically different then the network operator can mitigate the attack by ensuring that the following condition holds: there is no path between these two interfaces that does not go through the tunnel router.

The above condition ensures that an encapsulated packet which is transmitted over the tunnel interface (in the above example, ISATAP router B) will not get to another tunnel router (in the above example, ISATAP router A) and from there to the IPv6 interface of the first router. The condition also ensures the reverse direction, i.e., an IPv6 packet which is transmitted over the IPv6 interface (in the above example, by ISATAP router A) will not get to another tunnel router (in the above example, ISATAP router B) and from there to the tunnel interface of the first router. This condition essentially means that a tunnel router must be the only border router between the IPv6 network and the IPv4 network to which it is attached (as in broadband home network, for example).

Known IPv4 Address Check

Another approach is to simply pre-configure each tunnel router with the set of IPv4 addresses that are authorized to use the tunnel. In the above example, this means that ISATAP router B will forward a packet into its tunnel to IPv4 addresses that are explicitly authorized (via configuration at the router B) to use ISATAP B tunnel. This, of course, assumes that ISATATP router A is indeed not authorized to use ISATATP router B's tunnel. ISATAP router A will also filter received tunnel packet that are sent from end points which are not authorized to use ISATATP router A's tunnel Further (Assuming ISATATP router B is indeed not authorized to use ISATATP router A's tunnel)

These and other mitigation measures are detailed in the draft.