Technical Documentation

Services in IPv6-only networks

Joris Claassen

Table of Contents

1. Why did I do research on this subject

My study at the Hogeschool Utrecht is about the configuration and management of IT systems. I had to do an internship for my study. This internship needed to be at a location where there was a big chance to learn a lot of new skills. The RIPE NCC was such a place for me. It was very close to the top of Internet resource distribution, and it was actively promoting IPv6. Promoting IPv6 was quite a big deal for me. I wanted to learn more on this subject, as it is very new and buzzing in the industry.

A lot of companies will implement IPv6 in their networks in the years to come. They have to because IPv4 support will gradually be phased out in favour of this new network protocol. The RIPE NCC wanted to have an overview of services in IPv6-only networks. How do these services work? Do these services work at all? Questions like these were asked. I did not know the answers yet – almost no one knew. IPv6-only networks are very uncommon, even for early adopters at these stages of the implementation. Not even 10% of the world even runs IPv6 and IPv4 (dual stack) together. So running IPv6-only? Why should they?

The RIPE NCC has been running dual stack since 2004. Since then, it has made a lot of effort to get the spotlight on IPv6. End of the line; almost everyone waited for the pool of available IPv4 address space to run out, so they **have** to deploy IPv6. This goes for all ISPs in the Netherlands, except for XS4ALL. XS4ALL has made sure that it can offer native IPv6 to their residential customers before the pool of IPv4 address space runs out. Most of the bigger corporations, especially those with a big focus on the Internet (Google, Facebook) have been running dual stack since the World IPv6 Launch, held in June 2012.

In the following document I will explain the technical difficulties system operators will run into as they implement IPv6 in their networks. As there will be more and more IPv6-only networks in the years to come, I hope this document can be used as a blueprint to help system operators. Each network is specific, so of course many adaptations will have to be made, but it is very practical to know what services do work on IPv6, and which do not – and which require some configuration hacking to do so.

I sincerely hope this research saves you some precious time!

2. How did I do research on this subject

To start with a project like this is kind of hard. There is little to no reference, and when there is it is mostly scattered all over the Internet. As a result I had to make a reference for myself. I started out with some research on how to build an IPv6-only network and how to manage it. This research consisted of looking into other people's findings of implementing dual stack networks, as there was little information to be found about IPv6-only networks. I saved some precious links to websites, and carried on to the next phase of my testing: the virtual environment.

When I started researching I had to pick a virtualisation platform. As I had a Mac Mini to work with, I went ahead and chose VirtualBox; a free of charge virtualisation platform. The Mac Mini was connected to the office LAN though, which made it hard for me to have full control over the network environment. This is when I decided I needed an IPv6 router. To make my server reachable from all over the network, the Mac Mini was put away, and a Dell PowerEdge server was assigned for the job. I installed CentOS 6.2 on it, and on top of that VirtualBox with phpVirtualBox (as seen in Figure 1). This combination of software made it possible to control all the VMs from a remote location.

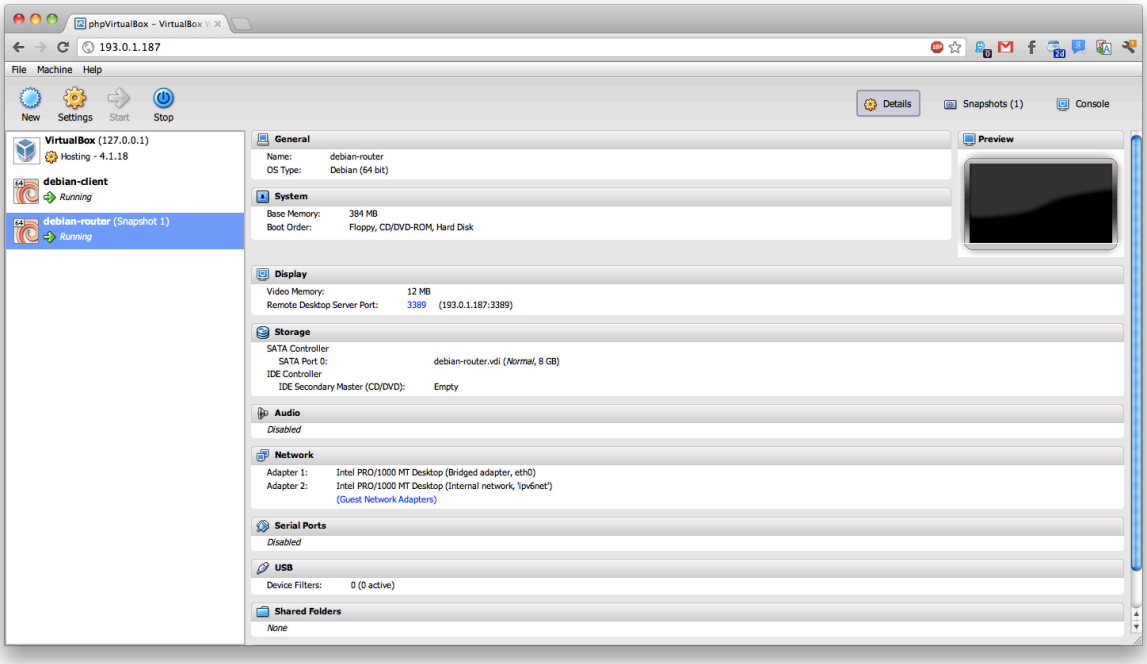

Figure 1 – Screenshot of phpVirtualBox

To check whether the virtual network was clean of any strange network traffic I checked it with Wireshark. The next part was to install a virtual IPv6-only router, which I could connect other VMs to. This router was based on Debian, and had its addressing assigned via a tunnel broker, so I had full control over the virtual network. The router could then be used to test all kind of

networking services to the full extent, as well as some server software. The client software was tested on the VMs that would afterwards act as test servers for services such as DNS.

The test servers I used were based on CentOS, Debian and Windows Server 2008 R2 because these three are the most commonly used in ISP networks, and there was in-house knowledge available for them (which was a requirement for my employment as Intern). These different operating systems allowed me to test differences between their behaviour in IPv6-only networks. Some results surprised me and some were as I expected. Testing services involved a lot of time spent on Google looking for people running into similar issues, or trying to fix scripts myself. Also it involved a lot of crosschecking of configurations on different operating systems; most of the Linux basics are the same, yet fine-tuning is done in slightly different ways.

The results I got out of this testing are compiled in the rest of this document. First off, there is a matrix that lists all the services and gives them a colour based on usability in an IPv6-only network. After the matrix there is a more precise definition of services and an explanation about whether they work in IPv6-only networks, need adjustments or are broken.

3. Results of my research

	CentOS 6.2		Debian 6.0			Windows Server 2008 R2	
NDP-server	radvd 1.6		radvd 1.6			add components	
NDP-client	rdnssd (ndisc6 1.0.1 from EPEL)		rdnssd 0.9.8			rdnssd-win32	
DHCPv6-server	dhcp 4.1.1-P1	dibbler-server 0.4.1	dhcp 4.1.1-P1-15	wide-dhcpv6-server 20080615	dibbler-server 0.7.1	add components	dibbler-server 0.8.2
DHCPv6-client	dhcp 4.1.1-P1	dibbler-client 0.4.1	dhcp 4.1.1-P1-15	wide-dhcpv6-client 20080615	dibbler-client 0.7.1	pre-installed	dibbler-client 0.8.2
DNS-server	bind 9.7.3		bind 9.7.2-P3			add components	
DNS-resolver	bind 9.7.3		bind 9.7.2-P3			pre-installed	
SSH-server	openssh 5.3		openssh 5.5			cygwin	
SSH-client	openssh 5.3		openssh 5.5			putty 0.62	cygwin
NTP-client	ntp 4.2.4-P8		ntp 4.2.6-P2			built-in	
SNMP-server	net-snmp 5.5		net-snmp 5.4.3			add components	
SNMP-client	net-snmp 5.5		net-snmp 5.4.3			net-snmp ports by snmpsoft	
SMB-server	samba 3.5.10		samba 3.5.6			pre-installed	
SMB-client	kernel 2.6.32		kernel 2.6.32			pre-installed	
NFS-server	nfs-utils 1.2.3		nfs-common =< 1.2.3			add components	
NFS-client	nfs-utils 1.2.3		nfs-common =< 1.2.3			add components	
Syslog-server	rsyslog 4.6.2		rsyslog 4.6.4			3rd party	
Syslog-client	rsyslog 4.6.2		rsyslog 4.6.4			3rd party	

	means	working after default system installation
	means	working after default installation from repository
	means	working after some manual configuration
	means	not working

4. Notes

Please note the following:

- These services are **limited to** commonly used server/client configurations. I have not been able to test all (especially Windows) software.
- Protocols such as HTTP, FTP and SMTP have **not** been included in this documentation. This is because they have already been thoroughly documented on the Internet.
- The following configurations have been tested in an IPv6-only environment, they **might** work in a dual stack environment (which is likely because of full IPv6 support)
- The following configurations have been tested on CentOS 6.2, Debian 6.0 and Windows Server 2008 R2. Other operation systems **might** work
- There will need to be **at least** and /64 IPv6 subnet available to test Dynamic Host Configuration Protocol version 6 (DHCPv6) or Stateless Autoconfiguration (SLAAC) (confirming to RFC standards)

5. Networking

5.1. NDP

NDP (Neighbor Discovery Protocol) is used by IPv6 nodes on the same link. It is used to discover each other's presence, to determine each other's link-layer addresses, to find routers, and to maintain reachability information about the paths to active neighbors. The protocol is mandatory for IPv6 to work, as IPv6 hosts need Neighbor Discovery and Neighbor Advertisement to work (it is a replacement for the IPv4 ARP messages). Therefore it is also required when one is using DHCPv6 for addressing, or using static addressing.

NDP can be implemented using a hardware router or a daemon (software). This documentation will describe the method using a daemon running on Linux.

NDP is defined in RFC 4861, which can be found here: https://tools.ietf.org/html/rfc4861

1.1.1.1. radvd

The Router Advertisement Daemon (radvd) is an open source product that implements link local IPv6 router addresses and routing prefixes. radvd is by far the most used daemon to send NDP packets from Linux. If there is need for a Linux machine with routing capabilities, there is no doubt radvd will need to be installed.

radvd has several options, of which not all are equally important. For basic functionality, radvd just needs a /64 prefix to advertise. It also has options to enable or disable SLAAC, and can be used to tell servers and computers to use other configuration protocols (such as DHCPv6) to configure options.

Recursive DNS Server (RDNSS) is a relatively new way of setting the client's DNS server. It requires the operating system to parse the RDNSS record, and add it as a DNS resolver.

My example configuration for a stateful configuration:

```
/etc/radvd.conf:

interface eth0 {
      # Send out Router Advertisements
      AdvSendAdvert on;
      # Use DHCPv6 addressing
      AdvManagedFlag on;
      # Use DHCPv6 optional configuration such as DNS
      AdvOtherConfigFlag on
      # Range for clients
      prefix 2001:0DB8::/64 {
            # Advertise when connected
            AdvOnLink on;
            # Make clients use SLAAC
            AdvAutonomous off;
      };
      # Set clients DNS resolver using RDNSS
      RDNSS 2001:0DB8::1:1 {};
};
```

For a stateless configuration the AdvManagedFlag setting has to be set to "off", and AdvAutonomous should be "on". radvd can be started and stopped using the /etc/init.d/radvd script on CentOS and Debian.

1.1.1.2. rdnssd

Recursive DNS Daemon (RDNSSD) is a daemon that enables the RDNSS flag to be fully understood by an operating system. rdnssd will import the RDNSS server into the resolver file. rdnssd can be installed from the Debian repositories. For CentOS the Red Hat Enterprise Linux Extra's (EPEL) repository will have to be installed, after which the ndisc6 package (which includes rdnssd) can be installed.

In CentOS there is no service installed, so rdnssd will have to be run manually. There are two commands to run:

```
1.    rdnssd

2.    /etc/rdnssd/merge-hook
```

The merge-hook script will merge the contents of the rdnssd buffer with /etc/resolv.conf.

To run rdnssd in Debian /etc/init.d/rdnssd script can be used to control the daemon, which in turn will merge the RDNSS server in /etc/resolv.conf for use of DNS lookups.

To enable Windows usage of rdnssd, there is a Windows port of rdnssd: rdnssd-win32. When using Windows 2003 or newer, the rdnssd-win32 will add the RDNSS server as a DNS resolver.

5.2. DHCPv6

DHCPv6 is a protocol to manage the addressing of a network using a server daemon sending out an address range and extra options. In addition to the server daemon, the client operating system must also support configuration-using DHCPv6 (either built in or using a daemon).

DHCPv6 can be used in two ways. The first way of using DHCPv6 is to grant clients addresses from a pool while also using DHCPv6 to push configuration options. This is called stateful configuration. The other option is to use DHCPv6 combined with SLAAC for addressing, while using DHCPv6 for configuration options (like DNS servers). This is called stateless configuration.

I have tested four different DHCPv6 servers that are production ready. The Internet Systems Consortium (ISC) has DHCPv6 support in their ISC DHCP package. Microsoft has built in a DHCPv6 server in Windows 2008 and higher. Then there are two open source projects, WIDE-DHCPv6 and Dibbler.

DHCPv6 is defined in RFC 3319, which can be found here: https://tools.ietf.org/html/rfc3319

5.2.1. ISC DHCP

ISC's DHCP software is the most widely used open source DHCP implementation on the Internet. It is a carrier and enterprise grade solution to your host configuration needs. The package is the preinstalled DHCPv6 implementation on CentOS and Debian.

DHCP supports DHCPv6 as of version 4.0.

5.2.1.1. dhcpd

CentOS users can control the daemon using the /etc/init.d/dhcpd script.

Using Debian, a separate instance of dhcpd has to be run for DHCPv6 to work as the /etc/init.d/dhcpd script in Debian only supports DHCP. The bug-report for this issue can be found here: http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=592539

Here is an example of how dhcpd can be run on interface eth0 using IPv6 on Debian:

```
/usr/sbin/dhcpd -6 -cf /etc/dhcp/dhcpdv6.conf eth0
```

This is the configuration I used during testing:

```
/etc/dhcp/dhcpdv6.conf:

default-lease-time 600;
max-lease-time 7200;
#Global options
option dhcp6.name-servers 2001:db8::1:1;
option dhcp6.domain-search "example.ripe.net";
subnet6 2001:db8:1:1::/64 {
      # Range for clients
      range6 2001:db8::1000 2001:db8::2000;
}
```

5.2.1.2. dhclient

dhclient is the DHCP client supplied by the ISC. The version in the Debian stable repositories has a bug though; it uses configuration files from a previous version which prevent it from being a usable DHCPv6 client. Either installing an unstable version or using one of the alternative

DHCPv6 clients listed in this documentation make DHCPv6 work using Debian. The bug report of this issue can be found here: http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=591589

To configure dhclient on CentOS the network scripts will have to be edited.

This is the stateful configuration I used during testing:

```
/etc/sysconfig/network-scripts/ifcfg-eth0:

DEVICE="eth0"
HWADDR="08:00:00:00:00:00"
IPV6INIT="yes"
IPV6_AUTOCONF="no"
DHCPV6C="yes"
NM_CONTROLLED="yes"
ONBOOT="yes"
TYPE="Ethernet"
```

To achieve stateless configuration the IPV6_AUTOCONF setting will have to read “yes”.

5.2.2. WIDE-DHCPv6

WIDE-DHCPv6 is a project that evolved from the KAME project. The KAME project was the joint effort of six companies in Japan to provide a free stack of IPv6, IPsec and Mobile IPv6 for BSD variants. WIDE-DHCPv6 has both server and client parts, which can be installed as packages on Debian. There are no precompiled packages available for CentOS, so I suggest that one of the alternative DHCPv6 servers and clients listed in this documentation are used – they work just as well.

5.2.2.1. *wide-dhcpv6-server*

The wide-dhcpv6-server is an easy to configure DHCPv6 server. The service supports all the normal DHCPv6 options like NTP, SIP and NIS server. The server is not pre-installed on any operating system.

This is the configuration I used during testing:

```
/etc/wide-dhcpv6/dhcp6s.conf:

#Global options
option domain-name-servers 2001:db8:1::1;
option domain-name "example.ripe.net";

interface eth0 {
      address-pool pool1 3600;
};

pool pool1 {
      # Range for clients
      range 2001:db8::1000 to 2001:db8::2000;
};
```

5.2.2.2. wide-dhcpv6-client

The wide-dhcpv6-client does not come pre-installed on any operating system. It can be installed from the Debian repositories. The client supports all the normal DHCPv6 options like NTP, SIP and NIS server.

This is the configuration for stateful DHCPv6 that I used during testing:

```
/etc/wide-dhcpv6/dhcp6c.conf:

interface eth0 {
      # Ask for a permanent address
      send ia-na 1;
      # Ask for a delegated prefix
      send ia-pd 0;
      # Request DHCPv6 options
      send domain-name-servers,domain-name,ntp-servers,sip-server-
address,sip-server-domain-name,nis-server-address,nis-domain-
name,nisp-server-address,nisp-domain-name,bcmcs-server-
address,bcmcs-server-domain-name;
};
id assoc na 1 {
};
id assoc pd {
};
```

When running wide-dhcpv6-client in a stateless environment a simpler configuration suffices:

```
interface eth0 {
      information-only;
      # Request DHCPv6 options
      send domain-name-servers,domain-name,ntp-servers,sip-server-
address,sip-server-domain-name,nis-server-address,nis-domain-
name,nisp-server-address,nisp-domain-name,bcmcs-server-
address,bcmcs-server-domain-name;
}
```

5.2.3. Dibbler

Dibbler is a portable DHCPv6 implementation. It supports both stateful and stateless autoconfiguration for IPv6. Currently Linux 2.4/2.6 and Windows XP and Windows 2003 ports are available.

5.2.3.1. dibbler-server

Dibbler-server is the only portable DHCPv6 server I tested. The server is lightweight and supports all the normal DHCPv6 options like NTP, SIP and NIS server.

This is the dibbler configuration I used during testing:

```
/etc/dibbler/server.conf:

iface eth0
{
      # Time between DHCPv6 lease renew
```

```
        T1 600
        # Time before querying another server in case of problems
        T2 900
        prefered-lifetime 1800-3600
        valid-lifetime 3600-86400
        class
        {
              # Range for clients
              pool 2001:db8::1000-2001:db8::2000
        }
        #Global options
        option dns-server 2001:db8::1:1
        option domain example.ripe.net
}
```

5.2.3.2. dibbler-client

The Dibbler client can work without configuration. Without configuration Dibbler will try to configure stateful DHCPv6.

To run stateless DHCPv6 a configuration is required. This is the configuration for stateless DHCPv6 that I used during testing:

/etc/dibbler/client.conf:

```
stateless
iface eth0
{
        # Request DHCPv6 options
        option dns-server
        option domain
        option ntp-server
        option time-zone
        option sip-server
        option sip-domain
        option nis-server
        option nis-domain
        option nis+-server
        option nis+-domain
}
```

5.2.4. Windows Server 2008 R2 DHCP

5.2.4.1. Windows Server 2008 R2 DHCP server

Windows Server 2008 R2 has a DHCPv6 implementation that can be enabled through the standard “Add Roles” screen Windows uses to install services. After the DHCP service has been installed, there is a choice either to run stateful or stateless DHCPv6 using the DHCP configuration control panel.

5.2.4.2. Windows Server 2008 R2 DHCP client

The Windows Server 2008 R2 DHCP client also supports a DHCPv6 implementation. Windows will detect whether stateful or stateless DHCPv6 is used, and configure the system accordingly.

6. Services

6.1. DNS

DNS is defined in RFC 4343, which can be found at http://tools.ietf.org/html/rfc4343

6.1.1. ISC BIND

ISC BIND is widely used DNS software on the Internet. It provides a platform on top of which organisations can build distributed computing systems with the knowledge that those systems are fully compliant with published DNS standards.

ISC BIND is open source software that implements the Domain Name System (DNS) protocols for the Internet. It is a reference implementation of those protocols, but it is also production-grade software.

ISC BIND supports IPv6 as of version 9.0.

6.1.1.1. named

named is the ISC's DNS server daemon. By default it will listen to IPv4 addresses only. To change this behaviour and get named to work in an IPv6-only environment, some changes will have to be made in /etc/named.conf.

Changes I made to the default configuration:

```
/etc/named.conf

#IPv4:
#listen-on

#IPv6:
listen-on-v6
```

After that, the standard options sent to the daemon will have to be edited as well. These files reside in different folders for respectively CentOS (/etc/sysconfig) and Debian (/etc/defaults).

```
/etc/sysconfig/named | /etc/defaults/named

#IPV6:
OPTIONS="-6"
```

Make sure your ip6tables do not block port 53. Afterwards a restart of the daemon will put the new changes in action. The daemon can be started and stopped using the /etc/init.d/named script on CentOS and Debian.

6.1.1.2. resolver

ISC BIND comes with a DNS resolver library. This resolver is capable of translating requests for websites (e.g. ripe.net) using an IPv4 and/or IPv6 address. The address that the resolver uses to send address requests is configured using the /etc/resolv.conf file, and can hold both IPv4 and IPv6 addresses.

Changes I made to the default configuration:

```
/etc/resolv.conf

#IPv4:
#nameserver 192.168.1.1

#IPv6:
nameserver 2001:db8::1:1
```

6.1.2. Windows Server 2008 R2 DNS

6.1.2.1. Windows Server 2008 R2 DNS Server

Windows Server 2008 R2 has a DNS server that can be added through the add roles dialog in Windows. It supports IPv6 out of the box. It will create AAAA records for the servers as well as forward AAAA records found on the DNS servers it was configured to forward.

6.1.2.2. Windows Server 2008 R2 DNS resolver

The DNS resolver integrated in Windows Server 2008 R2 supports IPv6.

This resolver can either be configured using the Network Connection Properties GUI, or automatically, using DHCPv6 or rdnssd-win32.

6.2. SSH

SSH is defined in RFC 4253, which can be found here: http://tools.ietf.org/html/rfc4253

6.2.1. OpenSSH

OpenSSH is a free version of the SSH connectivity tools that a lot of technical Internet users rely on. Users of telnet, rlogin, and ftp may not realise that their password is transmitted across the Internet unencrypted, but it is. OpenSSH encrypts all traffic (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other attacks.

6.2.1.1. sshd

sshd is the OpenSSH daemon, which accepts remote logins. It has quite some options, of which ListenAddress is particularly interesting for IPv6-only environments. To make sshd listen on all the IPv6 addresses of the server, /etc/ssh/sshd_config will have to be edited.

Changes I made to the default configuration:

```
/etc/ssh/sshd_config

#IPv4
#ListenAddress 0.0.0.0 # or any other IPv4 address

#IPv6
ListenAddress 2001:DB8::1
```

Make sure your ip6tables do not block TCP port 22. Afterwards a restart of the daemon will put the new changes into action. The daemon can be started and stopped using the /etc/init.d/sshd script on CentOS and Debian.

6.2.1.2. ssh

Secure Shell (SSH) is a network protocol for secure data communication, remote shell services or command execution and other secure network services between two networked computers that it connects via a secure channel over an insecure network.

The ssh client is ready for use in IPv6-only networks. Instead of:

```
#IPv4:
ssh user@192.168.1.1

#IPv6
ssh user@2001:DB8::1
```

should be used to login.

When connecting to a hostname (example.com), the SSH client will first try to look up an AAAA record. When this cannot be found, it might try to connect to an IPv4 address located in the A record. As the testing occurred in an IPv6-only network, this will give an error. This also applies for the SCP client listed in 6.2.1.3.

6.2.1.3. scp

Secure Copy or SCP is a means of securely transferring computer files between a local host and a remote host or between two remote hosts. It is based on the Secure Shell (SSH) protocol.

The scp client is ready for use in IPv6-only networks. Instead of:

```
#IPv4:
scp SourceFile user@192.168.1.1:directory/TargetFile

#IPv6:
scp SourceFile user@[2001:db8::1]:directory/TargetFile
```

should be used. These brackets are used by scp to differentiate the host from the directory, as IPv6 uses 2001:DB8::1 instead of 192.168.1.1 in IPv4.

6.2.2. Windows Server 2008 R2 SSH

6.2.2.1. PuTTY

PuTTY is a free implementation of Telnet and SSH for Windows and Unix platforms, along with an xterm terminal emulator. It makes it possible for a Windows client to connect to a telnet or SSH server.

PuTTY supports IPv6 as of version 0.62

6.2.2.2. OpenSSH port through Cygwin

OpenSSH can be installed on Windows clients through the Cygwin setup.exe. The package that needs to be installed is OpenSSH. After OpenSSH is installed it can be run through the Cygwin terminal emulator, or from the \cygwin\bin folder.

6.3. NTP

Network Time Protocol (NTP) is a networking protocol for synchronising the clocks of computer systems over data networks. In operation since before 1985, NTP is one of the oldest Internet protocols in use.

NTP supports IPv6 as of version 4.0.

NTP is described in RFC 5905, which can be found here: http://tools.ietf.org/html/rfc5905

6.3.1. ntpd

The Network Time Protocol daemon (ntpd) is an operating system daemon program that maintains the system time in synchronisation with time servers using the NTP.

6.3.1.1. ntpd

To enable NTP in an IPv6-only environment the server that is going to be queried by NTP will need to be IPv6 enabled. pool.ntp.org provides the official server pool of NTP servers, which are then subdivided in more specific areas (europe.pool.ntp.org for example). The whole pool is not yet IPv6 enabled. The hosts starting with 2.*.pool.ntp.org do support IPv6 though, so 2.europe.pool.ntp.org does support IPv6. Alternatively, RIPE NCC's NTP server (ntp.ripe.net) does support IPv6 as well.

Changes I made to the default configuration:

```
/etc/ntp.conf

#IPv4:
#*.pool.ntp.org

#IPv6:
2.*.pool.ntp.org
```

Afterwards a restart of the daemon will suffice to put the new changes in action. The daemon can be controlled using the /etc/init.d/ntpd script.

6.3.2. Windows Server 2008 R2 NTP

6.3.2.1. Windows Server 2008 R2 NTP client

The NTP client integrated in Windows Server 2008 R2 supports IPv6.

The default NTP server the Windows NTP client tries to reach (time.windows.com) is only IPv4 at the moment. Adding an IPv6 enabled NTP server through the adjust time/date GUI works.

6.4. SNMP

6.4.1. Net-SNMP

Simple Network Management Protocol (SNMP) is a widely used protocol for monitoring the health and welfare of network equipment (e.g. routers), computer equipment and even devices like UPSs. Net-SNMP is a suite of applications used to implement SNMP v1, SNMP v2c and SNMP v3 using both IPv4 and IPv6.

Net-SNMP supports IPv6 as of version 5.0.

SNMP is described in RFC 2711, which can be found here: http://tools.ietf.org/html/rfc2711

6.4.1.1. snmpd

snmpd is an SNMP agent which binds to a port and awaits requests from SNMP management software. Upon receiving a request, it processes the request(s), collects the requested information and/or performs the requested operation(s) and returns the information to the sender. To enable snmpd to bind to IPv6 sockets some changes in the configuration will have to be made. The rwcommunity and rocommunity are used for traditional access control, whereas the com2sec (community to security) is View Access Control Model (VACM) based access control. Both the access control manners are IPv6 ready, they just need a 6 behind their respective configuration lines.

Depending on the choice of access control made there will either be com2sec or ro/rwcommunity lines in the configuration. In this example the default network is shared to localhost, using the public community string.

Changes I made to the default configuration:

```
/etc/snmp/snmpd.conf

#IPv4:
#com2sec localhost default public
#com2sec examplenetwork 192.168.1.0/24 public
#rocommunity *
#rwcommunity *

#IPv6:
com2sec6 localhost default public
com2sec6 examplenetwork 2001:db8::/64 public
rocommunity6 *
rwcommunity6 *
```

Make sure your ip6tables do not block UDP port 161. Afterwards a restart of the daemon will put the new changes into action. The daemon can be started and stopped using the /etc/init.d/snmpd script on CentOS and Debian.

6.4.1.2. snmpget

snmpget is a little utility made to get a specific SNMP value. It can be used over TCP and UDP, both on IPv4 and IPv6. To do a query on an IPv6 network the syntax is a little bit different, since the default behaviour of snmpget is to use UDP over IPv4.

To force snmpget to use UDP over IPv6, the following syntax should be used:

```
snmpget -v 1 -c public udp6:[2001:db8::1]:161 1.3.6.1.2.1.1.3.0
```

The version, community and MIB can of course be different. This syntax works for other Net-SNMP utilities as well. Instead of udp6, tcp6 can be used too if the SNMP server serves over TCP.

6.4.2. Windows Server 2008 R2 SNMP

6.4.2.1. Windows Server 2008 R2 SNMP service

Windows Server 2008 R2 has a SNMP service that can be added through the add roles dialog in Windows. This service does support IPv6. After it has been installed SNMP appears in the services configuration. In that services configuration there are options to configure the community string, and the hosts Windows will accept SNMP packets from.

6.4.2.2. Net-SNMP port by SnmpSoft

A Windows port of some Net-SNMP tools is also available from Snmpsoft. They behave exactly like their Net-SNMP counterparts in Linux environments. The tools available for free are SnmpGet, SnmpSet, SnmpTrapGen and SnmpWalk.

6.5. SMB

Server Message Block (SMB), also known as Common Internet File System (CIFS) operates as an application-layer network protocol mainly used for providing shared access to files, printers, serial ports, and miscellaneous communications between nodes on a network.

6.5.1. Linux Kernel

The Linux Kernel is the core of all Linux distributions, such as CentOS and Debian. It has a CIFS module built in since version 2.6. This module makes sure there is an option to mount SMB shares using the default mount command, as well as adding an option to make the mount permanent using /etc/fstab.

6.5.1.1. mount

An example for mounting an IPv6 SMB mount point using the CIFS module:

```
mount -t cifs //SERVERNAME/share /mnt/share -o
ip=2001:db8::1000,user=user_name,pass=secret
```

To make the same mount permament first make sure the cifs-utils package is installed, then add the following entry to /etc/fstab:

```
//SERVERNAME/share /mnt/share cifs
rw,ip=2001::db8::1000,user=user_name,pass=secret
```

Please note that putting a username and password in a readable file might not be such a good idea. Alternatively there is an option to hide these credentials using a file with different permissions. Instead of:

```
#Unsafe
user=username,pass=password

#Safe
credentials=/path/to/file
```

should be used. The file /path/to/file should look like this:

```
username=user_name
password=secret
```

6.5.2. Samba

Samba is the standard Windows interoperability suite of programs for Linux and Unix. Since 1992, Samba has provided secure, stable and fast file and print services for all clients using the SMB protocol, such as all versions of DOS and Windows, OS/2, Linux and many others.

Samba supports IPv6 as of version 3.2.

6.5.2.1. smbd

smbd is the Samba Daemon. The default configuration enables all shared folders to be reached over IPv4 and IPv6. No extra configuration additional to the standard IPv4 configuration is necessary.

6.5.2.2. smbclient

smbclient is the Samba client. Samba client supports IPv6 the same way as it supports IPv4. Please note that not all Linux file managers may support the IPv6 notation yet. smbclient as-is, in the console, does support it when using the following syntax:

```
smbclient '\\2001:db8::1000\share'
```

6.5.3. Windows Server 2008 R2 SMB

6.5.3.1. Windows File Sharing

Windows File Sharing uses SMB by default and it is fully supported over IPv6. Make sure to add the File Sharing role to the server before attempting to share files. To share files, go into the Server Manager, File Services and then the Share and Storage Management. From here, you can provision a share, which will be available to mount over IPv6.

6.5.3.2. Windows SMB client

The Windows SMB client is integrated into Windows Server 2008 R2. Adding a share is as simple as going to the Computer window, and Map a network drive.

Use the syntax:

```
\\ipv6--host.ipv6-literal.net\share
```

For example, 2001:db8::1 would translate to 2001-db8--1.ipv6-literal.net.

6.6. NFS

Network File System (NFS) is a distributed file system protocol originally developed by Sun Microsystems in 1984. It allows a user on a client computer to access files over a network in a manner similar to how local storage is accessed.

NFS is described in RFC 3530, which can be found here: http://tools.ietf.org/html/rfc3530

6.6.1. Linux Kernel

To use the NFS-module for sharing files over NFS, nfs-utils should be installed because the kernel module cannot operate by itself.

6.6.1.1. *nfsd*

The nfsd daemon runs on a server and actually handles client requests for file system operations, as the kernel does not do this by itself.

Please note that nfs-utils in the Debian stable repositories does not support IPv6 yet, because the version is too old. The version in the unstable repository of Debian does work – yet it is not recommended to mix various versions of repositories as it can seriously mess up your system..

nfs-utils supports IPv6 as of version 1.2.3

The next step is to setup a folder shared for a host in the exports file. To do this, add the following entry to the /etc/exports file:

```
/share 2001:db8::1000(permissions)
```

It is also possible to create a NFS share accessible for a whole subnet (in this case a /64). This can be done by adding the following entry to the /etc/exports file:

```
/share 2001:db8::/64(permissions)
```

Make sure your ip6tables do not block TCP port 2049. Afterwards a restart of the daemon will put the new changes into action. The daemon can be started and stopped using the /etc/init.d/nfsd script on CentOS and Debian.

6.6.1.2. *mount*

An example for mounting an IPv6 NFS mount:

```
mount –t nfs '[2001:db8::1000]:/share' /mnt/share
```

To make the same mount permanent first make sure the nfs-utils package is installed, then add the following entry to /etc/fstab:

```
[2001:db8::1000]:/share /mnt/share default 0 0
```

6.6.2. Windows Server 2008 R2 NFS

6.6.2.1. *Windows File Sharing*

NFS over IPv6 is natively supported in Windows Server 2008 R2. Make sure to add the NFS role to the server before attempting to share files. To share files, go into the Server Manager, File Services and then the Share and Storage Management. From here, you can provision a share, which will be available to mount over IPv6

6.6.2.2. *Windows NFS Client*

The Windows NFS client is integrated into Windows Server 2008 R2. Make sure to add the NFS role to the server before attempting mount a NFS share. Adding a share is as simple as going to

the Computer window, and Map a network drive. Most likely they will have to be connected using different credentials as the NFS server won't have the same users and passwords. Use the syntax

```
\\20001:db8::1000\share
```

6.7. Syslog

Syslog is a standard for computer data logging. It separates the software that generates messages from the system that stores them and the software that reports and analyses them. Syslog can be used for computer system management and security auditing as well as generalised informational, analysis, and debugging messages. It is supported by a wide variety of devices (like printers and routers) and receivers across multiple platforms. Because of this, syslog can be used to integrate log data from many different types of systems into a central repository.

Syslog is described in RFC 5424, which can be found here: http://tools.ietf.org/html/rfc5424

6.7.1. rsyslog

rsyslog supports IPv6 as of version 1.14.1.

6.7.1.1. rsyslogd

rsyslogd is the default syslog daemon in CentOS and Debian.

The server needs to run on either UDP or TCP, which can be configured in the /etc/rsyslogd.conf file.

After that, the standard options sent to the daemon will have to be edited as well. These files reside in different folders for respectively CentOS (/etc/sysconfig) and Debian (/etc/defaults).

Additions I made to the default configuration:

```
/etc/sysconfig/rsyslogd | /etc/defaults/rsyslogd

#IPv6:
RSYSLOGD_OPTIONS="-6"
```

Make sure your ip6tables do not block port 514, then restart the rsyslogd using the /etc/init.d/rsyslogd script.

Ironically enough rsyslogd is also the program that can write to another server. When configured correctly rsyslogd can forward all messages to another syslog daemon. This can be configured in the /etc/rsyslogd.conf file.

Additions I made to the default configuration

```
#IPv6:
*.* @[2001:db8::1]:514
```

Where *.* means all events, and @ means UDP. For a TCP session, @@ instead of @ should be used.

Make sure your ip6tables do not block UDP port 514. Afterwards a restart of the daemon will put the new changes in action. The daemon can be started and stopped using the /etc/init.d/rsyslogd script on CentOS and Debian.

6.7.2. Windows Server 2008 R2 Syslog

There are two major commercial competitors for Syslog on the Windows platform that support IPv6, yet there is a (very basic) freeware server available as well.

6.7.2.1. Syslog Watcher

Snmpsoft's Syslog Watcher is fully IPv6 supported. The current version (4.3.0) is only a Syslog server, but in the next update (4.5.0) the Syslog forwarding now done by Eventlog Inspector will be integrated into Syslog watcher. Syslog Watcher can be configured to listen to IPv4 and IPv6, or IPv6 only. After the initial configuration, this piece of software does work as advertised.

6.7.2.2. WinSyslog

WinSyslog supports IPv6 as of version 11.0, yet it does not support the forwarding of Syslog messages to IPv6-only syslog hosts. WinSyslog can be configured to listen on IPv6 instead of IPv4. You can also select which RFC's WinSyslog will be able to parse, and change the listening address (all interfaces, or just one specific interface).

6.7.2.3. Tftpd32 and 64

Tftpd32 and 64 is a freeware DHCP, TFTP, DNS, SMTP and Syslog server, as well as a TFTP client. The default Syslog service of this client listens on UDP port 514, on IPv4 and IPv6. This piece of software can come in handy in a Windows network, with one or two Linux servers that support Syslog.

Sources

General

http://tldp.org/HOWTO/Linux+IPv6-HOWTO/

https://wikispaces.psu.edu/display/ipv6/Home

https://confluence.terena.org/display/~visser/Office+network+transition+to+IPv6+only

main pages of software packages

NDP

https://en.wikipedia.org/wiki/Neighbor_Discovery_Protocol

https://tools.ietf.org/html/rfc4861

https://en.wikipedia.org/wiki/Radvd

http://www.litech.org/radvd/

http://tools.ietf.org/html/rfc6106

http://www.remlab.net/ndisc6/

DHCPv6

https://en.wikipedia.org/wiki/DHCPv6

https://tools.ietf.org/html/rfc3319

https://www.isc.org/software/dhcp/

http://wide-dhcpv6.sourceforge.net/

http://klub.com.pl/dhcpv6/

http://ipv6int.net/software

http://technet.microsoft.com/en-us/library/cc732584.aspx

DNS

https://en.wikipedia.org/wiki/Domain_Name_System

http://tools.ietf.org/html/rfc4343

https://www.isc.org/software/bind/

http://technet.microsoft.com/en-us/library/cc730921.aspx

SSH

https://en.wikipedia.org/wiki/Secure_shell

https://en.wikipedia.org/wiki/Secure_copy

http://tools.ietf.org/html/rfc4253

http://www.openssh.org/

NTP

https://en.wikipedia.org/wiki/Network_Time_Protocol

http://tools.ietf.org/html/rfc5905

http://www.ntp.org/

SNMP

https://en.wikipedia.org/wiki/Snmp

http://tools.ietf.org/html/rfc2711

http://www.net-snmp.org/

http://technet.microsoft.com/en-us/library/bb726987.aspx

SMB

https://en.wikipedia.org/wiki/Cifs

https://www.samba.org/

http://technet.microsoft.com/en-us/library/cc734393.aspx

NFS

https://en.wikipedia.org/wiki/Network_File_System

http://tools.ietf.org/html/rfc3530

http://support.microsoft.com/kb/324089

http://support.microsoft.com/kb/324086

Syslog

https://en.wikipedia.org/wiki/Syslog

http://tools.ietf.org/html/rfc5424

http://www.rsyslog.com/

http://www.snmpsoft.com/syslogwatcher/syslog-server.html

http://winsyslog.com/en/

http://tftpd32.jounin.net/