IPv6 Security

M Babik[1], J Chudoba[2], A Dewhurst[3], T Finnern[4], T Froy[5], C Grigoras[1], K Hafeez[3], B Hoeft[6], T Idiculla[3], D P Kelsey[3], F López Muñoz[7,8], E Martelli[1], R Nandakumar[3], K Ohrenberg[4], F Prelz[9], D Rand[10], A Sciabà[1], U Tigerstedt[11], D Traynor[5] and R Wartel[1]

[1] European Organization for Nuclear Research (CERN), CH-1211 Geneva 23, Switzerland
[2] Institute of Physics, Academy of Sciences of the Czech Republic Na Slovance 2 182 21 Prague 8, Czech Republic
[3] STFC Rutherford Appleton Laboratory, Harwell Campus, Didcot, Oxfordshire OX11 0QX, United Kingdom
[4] Deutsches Elektronen-Synchrotron DESY, Notkestraße 85, D-22607 Hamburg, Germany
[5] Queen Mary University of London, Mile End Road, London E1 4NS, United Kingdom
[6] Karlsruher Institut für Technologie, Hermann-von-Helmholtz-Platz 1, D-76344 Eggenstein-Leopoldshafen, Germany
[7] Port d'Informació Científica, Campus UAB, Edifici D, E-08193 Bellaterra, Spain
[8] Also Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas (CIEMAT), Madrid, Spain
[9] INFN, Sezione di Milano, via G. Celoria 16, I-20133 Milano, Italy
[10] Imperial College London, South Kensington Campus, London SW7 2AZ, United Kingdom
[11] CSC Tieteen Tietotekniikan Keskus Oy, P.O. Box 405, FI-02101 Espoo

E-mail: `david.kelsey@stfc.ac.uk, ipv6@hepix.org`

Abstract. IPv4 network addresses are running out and the deployment of IPv6 networking in many places is now well underway. Following the work of the HEPiX IPv6 Working Group, a growing number of sites in the Worldwide Large Hadron Collider Computing Grid (WLCG) are deploying dual-stack IPv6/IPv4 services. The aim of this is to support the use of IPv6-only clients, i.e. worker nodes, virtual machines or containers.

The IPv6 networking protocols while they do contain features aimed at improving security also bring new challenges for operational IT security. The lack of maturity of IPv6 implementations together with the increased complexity of some of the protocol standards raise many new issues for operational security teams.

The HEPiX IPv6 Working Group is producing guidance on best practices in this area. This paper considers some of the security concerns for WLCG in an IPv6 world and presents the HEPiX IPv6 working group guidance for the system administrators who manage IT services on the WLCG distributed infrastructure, for their related site security and networking teams and for developers and software engineers working on WLCG applications.

1. Introduction

The much-heralded exhaustion of the IPv4 networking address space is with us. The HEPiX IPv6 Working Group [1] has been investigating the many issues feeding into the deployment of IPv6 in HEP in general and more specifically in WLCG. The group's paper at CHEP2015 [2] presented the testing and deployment of dual-stack data storage services with the aim of soon being able to support the use of IPv6-only CPU. Since then, WLCG has approved our plan to support such use of IPv6-only CPU from April 2017 [3].

One of the important concerns for this migration to IPv6 relates to operational security. While the IPv6 networking protocols do contain features aimed at improving security they also bring new challenges. We have spent many decades understanding and fixing security problems and concerns in the IPv4 world. Many WLCG site support teams have only just started to consider IPv6 security and they are far from ready to be able to follow best practice.

There is much information available on IPv6 security but the fact that there are so many documents on the topic does not make it easy for WLCG system administrators, hereafter abbreviated to "sysadmins", to digest and identify the key issues. This paper is not competing with the other information but gives pointers to these other books and papers. The IPv6 working group has decided to produce and maintain two short checklists of the key IPv6 security issues to be addressed as a starting point for WLCG sysadmins, for their site networking and security teams and also for WLCG/HEP application developers and software engineers. This is based on the experience of those sites active in the HEPiX IPv6 working group.

We have found the following to be useful sources of information on IPv6 Security; they contain much more background information and fuller exploration of the details:

(i) A large and complete text-book from Cisco Press studying the whole subject matter [4].

(ii) NIST SP 800-119, Guidelines for the Secure Deployment of IPv6. This is a shorter but still complete study of the topic [5].

(iii) "10 myths of IPv6 security from the Internet society"; all interesting and amusing! [6].

(iv) A SANS white paper on IPv6 attack and defense [7].

(v) An ERNW hardening guide for Linux servers; the best guidance we have found so far on IPv6 and Linux Systems [8].

(vi) There are many IETF RFC documents on IPv6 and IPv6 security. In particular, we recommend "Enterprise IPv6 Deployment Guidelines", RFC7381 [9].

The checklists we have produced are the IPv6 working group's current list of issues to be considered. They are maintained on the group web site [1]. We welcome feedback from sites and developers on the contents of these lists according to their experiences during the transition. Updates and additions will be made as required.

This paper is organised as follows. Section 2 presents a brief introduction to some of the potential vulnerabilities and concerns related to IPv6. Section 3 contains the checklist for sysadmins and site networking/security teams. Section 4 presents our checklist aimed at application developers. The paper ends with a short summary in Section 5.

2. Some IPv6 security issues

The introduction of any new protocol can of course introduce new security problems. For IPv6, these include the lack of maturity of the implementations which therefore are highly likely to still contain many new vulnerabilities. The need for IPv6-compliant monitoring and tools and the lack of education and experience of sysadmins both also cause concern. There are problems introduced by the need to support a transition process, including the complications arising from the use of dual-stack protocols and/or transition tunnels.

One of the potential advantages claimed by supporters of IPv6 was that security was explicitly addressed in the design. The mandatory use of IPsec (see RFC4301 [9]), for example, seemed to be an early success but problems related to key management led to the mandatory support of IPsec being relaxed (see section 11 of RFC6434 [9]).

Examples of some other new features resulting in security concerns include:

(i) **Many more ICMP message types**. It is not possible to filter all of them, e.g. Path Maximum Transmission Unit (MTU) Discovery has to work, but sites are recommended to filter some other types of ICMPv6. RFC4890 [9] gives advice.

(ii) **New methods for autoconfiguring addresses, and the locations of routers and DNS servers**. This is good news for the end-systems but networking teams must now protect against potential attacks, for example rogue Router Advertisements (see RFC6104 [9]).

(iii) **Longer IP addresses**. On first consideration, this appears to make brute force scans of end-systems much more difficult but that is not always true as there is often structure to the use of the address space.

(iv) **IPv6 does not allow packet fragmentation en-route**. The minimum MTU supported is 1280 bytes, but you can still hurt yourself by sending smaller fragments if you wish.

(v) Not really a feature of IPv6 proper, but much of the **network stack and application code is not yet hardened** and therefore is potentially vulnerable to attack.

(vi) **Transitional technologies** (e.g. tunnels) have intrinsic vulnerabilities but don't need to be there forever.

The bad news is that many previous security issues from the IPv4-era have also not changed. Many earlier attacks are still possible:

- Broadcasts and Multicasts are still there, with a vengeance;
- Can still use IP headers for out-of-band communications;
- Can still pollute Ethernet address discovery (Neighbor Discovery instead of Address Resolution Protocol);
- Can still run a rogue DHCP server;
- Can still try forging and injecting packets into the local network;
- The upper-layer protocols did not change.

3. Checklist for WLCG site system administrators and networking teams

This section contains the IPv6 security checklist for sysadmins and site networking/security teams. This presents the key IPv6 security issues to be addressed by WLCG sites as a starting point. More information is available in the documents referenced here or in the Introduction.

(i) **Make an addressing plan**
One of the most important design decisions for a site team creating an IPv6 deployment plan is to create an IPv6 addressing plan. This should match the acceptable usage and access policy of the organisation. It also needs to include consideration as to how to manage a dual-stack network, possibly reviewing the security aspects of the existing IPv4 infrastructure. IPv6 address space (typically a /48 - default size as in RFC3177 [9]) will have been allocated to the site by its NREN or other ISP. Consideration needs to be given to the routing and switching design of the network [10]. The number of subnets, the routing architecture, and the address allocation within subnets etc. all need to be included.

(ii) **Decide whether to use DHCPv6 or SLAAC (+dDNS)**
The second most important decision - and very much linked to the one above - to be made by a site networking team is whether or not to use one of the important new features of IPv6, i.e. the end-system use of IPv6 Stateless Address Autoconfiguration (see RFC4862 [9]). You may prefer that server systems have fixed IPv6 addresses (either manual or DHCPv6), while the use of SLAAC for servers should be accompanied by dynamic DNS. Looking at these three protocols (SLAAC, dDNS and DHCPv6) from a security standpoint

(see RFC4942 [9]), the main difference in available spoofing/hijacking channels is due to the fact that configuration via the ICMPv6 Router Advertisement protocol is multicast to the network segment and received by hosts.

(iii) **Ensure all security/network monitoring/logging tools are IPv6-capable**
All monitoring and logging tools (commercial, open-source, home written), at any hierarchical level of the organisation (central and end-user), need to be evaluated and tested for operation on IPv6. They should properly deal with longer addresses, the new address syntax, multiple addresses per network card, etc. Tools should be certified to work in a dual-stack environment, with the ability to simultaneously monitor both stacks. Tools analysing log files should be capable of parsing address syntax for both stacks.

(iv) **Filter IPv6 packets that enter and leave your network/system**
IPv4-only networks include end systems where IPv6 is enabled by default. This may open significant security breach opportunities if intrusion detection systems and Firewalls are not handling IPv6 traffic correctly. Consideration should be given to the various options of setting up system- or network-level filtering. See also the following items for hints on the performance implication of IPv6 filtering and the need to keep these measures synchronised with the IPv4 ones. Switching off/filtering IPv6 at the network level isn't a realistic option anymore, while specific legacy systems may need to be protected by disabling their IPv6 stack. The specific issue of filtering of ICMPv6 packets is dealt with in the next topic.

(v) **Filter ICMPv6 messages wisely**
Many ICMPv6 messages have an essential role in establishing or maintaining IPv6 communication. Some ICMPv6 messages can therefore not be blocked (unlike in IPv4). Other types of ICMPv6 messages may lead to security issues if allowed through the site border. See RFC4890 [9] for a full discussion and detailed advice. Our group's knowledge base [1] offers RFC4890-compliant example rules for IOS and JunOS devices.

(vi) **Allow special-purpose IPv6 Extension Headers only if needed**
Although the requirement for IPv6 stacks to fully implement the handling of *all* IPv6 Extension Headers (for IPSEC, Mobile IP, etc.) has been lifted (per RFC5095 [9]), the processing of these may open up end systems and sites to a large vulnerability space. Packets with inappropriate, unexpected or malformed Extension Headers should be filtered. The processing of Extension Headers can significantly complicate the task of firewall ACLs, especially if headers extend beyond the first packet fragment. Sites need to verify whether running ACLs have the ability to process Extension Headers and block unwanted extension headers as prescribed by RFC7045 [9].

(vii) **Use synchronised IPv4/IPv6 access rules**
The management of production dual-stack networks, especially when reacting to security incidents, is much simpler if semantically equivalent firewall rules (site and end-system) are deployed for both stacks. Having to always update lists in tandem is error-prone and may leave unnoticed security holes (or unwanted denial of service) behind. Lacking the capability of networking firmware to express rules in common lists, one may think of generating such lists from a common template. The fact that there are IPv6-specific measures to take (see e.g. the ICMPv6 point above) may lead sites to assemble entirely new configurations. These should never forget *any* of the existing IPv4 rules. The option between modeling IPv6 lists on existing IPv4 lists or taking the IPv6 rollout as a chance to refresh both lists should be evaluated.

(viii) **Deploy RA-Guard or otherwise deal with Rogue RAs**
Neighbor Discovery and Router discovery are two important new features of IPv6 (actually ICMPv6). The fact that they are generally based on IPv6 multicast exposes sites to several different security problems. Rogue routers or attackers can send out false router

announcements (RAs) to persuade end systems to send packets to them for routing allowing for lack of privacy and man in the middle attacks. RFC6104 describes the issue in detail and covers several ways of addressing the issue. The only sure-fire measure is to apply a filter on all local area network (LAN) active parts, e.g. by having them accept RAs only from a single endpoint of the network, via the RA-Guard (RFC6105) protocol. Making sure all of the network infrastructure supports RA-Guard may take a significant amount of effort (or equipment replacement), so the other measures described in RFC6104 may need to be deployed in the meantime.

(ix) **Do not be tempted by transition technologies**
Think carefully before using or allowing tunnelling technologies as an IPv4→IPv6 transition tool. These usually do not save IPv4 address space. Their additional complexity doesn't gain any real advantage over the simpler approach of deploying dual-stack systems and offers a wide space for security exploits. NAT64/DNS64 (RFC6146/6147 [9]) may become a useful tool when the transition process is almost complete and the burden of managing IPv4 on the LAN can be eventually removed.

(x) **Filter/disable IPv6-on-IPv4 tunnels**
As we don't recommend using such tunnels (see above), the IPv4 protocol number 41 (IPv6 encapsulation) should be disabled and filtered throughout the site network.

4. Checklist for developers

When applications developed in the golden era of IPv4-only Internet face the transition to IPv6, the brunt of the work often falls on the shoulders of developers who belong to a different generation from the original authors. Figure 1 tries to visualise the extent of the changes that the core code of any IP-capable application undergoes in the transition. In addition to the extensively different syntax, there is a fundamental $1 \to N$ change here: no IP endpoint can be satisfied with handling just *one* IP address (as any public IPv6 endpoint communicates via the public and on the link-local address at least), but loops and address ordering start appearing everywhere. We identify the following implications of this fundamental fact on developers' practice, in roughly descending order of importance:

(i) **Plan for extensive testing.**
The *syntactic* change of the core IP networking code in the IPv4→IPv6 transition is large enough to oftentime justify the refactoring of larger portions of code. The *semantic* $1 \to N$ change may be *forcing* some rethinking at the design level. A possible temptation here is to provide parallel sections of code that handle the IPv6 case only: a few other reasons why this may not be a good idea are listed below. In any case, there is an implicit expectation that a change that should be affecting the *transport* layer of the network only should cause no ripple in the upper layers, i.e. that the perceived responsiveness, performance and reliability of the code remain unchanged. *Extensive* stress-testing should therefore be planned on IPv6-ported code.

(ii) **Respect the sysadmin protocol preferences.**
Code that binds and connects IP sockets is suddenly faced with making choices that used to be delegated to the operating system or networking-capable libraries. Lists of addresses may be received in a given order, but it's now the responsibility of the socket-handling code to iterate and re-iterate on the list, handle exceptions and possibly operate in parallel on various entries to implement some form of 'happy-eyeballs'[1] algorithm. As the ordering of both source and destination addresses established at the system level by the system administrator[2] may have security implications, developers should go the extra mile to keep

[1] See RFC6555 [9].

[2] Via `/etc/gai.conf`, `ip addrlabel` or their equivalent.

```
struct hostent *resolved_name=NULL;
struct servent *resolved_serv=NULL;
struct protoent *resolved_proto=NULL;
static char *dest_host="some.ip.host", *dest_serv="ipservice";
struct sockaddr_in destination;
resolved_host = gethostbyname(dest_host);
resolved_serv = getservbyname(dest_serv, NULL);
if (resolved_host != NULL && resolved_serv != NULL) {
     destination.sin_family = resolved_name->h_addrtype;
     destination.sin_port = htons(resolved_serv->s_port);
     memcpy(&destination.sin_addr, resolved_host->h_addr_list[0],
          resolved_host->h_length);
     resolved_proto = getprotobyname(resolved_serv->s_proto)
     if (resolved_proto != NULL) {
          int fd = socket(AF_INET, SOCK_STREAM, resolved_proto->p_proto);
          connect(fd, &destination, sizeof(destination));
          /* Check for errors, connect, etc...  */
     }
}
```

```
struct addrinfo ai_req, *ai_ans, *cur_ans;
static char *dest_host="some.ip.host", *dest_serv="ipservice";
ai_req.ai_flags = 0;
ai_req.ai_family = PF_UNSPEC;
ai_req.ai_socktype = SOCK_STREAM;
ai_req.ai_protocol = 0; /* Any protocol is OK */
if (getaddrinfo(dest_host, dest_serv, &ai_req, &ai_ans) != 0) {
     for (cur_ans = ai_ans; cur_ans != NULL; cur_ans = cur_ans->ai_next) {
          int fd = socket(cur_ans->ai_family, cur_ans->ai_socktype,
               cur_ans->ai_protocol);
          connect(fd_socket,cur_ans->ai_addr,cur_ans->ai_addrlen);
          /* Check for errors - This loop has the ability to change the */
          /* order of the getaddrinfo results!  */
     }
}
```

Figure 1. C code snippets showing how the basic IP service resolution and connection changes from legacy IPv4-only to a dual-stack or IPv6-only environment. This represents the zeroth-order porting effort for much IPv4-only code. The newer structure is more terse, but the changes are extensive enough, both syntactically and semantically, to probably trigger the refactoring of much larger sections of code.

that ordering even if they have to reshuffle the list for any reason. Applications should allow users to prefer/enable either IPv4 or IPv6 via configuration, but should always honor the system-level administrator's choice by default.

(iii) **Port all existing security measures.**
Fresh new code that hasn't been tested broadly and in the wild is *per se* attractive to anyone looking for malicious exploits. Especially in the case where IPv6-specific code or processes are developed for *parallel* deployment with well-proven IPv4 code, one should make sure that any security measure, filter or wisdom that was included in the code for the IPv4 case isn't simply forgotten for IPv6. While it may not be immediately apparent, *all* constructs that are meaningful for IPv4 have their translation or counterpart for IPv6.

5. Summary

In this paper, we have discussed some of the security concerns that arise from the deployment of the IPv6 networking protocols. In particular, we have presented our IPv6 security checklist for sysadmins and site networking/security teams at WLCG sites. We have also presented a checklist for WLCG/HEP application developers and software engineers. We welcome feedback from sites and developers on the contents of these lists according to their experiences during the transition. HEP IT support staff have spent many decades understanding and fixing security problems and concerns in the IPv4 world. We confidently predict that securing IPv6 will take an equally long time!

Acknowledgements

The authors acknowledge the contributions to this work made by former members of the HEPiX IPv6 Working Group and other colleagues within WLCG. In particular we express our thanks to Raul Lopes and Simon Furber at Brunel University London for their efforts in the configuration and testing of dual-stack DPM and Argus from IPv6-only compute nodes. We also thank Edgar Fajardo Hernandez from the University of California, San Diego for his work on the validation of IPv6 and glideinWMS for the CMS experiment. The authors also acknowledge the support and collaboration of many other colleagues in their respective institutes, experiments and IT Infrastructures, together with the funding received by these from many different sources. These include but are not limited to the following:

(i) The Worldwide LHC Computing Grid (WLCG) project is a global collaboration of more than 170 computing centres in 42 countries, linking up national and international grid infrastructures. Funding is acknowledged from many national funding bodies and we acknowledge the support of several operational infrastructures including EGI, OSG and NDGF/NeIC.

(ii) EGI acknowledges the funding and support received from the European Commission and the many National Grid Initiatives and other members. The EGI-Engage project is co-funded by the European Commission (grant number 654142).

References

[1] The HEPiX IPv6 Working Group web site is to be found at http://hepix-ipv6.web.cern.ch

[2] Bernier J *et al* 2015 The production deployment of IPv6 on WLCG *J. Phys.: Conf. Ser.* **664** 052018

[3] Babik M *et al* Deployment of IPv6-only CPU resources at WLCG sites - submitted to this conference CHEP2016 - *Journal of Physics: Conference Series*

[4] Hogg S, Vyncke E - IPv6 Security, Cisco Press 2009, ISBN-13: 978-1-58705-594-2

[5] Frankel S *et al* 2010 Guidelines for the secure deployment of IPv6 - NIST Special Publication 800-119 available at http://dx.doi.org/10.6028/NIST.SP.800-119

[6] Ten myths of IPv6 security - Internet Society available at http://www.internetsociety.org/deploy360/blog/category/ipv6/security/

[7] Pilihanto A 2011 - A complete guide on IPv6 attack and defense - available at https://www.sans.org/reading-room/whitepapers/detection/complete-guide-ipv6-attack-defense-33904

[8] Atlasis A 2014 - IPv6 hardening guide for Linux servers available at https://www.ernw.de/download/ERNW_Guide_to_Securely_Configure_Linux_Servers_For_IPv6_v1_0.pdf

[9] All Internet Engineering Task Force Requests For Comments (RFC) documents are available from URLs such as http://www.ietf.org/rfc/rfcNNNN.txt where NNNN is the RFC number, for example http://www.ietf.org/rfc/rfc2460.txt

[10] There is abundant reference material at http://www.internetsociety.org/deploy360/resources/ipv6-address-planning-guidelines-for-ipv6-address-allocation/ and https://www.ripe.net/support/training/material/IPv6-for-LIRs-Training-Course/Preparing-an-IPv6-Addressing-Plan.pdf.