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], D P Kelsey[3], F López Muñoz[7], E Martelli[1], R Nandakumar[3], K Ohrenberg[4], F Prelz[8], D Rand[9], A Sciabà[1], D Traynor[5], U Tigerstedt[10] and R Wartel[1]

[1] CERN, CH-1211 Genève 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, 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] INFN, Sezione di Milano, via G. Celoria 16, I-20133 Milano, Italy
[9] Imperial College London, South Kensington Campus, London SW7 2AZ, United Kingdom
[10] 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) have deployed 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. We have spent many decades understanding and fixing security problems and concerns in the IPv4 world. Many WLCG IT support teams have only just started to consider IPv6 security and they are far from ready to follow best practice, the guidance for which is not easy to find. The lack of maturity of IPv6 implementations together with the increased complexity of the protocol standards and the fact that the new protocol stack allows for pretty much the same attack vectors as IPv4, raise many new issues for operational security teams.

The HEPiX IPv6 Working Group is producing guidance on best practices in this area. This paper will consider some of the security concerns for WLCG in an IPv6 world and present the HEPiX IPv6 working group guidance both for the system administrators who manage IT services on the WLCG distributed infrastructure and also for their related security and networking teams.

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 move to the use of IPv6 in HEP in general and more specifically in WLCG. The group's paper at CHEP2015

[**?**] 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 now has an agreed plan to support such use of IPv6-only CPU from April 2017 (see other paper submitted to this conference).

One of the important concerns for this migration to IPv6 relates to operational security. The IPv6 networking protocols while they do contain features aimed at improving security also bring new challenges for operational security. 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 acts as pointers to these other books and papers. The IPv6 working group has decided to produce and maintain two short checklist of the key IPv6 security issues to be addressed as a starting point for both sysadmins and site network teams and also for WLCG/HEP application developers and software engineers. This is based on the experience of a few sites active in the HEPiX IPv6 working group. We have found the following to be useful sources of fuller information on IPv6 Security. These contain much more background information and fuller exploration of the details.

a) The Cisco book. A large and complete study of the whole subject matter. b) NIST 800-119 "title". A shorter but still complete study providing guidance. c) 10 myths (Internet Society) - interesting and amusing! d) "Johanna" paper e) SANS guidance f) ERNW guidance - best guidance we have found so far on IPv6 and Linux Systems. g) Lots of IETF RFC documents - which ones do we include?

The checklists are the IPv6 working group's current list of issues to be addressed. The checklists will be maintained on the group web site (Ref). We welcome feedback from sites and developers on the lists according to their experiences during the transition. Updates and additions will be made as required.

This paper is organised as follows. Section 2 will present 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.

2. IPv6 security issues

IPv6 Security issues

Paper starting with extension headers, fragmentation and other native header fields. Subsequently, Neighbor and Multicast Listener Discovery are discussed, followed by tunneling and mobility support.

New features (from the poster)

Many more ICMP message types! Cannot filter all of them (MTU discovery has to work). Must filter some of them. RFC4890 gives advice

New methods for autoconfiguring addresses, routes, DNS. Good for the end-user. Must do something against rogue Router Advertisements (see RFC6104)

Longer IP addresses. Hey, everyone knows that. They may slow down brute force scans. But no bad guy is that crude...

Cannot fragment packets en-route. Minimum MTU: 1280. But you can still hurt yourself and send small fragments if you wish. Some good news, at least

Not really a feature of IPv6 proper, but much of the network stack and application code is enticingly fresh!

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

Business as usual (from the poster)

As long as all network monitoring and administration tools are up-to-date and (therefore) aware of IPv6.

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 (ND instead of ARP). Can still run a rogue DHCP server. Can still try forging and injecting packets into the local network. Upper-layer protocols did not change!

Advantages of a new design Security: important part of the IPv6 initial design Down-sides Lack of maturity New vulnerabilities and attack vectors Need IPv6-compliant monitoring and tools Lack of education and experience Problems of transition ? dual-stack, tunnels BUT - Many threats/attacks happen at layers above/below the network layer And are therefore exactly the same as in IPv4 Malware, phishing, buffer overflows, cross-site scripting, DDoS etc etc

3. Checklist for system administrators

(i) Make an addressing plan One of the most important design decisions for a site networking team doing a deployment plan is to create a well thought out management plan for their IPv6 address space (linked to next topic too). Needs to include thoughts as to how to manage a dual-stack network. Address space (typically a /48 - default RFC3177) will have been allocated to the site by its NREN or other ISP. How many subnets? Routing architecture, address allocation within subnets etc. At the very least this should include .... See 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

(ii) Decide whether to use DHCPv6 or SLAAC+DynDNS 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). And use of dynamic DNS. Server systems may want to have fixed addresses (either manual or DHCPv6).

(iii) Ensure all security/network monitoring/logging are IPv6-capable Important for networking teams, security teams and also end systems tools for sysadmins. All monitoring and logging tools (commercial, open-source, home written) need to be evaluated and tested for operation on IPv6. New longer addresses and multiple addresses per network card. Do they work in a dual-stack environment and can they simultaneously monitor both stacks. What about tools analysing log files - does parsing work?

(iv) Filter IPv6 packets that enter and leave your network/system IPv4-only networks will have end systems where IPv6 is enabled by default. May cause big security problems with IDS and Firewalls not handling IPv6 traffic correctly. If you don't want IPv6 best to turn it off and/or filter it at both the network and system level.
Filter packets with Extention headers.
For filtering of ICMPv6 packets - see next topic.

(v) Filter ICMPv6 messages wisely An important feature of IPv6. needed for path packet size determination and for ... Some things can therefore not be blocked (unlike in IPv4).

(vi) Allow special-purpose headers only if needed Extension Headers open up to all sorts of vulnerabilities - see RFC

(vii) Use synchronised IPv4/v6 access rules For dual-stack networks MUCH much better to have identical firewall rules (site and end-system) for both stacks. Making them different can cause problems with

(viii) Deploy RA-Guard or otherwise deal with Rogue RAs Neigbor Discovery and Router discovery are two important new features of IPv6. Opens up to several different security

problems. Rogue routers 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. Several ways of addressing this

(ix) Do not be tempted by transition technologies By this we mean think very carefully before deciding to use or allow the use of tunnelling technologies. Dual-stack systems are the best approach. NAT64 is being used by some WLCG sites but we do not in general recommend this unless ... problems with tunnels and protocol translations are ...

(x) Filter/disable IPv6-on-IPv4 tunnels We recommend not using such tunnels (see above). So we suggest that sites should filter or disable these - done as follows?

From ISGC2016 talk:

Control IPv6 if not using it Use Dual-stack and avoid use of tunnels wherever possible Drop packets containing RH Type 0 and unknown option headers Deny packets that do not follow rules for extension headers Filter IPv6 packets that enter and leave your network Restrict who can send messages to multicast group addresses Create an Address management plan Create a Security Policy for IPv6 (same as IPv4) Block unnecessary ICMPv6 Protect against LAN RA, ND and DHCP attacks NDPMON and RAFIXD on critical segments Check/modify all security monitoring, logging and parsing tools

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 often belong to a different generation as 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 $1 \rightarrow N$ 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 rough 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* $1 \rightarrow N$ change may be *forcing* some rethiking 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] alghorithm. 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 [2].

[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.

References

[1] `http://hepix-ipv6.web.cern.ch`

[2] 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`

[3] See for instance `http://www.google.com/ipv6/statistics.html`. The 10% global connectivity threshold was crossed in January 2016.

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