UPnP security in Internet gateway devices

Kristian Selén
Helsinki University of Technology
kselen@cc.hut.fi

Abstract

Current Internet gateway devices, such as broadband routers and firewalls, are frequently equipped with a *Universal Plug and Play* compliant daemon. In many cases, implementations of the protocol have been proven flawed. Also, the operating principle of UPnP assumes a certain level of security in the network environment. In practice, this is usually not achieved. As this service is often enabled by default, several security implications arise.

This study focuses on security problems associated with using the standardized *Device Control Protocol (DCP) V 1.0* for *Internet Gateway Device*s (IGD), as defined by UPnP.[17]

KEYWORDS: UPnP, IGD, security, SSDP

1 Introduction

Internet gateway devices, sometimes referred to as *broadband routers* by various manufacturers, are often used for protection in home networks accessing the Internet. These devices prevent potential attackers on the Internet from accessing hosts connected to the local network. In many cases, however, it is necessary for applications or services on the protected network to be visible to the Internet. Sometimes, a skilled administrator can configure the gateway device to forward selected traffic to the appropriate host. Often, this is not an option, and consequently there is a need for applications and services themselves to be able to direct the flow of traffic from the Internet.

Universal Plug and Play, a technology driven forward by the UPnP Forum, offers a broad range of solutions for various challenges in mobile or traditional IP networks, including service discovery and autoconfiguration. Certain aspects of UPnP, which are associated with IP traffic direction and filtering, are already implemented in many gateway devices.

In this paper, we will try to make a realistic assessment of the major security aspects involved in this arrangement.

2 Background

Users accessing the Internet have different needs. Some use complex applications while others just surf the web. Regardless of the underlying technology, most people want solutions that both work as intended and are easy to use.

2.1 Home networking

A steadily increasing number of home networks is connecting to the Internet. It is no longer correct to assume that only highly skilled professionals will be installing routers and other networking equipment.

Also, there is a broader scale of different devices connecting through these networks. Mobile phones, laptops, televisions and other consumer electronics are connecting both by wire and through various wireless links.

2.2 Protocols and firewalls

Most firewalls on the home consumer market offer protection by performing network address translation (NAT) on the IP traffic to and from the Internet. This process ultimately hides the local network from hosts on the Internet, making all outgoing communication appear as if it was originating from the gateway. Without special arrangements, this also makes it impossible for Internet hosts to directly access hosts on the protected network.[6]

Many protocols and applications today are designed to be fully, or at least partially, functional in environments employing NAT devices. Many, however, are not, and therefore require port forwardings or other configurations in the firewall.

For example, popular peer-to-peer (P2P) systems, such as Skype or BitTorrent, can often be used in a firewalled NAT environment. The operating efficiency of these applications, however, will usually greatly benefit from full IP connectivity.

2.3 Autoconfiguration

In some cases, there is a need for allowing applications to discover and configure firewall devices themselves. This may be due to the fact that the users of the application do not have the technical skills needed for manual configuration, or it is simply not possible to complete the configuration in advance, for example in environments with very large networks or complex protocols.

In most cases, it is also desirable to keep, for instance, port forwardings open only while they are actually in use by an application. Additionally, some protocols use dynamically changing ports, which cannot be preconfigured into the firewall.

3 UPnP architecture

Universal Plug and Play includes a broad range of specifications for communication and service discovery for differ-

ent types of devices in various environments.[22] Here, we will focus on the specific techniques required for autoconfiguring the firewall on consumer market broadband routers, foremostly the specifications for the Internet Gateway Device and other associated protocols.

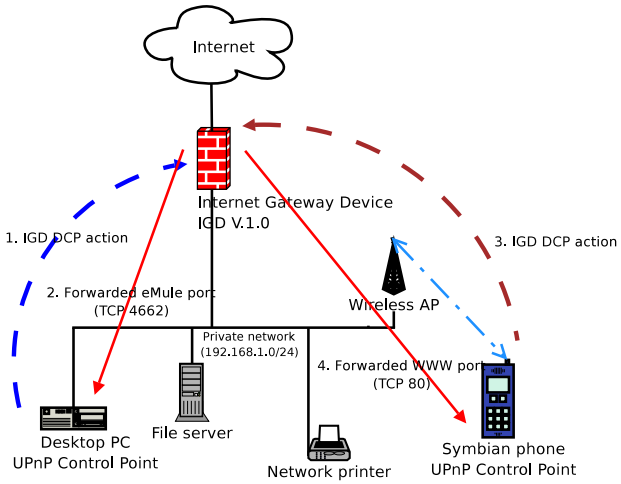

Figure 1: A UPnP enabled network

A possible usage scenario involving Universal Plug and Play -aware network nodes is depicted in figure 1. Here, a gateway device, conforming to the UPnP IGD V1.0 standard, is protecting a small private network with various nodes. The WAN interface of the gateway is connected to the Internet using some available technology, like Ethernet or xDSL. The LAN interface is connected to the local Ethernetwork.

In this case, a normal desktop PC is running an instance of eMule [11]. Once this peer-to-peer network client has located the gateway, it is then able to command (1) the gateway to open up the TCP port for incoming connections. If the action is successful, other external nodes of the P2P network can then contact the client (2).

In a similar fashion, a mobile phone connecting to the local network using a wireless link, could command (3) the gateway device to expose some web application running on the phone. This would enable users on the Internet to access web resources on that device (4). UPnP clients may be referred to as Control Points (CPs).

3.1 Operation

When a UPnP client (Control Point) enters a network, it sends out a broadcast message to locate other UPnP devices, who then respond. Once they are located, the client can retrieve an XML description of the nodes. The descriptions contain information about the properties the devices possess, mainly what state variables are defined and what actions are supported.

UPnP also defines eventing, which allows devices to inform CPs of changes in their state.

A simple example of a message exchange between a CP and an Internet Gateway Device is presented in figure 2. [13]

Here, the action might involve adding a port forwarding using *addPortMapping*. The event message might indicate that the WAN IP address of the gateway has changed as a result of a DHCP renewal process.

Note that clients request services when they need them, but devices also broadcast presence information upon entering the network.

3.2 Service model

Universal Plug and Play defines service models for different types of devices. These definitions include service types, state variables, event handling and supported actions. These properties are defined in detail in corresponding service templates, which also include a precise XML service description.

The UPnP service templates form a hierarchy, where subcomponents or -devices of root (main) devices are defined in separate service templates. Supporting certain service templates in a UPnP device is mandatory, while other templates may be optional. Together, these service templates specify the Device Control Protocol (DCP) for the device.[17]

3.3 Internet Gateway Device

The UPnP root device template for the Internet Gateway Device (IGD) specifies that *The Internet Gateway is an "edge" interconnect device between a residential Local Area Network (LAN) and the Wide Area Network(WAN), providing connectivity to the Internet.*[17]

The IGD templates include numerous sub-devices and services. For port forwarding, the *AddPortMapping* action, defined in the *WANIPConnection* service template, can be used. Both the service template and the action are required in UPnP implementations of the Internet Gateway Device.[18]

The *AddPortMapping* -action makes it possible to configure the gateway to forward IP packets destined to a specific port on the gateway's external interface to a (possibly different) port on a host on the internal network. [18]Effectively, this exposes any service running on that particular port on the internal host to the Internet.

The service templates of the IGD root device and its subdevices specify a large number of actions and state variables, that may also have implications on security. Analyzing these,

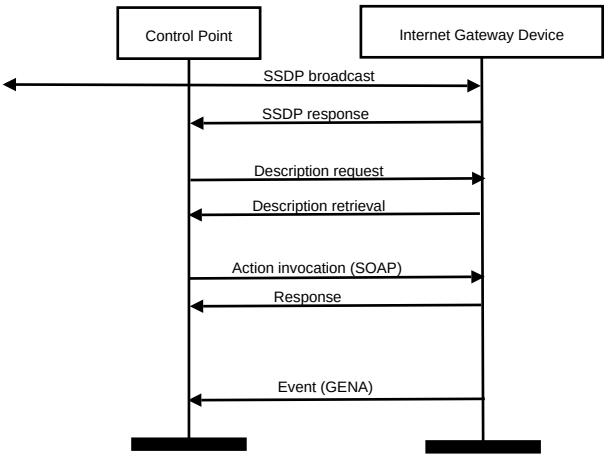

Figure 2: A UPnP message exchange

however, is beyond the scope of this paper. Please refer to the IGD specifications for additional details.[17]

3.4 SSDP

Universal Plug and Play -enabled applications and devices use the Simple Service Discovery Protocol (SSDP) for locating other UPnP-compliant network components. Here, clients can send out requests for specific services they require and servers can announce the services they provide. Initial discovery uses a multicast transmission to port 1900 on address 239.255.255.250, while later communication switches to unicast. SSDP uses HTTP over UDP for messaging. [15]

The point made here is that basically any application on the local administrative multicast domain [15] can discover and also control UPnP gateway devices.

3.5 SOAP

The Simple Object Access Protocol (SOAP) allows a UPnP client (CP) to actually invoke supported actions on a device, for example setting a port forward on an Internet Gateway Device.

SOAP is widely used in web service frameworks. [1]

3.6 GENA

The General Event Notification Architecture makes it possible for CPs to subscribe to event reports and receive notifications about state changes in controlled devices. This is, for example, necessary when several CPs are controlling the same device, as to synchronize client information. [4]

4 UPnP security

UPnP is not a single standard defined in one document, but rather a collection of service templates developed asynchronously. Because of this, UPnP is basically changing all the time. While many DCPs were released as version 1.0 in 2001, others have reached this state as late as in 2005. Release level UPnP security mechanisms were introduced in 2003.

The basic security architecture is presented in the UPnP Security Ceremonies Design Document. [9] The new component in this scenario is the Security Console (SC) [8], which handles the interaction between user and device, allowing the user to control access to a device. Without going into too much detail, the UPnP Security Ceremonies require security-aware devices to implement the Device Security service template.

Upon entering a network, such a device would announce its presence using SSDP and any SC requiring control of the device would then have to present a matching password. An SC taking ownership of a device in this way would then have complete control over it, including having the right to grant ownership to other clients (SCs). This architecture also provides both message integrity and confidentiality between devices and Control Points, while allowing previously defined service templates to be used in a normal fashion. [7]

4.1 Implementations

There has been some strong criticism towards UPnP because of vulnerabilities in various implementations of the protocol. One of the first problems to face a broader community was a security flaw in the UPnP system service of *Windows XP*. [21]

While this vulnerability certainly was critical in the sense that it was present on a very large number of hosts connected to the Internet, it is not relevant to this paper. UPnP security in Internet gateways that are not implementing the security service template anyway, is not affected by security issues in client software, one way or the other.

Still, it is worth noting that the UPnP specifications are complex and large, and implementations are therefore likely to have flaws.

4.2 Architectural issues

As a UPnP-compliant device is not required to implement the Device Security template, one could argue that the UPnP architecture is inherently insecure. Unfortunately, this problem is also present in reality. Most devices on the consumer market do not, in fact, implement any security mechanisms, but allow full access to their UPnP daemons. In many cases, a single piece of software running on any client, even with limited system privileges, can easily undermine the security of an entire protected network. This is such a big issue, that we feel that UPnP should really only be used in isolated networks or in networks with low security but high usability requirements.

Yaron Golan, one of the authors of the SSDP protocol specification, however, maintains that UPnP itself can not be blaimed for security implications in the context of gateway devices. [14]

4.2.1 Overview

It would seem that the most acute security problems in UPnP-enabled devices are associated with access control. There is a need for easily identifying clients who are authorized to alter gateway configuration.

In UPnP security, this is achieved by first having legitimate users identify themselves as such. Users will enter a predefined device password using the Security Console user interface. Typically, the password could be available on a small display on the device itself or on a permanent sticker, in the case of a static password. Supplying the correct password makes the SC owner of that device. This information exchange uses public key cryptography to ensure confidentiality. Public keys are retrieved using the *GetPublicKeys* action.

Once administrator identity has been established, session keys are exchanged between the SC and the device, using the *SetSessionKeys* action. These symmetric keys will subsequently be used for digitally signing all action invocations.

Other CPs can be granted access to devices by having the SC enter specific Access Control Lists (ACLs) or by using certificates to establish legitimacy.

If an action needs to be confidential, a special *DecryptAndExecute* action can be invoked. [9]

4.2.2 SSDP

Security of the discovery process of devices on the network is difficult to enhance, as we cannot limit announcements or answers to service requests to trusted hosts or clients. However, in UPnP security, we can limit the amount of information we output to unauthenticated CPs. For example, a device could publically announce itself simply as a "Security Aware Device".

4.2.3 SOAP

SOAP security is achieved by implementing the UPnP security template, which provides access control, message integrity, replay prevention and if necessary, confidentiality. Basically, SOAP security relies on XML-Signatures. [10]

For details on SOAP and XML security, please refer to this article by Alex Selkirk. [20]

4.2.4 GENA

Again, the UPnP security template supports limiting access to event variables to only certain CPs. Events are also encrypted using the normal session keys to hinder eavesdropping. [10]

A detailed analys of the UPnP security ceremonies is not presented in this paper. Readers may refer to the summary written by Robin Cover for a brief introduction. [5]

5 Discussion

Most problems stem from a combination of several conditions occurring at once. In this case, we are dealing with environments having certain technical properties and users possessing limited skills. This framework is then combined with relatively complex communication requirements and a current solution that only solves part of the problem.

5.1 Security vs. flexibility

Users have come to rely on their hardware firewalls and Internet gateways for security, when software applications and operating systems are failing them. A well configured firewall could even protect an otherwise compromised system from additional harm and attack.

Let us consider the outbreak of the malicious Nimda worm in the year 2001. If a system was infected with the virus, via email or another locally infected host, it would still not have been automatically exposed to the Internet for outsiders to exploit, provided that the local network was protected by a decent firewall. As the worm, among other things, exposed the system drives of the infected machines to anyone accessing them, it is obvious that firewalls greatly helped reduce the overall security impact of the outbreak and certainly were a lifesaver for the reputation of numerous companies.[3]

When autoconfiguration schemes are introduced into this scenario, the situation changes. A hardware firewall is no longer isolated from the software running on the network it is protecting. Rather, these entities become strongly connected, with the firewall dynamically responding to the communication needs of the applications.

As in many other cases, ease of use and security do not mix well. One of the key questions is how to achieve strong authentication and authorization for devices that are communicating for the first time. How do we make sure that only legitimate applications are allowed to make alterations in the firewall's configuration?

5.2 Industry solution

As stated previously in this paper, the UPnP architecture currently provides one solution for security. The UPnP Security Ceremonies specify a relatively complex set of protocols and procedures, which make designing a secure UPnP-compliant device both expensive and time-consuming. Also, since this scheme requires active human interaction, it is reasonable to ask whether this solution still is *Plug and Play* or not. Can we expect an average user to be able to enter a device password for authentication, and can we even require all legitimate users to have access to this password?

It would seem that this architecture is not quite mature, and requires some additional work. This paper, however, does not provide the answers to these key questions, but instead endeavours to point out where work needs to be done.

5.3 Related work

A quite interesting proposal for managing dynamic security configurations on network entities is being developed within the SECURE (Secure Environments for Collaboration among Ubiquitous Roaming Entities) [16] project. This research project essentially bases configuration of security on the human notion of trust, that is if A trusts B and B trusts C, most likely A can trust C. Hence, a key concept in this framework is trustworthiness, including trust formation, trust evolution and trust propagation. [12][16]

Applying SECURE to smart consumer market appliances has been proposed by others [19] as an alternative to, for instance, UPnP. In this architecture, trust and access rights propagate naturally without any direct user intervention, once sufficient information has been gathered or supplied by the interconnected devices. In this scenario, it would indeed be possible for an Internet gateway to automatically configure itself in a secure manner, based on trustworthiness information in other devices on the network.

In the case of UPnP, specifically, instead of building a totally new configuration framework for security, the principles of SECURE could be integrated into UPnP. This should be possible to achieve without breaking the existing UPnP framework, simply by adding this functionality into a new, albeit optional, service template.

Currently, several technologies offer functionality similar to UPnP, in the context of service discovery and dynamic configuration. Sun's Jini, the open Salutation architecture and OSGi (the Open Services Gateway Initiative) all solve some parts of the problem of autodiscovery or -configuration. [2]

5.4 Proposals and alternatives

Corporate firewalls have traditionally been designed to limit both incoming as well as outgoing traffic. It would seem fair

to assume that also gateway devices intended for the home consumer market should implement this feature, as is already the case in most software firewall applications. In the latter case, it is common for the firewall to have the user separately acknowledge each unknown outgoing connection attempt. While this is much easier to achieve with a piece of software running on the same computer as the communicating process, the concept of strictly limiting communication at the process level, using a border gateway device, could be appealing. This should greatly enhance overall security and provide additional protection against worms and virus outbreaks.

There are also other alternatives to using UPnP in gateway devices. One could even argue that the Internet gateway device is a mission-critical component of overall network security, and should therefore be totally separated from the weaknesses in any standard autoconfiguration scheme. Instead, effort should be put into simplifying the user interface of the gateway device, making manual configuration easier.

Standardizing the format of firewall rulesets could also prove helpful, making the process of opening a specific port on the device more of uploading a preformatted module, containing support for a new application level protocol. The user interface would then display an updated view of uploaded modules and dependencies between protocols and internal hosts.

Eventhough much can be done to facilitate the user experience of configuring a firewall, we still feel, however, that there is a need for more or less complete autoconfiguration. The alternatives presented above are probably not easy enough, and still require some sort of authentication scheme and user decision.

5.5 Future work

It is clear that this paper leaves many problems unresolved and possibly even raises more questions than it answers. It is our hope that some of the ideas presented may lead to other studies and findings.

5.5.1 Process-aware border firewall

It appears that border gateway devices are becoming an integrated part of local networks. Nodes and applications are communicating with the firewall for various reasons. Besides the security problems, this also opens up new possibilities.

In security-critical environments, one could picture a need for separately authorizing communication for individual processes running on certain hosts with certain privileges. For example, a gateway might allow the BIND name server daemon, running as user 'named', client DNS access to the Internet, while denying it to everyone else, even 'root'.

To make this possible, we need a protocol for securely communicating user, process and authorization information between local network nodes and the gateway. Obviously, this arrangement would require several components not discussed here, but could, when implemented, provide substantial improvement in security.

5.5.2 Standardizing rulesets

Configuring ports and firewall rules for various protocols is still quite a technical task, prone to errors and misconfigurations. Users and network administrators would greatly benefit from a modular, standardized ruleset format. Modules conforming to this specification could then be uploaded into gateways and firewalls, allowing them to be more manageable and easily expandable to support new protocols.

We feel that this framework should be built on a human-readable XML schema and a set of minimum specifications for devices supporting it. The framework should contain, but not be limited to, support for the following general features:

- IPv4 and IPv6
- IP protocol numbers
- Ports
- Source and destination
- DNS name resolution
- Protocol and port triggers
- Port forwardings
- Address translations
- Policies (time schedule, bandwidth, QoS etc.)

6 Conclusion

There is an increasing need for easily configurable, or autoconfigurating, network devices. This is particularly important on the home consumer market, as it is not realistic to expect an average user to grasp concepts like ports and advanced protocols.

One of the proposed, and partially implemented, solutions, Universal Plug and Play, allows applications themselves to commandeer firewalls and gateway devices. This greatly improves usability for users with limited technical skills, but also introduces problems. Most implementations of UPnP today are not secure, as they do not comply with the UPnP security specification. This is partially due to the fact that the original UPnP specification never included any security features, and partially due to the complex nature of the UPnP Security Ceremonies, which were released much later.

Eventhough Universal Plug and Play solves many problems in usability, it would seem that current security issues must be resolved before the protocol can be adopted in a broader context. In practice, this could mean redefining the whole security architecture of UPnP, effectively specifying a UPnP-2 protocol, which would make security a mandatory service on all devices complying to the standard. Generally, it is our belief that the UPnP architecture still has many good qualities, and should therefore not be abandoned.

References

[1] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. Simple Object Access Protocol (SOAP) 1.1. Technical report, W3C, May 2000.

[2] P. Bull, P. Benyon, and P.R.Limb. Residential gateways. *BT Technology Journal*, 20(2), April 2002.

[3] CERT/CC. Nimda worm. CERT Advisory CA-2001-26, CERT Coordination Center, September 2001.

[4] J. Cohen, S. Aggarwal, and Y. Goland. General Event Notification Architecture Base: Client to Arbiter. Technical report, IETF, September 2000.

[5] R. Cover. UPnP forum releases new security specifications for industry review. At `http://xml.coverpages.org/ni2003-08-22-a.html`, 2003. Referenced 8.2.2006.

[6] K. Egevang and P. Francis. The ip network address translator. RFC 1631, Internet Engineering Task Force, May 1994. `http://www.ietf.org/rfc/rfc1631.txt`.

[7] C. Ellison. Device Security:1 Service Template. Technical report, UPnP Forum, November 2003.

[8] C. Ellison. SecurityConsole:1 Service Template. Technical report, UPnP Forum, November 2003.

[9] C. Ellison. UPnP Security Ceremonies. Technical report, UPnP Forum, November 2003.

[10] C. M. Ellison. Home network security. *Intel Technology Journal*, 06(04), November 2002.

[11] eMule-Project.net. eMule. At `http://www.emule-project.net`. Referenced 19.4.2006.

[12] C. English, P. Nixon, S. Terzis, A. McGettrick, and H. Lowe. 5th ieee international workshop on networked appliances. In *Security Models for Trusting Network Appliances*, Liverpool, UK, October 2002.

[13] U. Forum. UPnP Device Architecture 1.0. Technical report, UPnP Forum, December 2003.

[14] Y. Goland. UPnP security flaws. At `http://www.goland.org/upnp_security_flaws/`, 2002. Referenced 8.2.2006.

[15] Y. Goland, T. Cai, P. Leach, and Y. Gy. Simple Service Discovery Protocol/1.0. Technical report, IETF, October 1999.

[16] D. S. Group. Secure, secure environments for collaboration among ubiquitous roaming entities. At `http://secure.dsg.cs.tcd.ie`. Referenced 22.3.2006.

[17] P. Iyer and U. Warrier. InternetGatewayDevice:1 Device Template Version 1.01. Technical report, UPnP Forum, November 2001.

[18] M. Schmitz, P. Iyer, and U. Warrier. WANIPConnection:1 Service Template Version 1.01. Technical report, UPnP Forum, November 2001.

[19] J.-M. Seigneur, C. D. Jensen, S. Farrell, E. Gray, and Y. Chen. Smart objects conference soc'2003. In *Towards Security Auto-Configuration for Smart Appliances*, Grenoble, France, May 2003.

[20] A. Selkirk. Using xml security mechanisms. *BT Technology Journal*, 19(3), July 2001.

[21] Technet. Unchecked buffer in universal plug and play can lead to system compromise. Microsoft Security Bulletin MS01-059, Microsoft, December 2001. At `http://www.microsoft.com/technet/security/bulletin/ms01-059.mspx`.

[22] UPnP Forum. About the UPnP Forum. At `http://www.upnp.org/about/default.asp`. Referenced 21.3.2006.