

Explicit Mechanisms for Controlling NAT/Firewall Systems Dynamically

Teemu Mäki

Helsinki University of Technology

Teemu.Maki@hut.fi

Abstract

NAT/Firewall systems are very widely utilized in today's network environments, such as intranets and the Internet. Together with many security enhancements, however, the use of NATs/Firewalls has raised a number of new issues and problems. Most of them related to the complex NAT/Firewall traversal problem, where applications and services are blocked to connect to each other by NATs/Firewalls. In order to solve this kind of problems, many implicit mechanisms have been proposed to configure NAT/Firewall systems dynamically on run-time. Although implicit mechanisms overcome, at least partially, NAT/Firewall traversal problems, they also introduce new ones. In addition to the new problems, the biggest disadvantage with implicit methods is the lack of control over the configuration process. To overcome this, many explicit mechanisms (e.g. UPnP, NAT/Firewall NSLP, MIDCOM and Port Knocking) have been developed.

This paper describes existing and most commonly used explicit mechanisms for controlling NAT/Firewall systems dynamically. The focus of this paper, in addition to the overview of the mechanisms, is to compare the mechanisms with each other and to give some concluding remarks. Existing mechanisms are reviewed from the security and performance perspectives, but some usability issues are also covered.

KEYWORDS: Dynamic Firewall Configuration, UPnP, NAT/Firewall NSLP, MIDCOM, Port Knocking

1 Introduction

Firewalls are commonly deployed for controlling network traffic by means of traffic filtering. Traditionally, the role and the existence of firewall systems have been viewed as static features only. In a typical scenario, the firewall exists at the boundary between the private and external networks protecting the private network from unauthorized access from external networks. The protection is based on a predefined firewall rule set, namely the firewall security policy that determines what is allowed to pass the firewall and what is not. Changing the above-mentioned firewall rule set and the firewall maintenance process overall, i.e. firewall configuration, has mainly been a complex and manual job done by experienced network administrators only. However, only a few of today's firewall implementations are purely static. Most

of them also have dynamic aka run-time reconfigurable features such as NAT (Network Address Translation). NAT can be utilized to masquerade outgoing traffic with global IP address(es) (i.e. address + port) and to allow incoming traffic only from the established connections (that is, outbound connections already initiated by NAT). In other words, incoming traffic is limited according to the dynamic rules that are implicitly added/removed for outbound connections by NAT. Through that way NAT makes it possible to hide addresses in private networks and to deny connection attempts from external networks. In this paper, the NAT/Firewall system is the intended type when referring to firewalls or firewall systems.

Constant innovation and development of new, more and more complex applications and services set new requirements for traditional network systems [1]. In many cases, it is a known fact that purely statically controlled firewalls can not be utilized, as such, anymore. The fundamental problem with statically configured firewalls is known to be their inflexibility in dynamic environments. This holds especially true for grid-based computing and distributed systems where a variety of dynamic services and resources are distributed across a number of networks [3]. Because of the distributed architecture, services and resources are typically blocked from each other by several firewalls. However, using static firewall configuration to enable the communication between different services and resources would make the system permanently more open to intruders and dramatically lower the whole system's security level [3]. Very often these kinds of problems are solved using some specific NAT/Firewall solution. However, in most of these solutions the dynamism is implicit, but not explicit. That is, the firewall rule set is implicitly altered by NAT, not explicitly on request from an application or from any other entity. As a result, the lack of control over configuration process still remains as a problem to be solved. One example are average home users and their home LANs that are becoming more and more general, but cannot be typically accessed from outside due to NATs/Firewalls. However, home LANs are a useful way to share resources and services such as music, videos, electronic devices, and should be securely accessible from remote locations without worrying about complex firewall configurations [8]. In other words, home users should be able to explicitly send configuration requests (e.g. pinhole requests) to control their own NAT/Firewall system.

As the previous paragraphs indicate, many kinds of problems exist, most of which are related to wasting existing resources. To address these problems, many explicit meth-

ods have been developed for configuring NAT/Firewall systems dynamically. This paper provides information about those methods. The subject is reviewed through presenting commonly used protocols that provide required configuration services. However, the aim is not to review all existing protocols. The presented protocols are mainly discussed and analyzed from the security and performance perspectives. The usability is an issue that should not be totally ignored. That's why this paper also provides some information on that issue.

The organization of this paper is as follows. The overall problem setting for using dynamic firewall configuration is introduced in section 2. Section 3 gives some background information on existing and commonly used protocols for solving this problem. Analysis of most important factors and features of existing protocols is done in the section 4. Finally, section 5 summarizes the paper and provides some concluding remarks.

2 Problem Definition

As pointed out in the introduction, an obvious need exists for explicit and dynamic firewall configuration methods. However, utilizing those methods is not a matter of course that could be bypassed totally. Therefore, certain things must be thoroughly considered when ensuring that new solutions meet requirements and can be distributed to the general public [6].

Of course, the most important and interesting problems relate to the security. In theory, dynamic firewall configuration should increase the security level and responsiveness compared to static methods, because firewall pinholes are made through strong authentication and on demand only [2, 10]. Unfortunately, this is not the only viewpoint on this issue. In particular, strong criticism has been leveled against some authentication methods [6].

In addition, there are the issues related to the performance. It is a commonly known fact that, at least enterprise firewalls, may be quite fully employed already and therefore performance issues are very critical [12]. Thus any additional load caused by firewalls and their configuration is unwelcome.

The last problem discussed in this paper is the usability of existing mechanisms. This subject is not covered very thoroughly, but enough to cover the bases. Certainly, there are many other problem areas associated with firewall configuration methods, but those areas are not covered in this paper.

3 Background

Sections 3.1 - 3.5 provide some background information regarding existing and most commonly used protocols for configuring firewalls explicitly and dynamically.

3.1 UPnP

This section is based on the white paper "Understanding UPnP" and on the description document of the Internet Gateway Device (IGD) published by UPnP forum [14, 15].

UPnP (Universal Plug and Play) is an industrial standard defined and maintained by group of companies called UPnP Forum. UPnP is not a single protocol but rather an open network architecture consisting of many standardized and existing Internet protocols. In other words, UPnP architecture is not just for firewall configuration, but it provides one way of doing it. UPnP leaves the work concerning API definition and implementation totally to other parties. On top of that, UPnP is not dependent on existing operating systems, programming languages or physical medias.

One of UPnP's primary goals is to provide fully automatic configuration mechanisms (i.e. zero configuration) for all network devices supporting UPnP and to enable direct peer-to-peer communication between them. As a result, a wide variety of devices can be easily connected, and more networking services can be made available to low end users too.

UPnP network consist of three main components which include devices, services and control points. Briefly said, devices provide different kinds of services, and are discovered and controlled by control points. The more detailed information of UPnP is out of this paper's scope and the rest of this section focuses on how to utilize UPnP with firewall configuration.

UPnP's solution to the NAT traversal problem is the Internet Gateway Device (IGD). It provides automatic NAT traversal as shown in the example below (Figure 1).

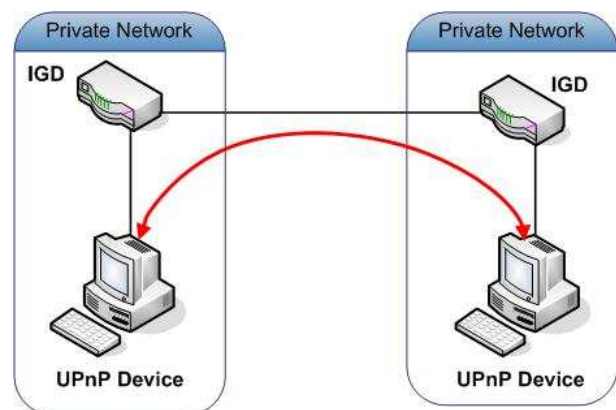


Figure 1: UPnP IGD enabled peer-to-peer connection between two UPnP devices.

Unfortunately the current version 1.0 of IGD does not directly support firewall features. Those features are handled only as additional features (i.e. vendor extensions). Similarly, out of scope are security issues such as those related to authentication, authorization and access control. However, those issues may be addressed through use of the UPnP security architecture that extends the UPnP device architecture. In addition, some absent issues are meant to be fixed or implemented as new features in the next version of IGD. For example, support for the firewall configuration is a new feature that is introduced in the version 2.0 of IGD.

3.2 NAT/Firewall NSIS Signaling Layer Protocol (NSLP)

Issues addressed in this section are built on notions introduced and discussed in the Internet-Draft paper on the NAT/Firewall NSIS Signaling Layer Protocol published by IETF NSIS work group [13].

The IETF NSIS work group is currently working on defining a protocol called NAT/Firewall NSIS Signaling Layer Protocol (NAT/Firewall NSLP). Just as the name implies, NAT/Firewall NSLP is a signaling protocol for IP networks, and utilized for explicit and dynamic configuration of NAT/firewall systems along the path between any two network devices. The purpose is either to enable or to block a data flow between these two end points.

NAT/Firewall NSLP utilizes so called path-coupled approach to signaling. In other words, signaling packets go through exactly the same NAT/Firewall systems (i.e. same path) as actual data packets. An example of a possible NAT/Firewall configuration using the NAT/Firewall NSLP protocol is illustrated in the figure 2.

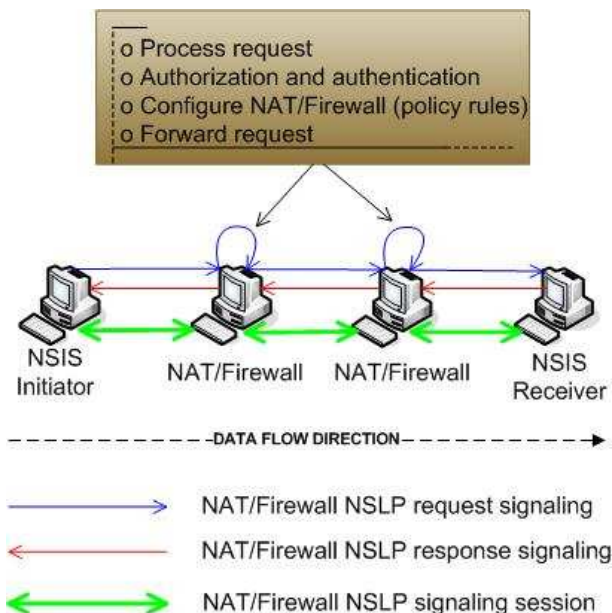


Figure 2: Dynamic firewall configuration using NAT/Firewall NSLP protocol [13].

The NSIS framework defines a clear boundary (i.e. layers) between the transport of signaling messages and the signaling logic. The NSIS Network Transport Layer Protocol (NTLP) with General Internet Signaling Transport (GIST) is responsible for transport issues. The signaling logic, on the other hand, is based on signaling specific NSIS Signaling Layer Protocols (NSLP), such as NAT/Firewall NSLP.

The security framework of the NAT/Firewall NSLP is mainly based on authentication and authorization of NAT/Firewall NSLP entities requesting a change to NAT/Firewall policy rules. Furthermore, integrity of NAT/Firewall NSLP messages is a fundamental aspect and must be guaranteed. However, the confidentiality protection may be optional.

3.3 Middlebox Communication (MIDCOM)

The information presented in this section comes mainly from the RFC 3303 document on the "Middlebox communication architecture and framework" published by the IETF MIDCOM working group [11]. In this section the term middlebox is used to refer to a NAT/Firewall system.

MIDCOM is a framework, whose primary objective is to separate all application specific aspects (i.e. application intelligence) from middleboxes. Application intelligence is delegated from middleboxes to trusted third parties which are called as MIDCOM agents in this context. The framework sets out a series of requirements for the MIDCOM protocol in order to make sure the independence of applications. All the things mentioned above have many positive effects. For example, applications and services can be easily added or removed without the need to maintain middleboxes. Furthermore, simple implementations of middleboxes increase their robustness and performance.

Figure 3 shows the MIDCOM architecture and framework in action. Peer-to-peer connection between hosts is enabled by means of utilizing the MIDCOM protocol to the NAT/Firewall configuration. Hosts use MIDCOM agents to open the direct peer-to-peer connection.

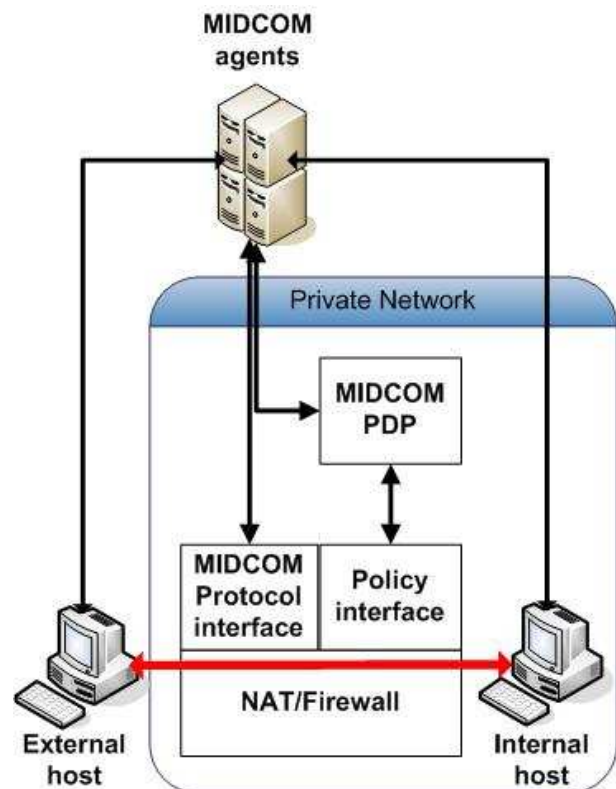


Figure 3: MIDCOM enabled peer-to-peer connection between two hosts [11].

MIDCOM agents register/unregister to the middlebox after the authentication/authorization process with the MIDCOM Policy Decision Point (PDP). Registered agents are authorized to access some services (e.g. opening a firewall port) provided by the middlebox. On the other hand, the middlebox is allowed to request MIDCOM agents to han-

dle application level tasks, such as ALG (Application Level Gateway) functions.

In addition to the 2-way authentication and to the authorization processes between MIDCOM agents and the middlebox, several security mechanisms (e.g. IPSec, TLS) can be used to provide data authentication, integrity and confidentiality.

3.4 Port Knocking

The content of this section is based on the article about port knocking, a concept that was firstly introduced by Martin Krzywinski [4].

Martin Krzywinski himself has described port knocking briefly as follows: "Port knocking is a method of establishing a connection to a networked computer that has no open ports". In other words, port knocking is an authentication mechanism for firewall systems to dynamically open ports to authenticated parties only. The authentication is based on the "knock sequence" provided by the party that is trying to connect. In order to give a more detailed explanation of port knocking, a series of figures available on the project's homepage is presented here with some explanatory notes.

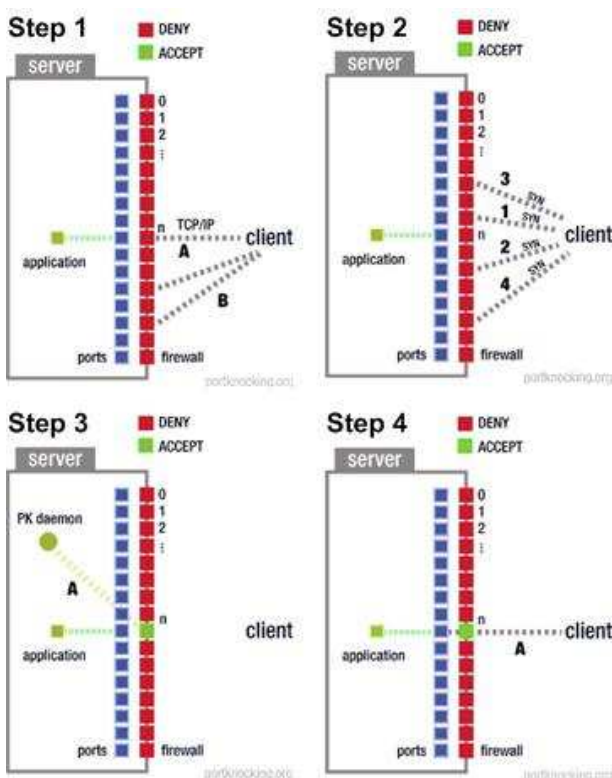


Figure 4: Port knocking method described in four separate steps (<http://www.portknocking.org>, 2006)

As we can see in the figure 4, step 1 shows the initial situation in which a firewall is configured by default to block all traffic to a server. There are no open ports for clients to connect. In step 2, a client does the "authentic knocking" by means of connecting some server ports in certain order (i.e. the knock sequence). The client gets no response to connection attempts, because the firewall is still blocking everything. In step 3, the server authenticates and analyzes the

knock sequence, and opens a port if the knock sequence is successful. Finally, the client is capable to connect to the server (Step 4).

There are many slightly different versions and implementations of port knocking mechanism. However, a number of security issues have been raised concerning those implementations. Some have even said that port knocking mechanisms rely on the "security by obscurity" [9]. In addition, many implementations have some compatibility problems with existing systems, especially with client NATs [9].

3.5 Other Mechanisms

There are also many other mechanisms meant to explicit and dynamic configuration of middleboxes. One of them (and one of the oldest too) is the protocol called SOCKS [5]. In addition to SOCKS, a number of very specific mechanisms exist. However, those mechanisms are out of scope and not discussed in this paper.

4 Analysis and Evaluation

This section gives an analysis and evaluation of most important factors (defined in the Introduction) of the NAT/Firewall configuration mechanisms described in the sections 3.1 - 3.4. The section is divided into subsections according to the factors which are used to analyze and evaluate the mechanisms.

4.1 Security

There is no doubt that the security is the most important thing, when considering NAT/Firewall systems. Therefore, as already said before, these issues must be carefully evaluated before utilizing new mechanisms to NAT/Firewall configuration.

The previously mentioned reason is one of the biggest threats to UPnP and to its popularity as a NAT/Firewall configuration mechanism. That is mainly because security issues have not been addressed in the original design of UPnP framework [7]. As a result, many issues such as firewall features are out of the scope of the current Internet Gateway Device Control Protocol (DCP). In order to address the lack in the security model, UPnP utilizes so-called security extensions. However, those extensions are not widely adopted because vendors and developers find them too complex to utilize [7]. This is a serious problem and must be solved before large-scale adoption of UPnP is possible.

In contrast to UPnP, the design of NAT/Firewall NSLP protocol has included security concerns already from the beginning. That is why NAT/Firewall NSLP should not suffer from the same kind of "surprising" threats as UPnP. However, NAT/Firewall NSLP protocol has not been widely utilized because it is still in the Internet-Draft phase. Therefore, in all probability, new issues will be found when the final version of the protocol is published.

As pointed out in the 3.3, several security methods can be utilized with MIDCOM framework to provide the required authentication, authorization, confidentiality and integrity

mechanisms. Most of the threats against MIDCOM framework relates to the authentication and authorization process between MIDCOM entities [11].

Port Knocking is the most criticized NAT/Firewall configuration mechanism presented in this paper. Especially the authentication mechanism in many existing solutions is not strong enough. At least cryptographic methods should be employed in order to prevent intruders, for example, to replay authenticated port knocks [9]. Unfortunately, more problematic issue is the DoS attacks. Those can be prevented only partially using trusted/blocked IP ranges for incoming port knocks [4]. Last problem relates to client NATs and was already pointed out in section 3.4. In the Port Knocking framework it is possible for a client (behind a NAT) to open a port to all clients behind the same NAT because they share the same address [6]. This is a serious security flaw that needs to be addressed.

4.2 Performance

The second area of consideration, related to the NAT/Firewall configuration mechanisms, is the performance. Performance issues of NAT/Firewall systems are generally considered very important. Nevertheless, the only performance study found for this paper concerns the NAT/Firewall NSLP protocol [12]. Other mechanisms are analyzed and evaluated on the basis of the architectural information only.

The example implementation of the NAT/Firewall NSLP described in the article [12] verifies that the protocol is applicable for the intended purpose. On top of that, the result of the performance study is very positive. Performance seems to be more dependent on the NAT/Firewall implementation than the NAT/Firewall NSLP implementation [12].

As the section 3.3 describes, the MIDCOM framework is designed so that the application specific logic is delegated to MIDCOM agents. This kind of architecture is very scalable in performance because of the simple NAT/Firewall implementation [11]. In other words, performance issues in MIDCOM protocols should not be the problem.

Most of the implementations of Port Knocking systems are quite minimal and said to be scalable in performance too [4]. However, it is worth remembering that some issues (e.g. DoS attacks) discussed in 3.4 may have a great impact to the performance of Port Knocking systems.

UPnP framework (with security extensions and other features) is much more complicated than any other mechanism presented in this paper. Most of the components in the UPnP environment are not loosely coupled. This implies that UPnP implementations are not as scalable in performance as other mechanisms described above.

4.3 Usability

Usability is an issue that should not be totally omitted when considering NAT/Firewall configuration mechanisms. Especially this holds true when the user has a significant role in the configuration process. Of course the user interface has a direct impact on usability. However, UPnP is the only mech-

anism in this paper that defines the remote UI client for managing, for example, remote connections.

The paper [8] shows a convenient UPnP based implementation of the dynamic firewall configuration mechanism. The solution is targeted at common home-users. Usability has been carefully taken into account and the NAT/Firewall system can be configured even with a TV remote control.

5 Conclusion

Explicit NAT/Firewall configuration mechanisms are needed, along with the implicit ones, to cover the lack of control to the configuration process. A number of proposals have been introduced in order to overcome this problem. This paper made a throughout review on existing mechanisms, namely UPnP, NAT/Firewall NSLP, MIDCOM and Port Knocking, for controlling NAT/Firewall systems explicitly and dynamically. The existing mechanisms were reviewed from three different aspects: security, performance and usability.

All the reviewed mechanisms seemed to have open security issues or vulnerabilities. This is especially true for UPnP and Port Knocking. By contrast, the analysis revealed that performance is less critical factor than security. However, the most critical problem among explicit NAT/Firewall configuration mechanisms seemed to be the scalability in different network environments, such as multiple NAT/Firewall systems. Anyway, this subject was out of scope for this work, and was not examined to a greater depth.

The future of explicit NAT/Firewall configuration mechanisms is currently unknown. There is no single solution that suits every situation. However, UPnP is the only industrial standard among them. In other words, UPnP has a strong lead over the others. It is already supported by a growing number of hardware/software vendors. For example, UPnP is supported by Windows ME/XP.

References

- [1] M. S. Blumenthal and D. D. Clark. Rethinking the design of the internet: the end-to-end arguments vs. the brave new world. *ACM Transactions on Internet Technology (TOIT)*, 1(1):70–109, 2001. Available from <http://doi.acm.org/10.1145/383034.383037>.
- [2] R. Ellis. Dynamic firewall techniques for residential use. Master's thesis, Southern Connecticut State University, New Haven, Connecticut, 2004. Available from http://home.southernct.edu/~lancor11/GraduateStudents/Robert%20Ellis/thesis_draft_01_26_04.pdf.
- [3] M. L. Green, S. M. Gallo, and R. Miller. Grid-enabled virtual organization based dynamic firewall. In *GRID '04: Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID'04)*, pages 208–216, Washington, DC, USA, 2004. IEEE Computer Society. Available from <http://dx.doi.org/10.1109/GRID.2004.35>.

- [4] M. Krzywinski. Port knocking: Network authentication across closed ports. *SysAdmin Magazine*, 12(6), 2003. Project's homepage at <http://www.portknocking.org>.
- [5] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. Socks protocol version 5. RFC, 1996. Available from <http://tools.ietf.org/html/rfc1928>.
- [6] A. I. Manzanares, J. T. Márquez, J. M. Estevez-Tapiador, and J. C. H. Castro. *Attacks on Port Knocking Authentication Mechanism*, page 1292–1300. Springer Berlin / Heidelberg, 2005. Available from http://dx.doi.org/10.1007/11424925_134.
- [7] MediaNet. Report on a secure home network architecture and related protection profiles specification. Technical report, MediaNet, 2005,2006. Available from http://www.ist-ipmedianet.org/DA.3.7_UPnP_Security_V1_0.pdf.
- [8] S. Mizuno, K. Matsuura, K. Yamada, and K. Takahashi. A new remote configurable firewall system for home-use gateways. In *Second IEEE/Consumer Communications and Networking Conference. CCNC.*, pages 599–601, 2005. Available from <http://dx.doi.org/10.1109/CCNC.2005.1405247>.
- [9] J. A. Rennie deGraaf and M. J. Jr. Improved port knocking with strong authentication. In *21st Annual Computer Security Applications Conference (ACSAC 2005)*, pages 451–462, 2005. Available from <http://dx.doi.org/10.1109/CSAC.2005.32>.
- [10] S. Son, B. Allcock, and M. Livny. Codo: Firewall traversal by cooperative on-demand opening. In *14th IEEE International Symposium/High Performance Distributed Computing. HPDC-14.*, pages 233–242, 2005. Available from <http://dx.doi.org/10.1109/HPDC.2005.1520965>.
- [11] P. Srisuresh, J. Kuthan, J. Rosenberg, A. Molitor, and A. Rayhan. Middlebox communication architecture and framework. RFC, 2002. Available from <http://tools.ietf.org/html/rfc3303>.
- [12] N. Steinleitner, H. Peters, and X. Fu. Implementation and performance study of a new nat/firewall signaling protocol. In *ICDCSW '06: Proceedings of the 26th IEEE International Conference Workshops on Distributed Computing Systems*, page 8, Washington, DC, USA, 2006. IEEE Computer Society. Available from <http://dx.doi.org/10.1109/ICDCSW.2006.63>.
- [13] M. Stiernerling, H. Tschofenig, C. Aoun, and E. Davies. Nat/firewall nsis signaling layer protocol (nslp). Internet-Draft, 2007. Available from <http://www.ietf.org/internet-drafts/draft-ietf-nsis-nslp-natfw-14.txt>.
- [14] UPnP Forum. Understanding universal plug and play: A white paper. Technical report, Microsoft, 2000. Available from http://www.upnp.org/download/UPNP_UnderstandingUPNP.doc.
- [15] UPnP Forum. Upnp internetgatewaydevice:1 device template version 1.01. Technical report, Microsoft, 2001. Available from http://www.upnp.org/standardizeddcpes/documents/UPnP_IGD_1.0.zip.