Published as: Hosnieh Rafiee, Ahmad AlSa'deh, Christoph Meinel, WinSEND: Windows SEcure Neighbor Discovery, in Proceedings of the 4th international conference on Security of information and networks - SIN '11, pp. 243-246, ACM Press, , Sydney, Australia, November 14–19, 2011.

WinSEND: Windows SEcure Neighbor Discovery

Hosnieh Rafiee, Ahmad AlSa'deh, Christoph Meinel
Hasso-Plattner-Institut, University of Potsdam
P.O. Box 900460, 14440 Potsdam, Germany
{Hosnieh.Rafiee, Ahmad.Alsadeh, Christoph.Meinel}@hpi.uni-potsdam.de

ABSTRACT
Neighbor Discovery Protocol (NDP) is an essential protocol in IPv6 suite, but it is known to be vulnerable to critical attacks. Thus, SEcure Neighbor Discovery (SEND) is proposed to counter NDP security threats. Unfortunately, operating systems lack the sophisticated implementations for SEND. There is limited success with SEND implementation for Linux and BSD, and no implementation for Windows families. Therefore, the majority of the users are not secured with SEND. In this paper, we will introduce an implementation of SEND for Windows families (WinSEND). WinSEND is a user-space application which provides the protection for NDP in Windows. It has direct access to Network Interface Card (NIC) and efficiently handles NDP messages by using *Winpcap*. WinSEND works as a service with easy user interface to set the security parameters for selected NIC.

Categories and Subject Descriptors
C.2.2 [**Computer-Communication Networks**]: Network Protocols; D.4.6 [**Operating Systems**]: Security and Protection; D.4.9 [**Operating Systems**]: Systems Programs and Utilities

General Terms
Security, Experimentation

Keywords
SEND implementation, Cryptographically Generated Addresses (CGA), IPv6 security and protection, Neighbor Discovery Protocol (NDP)

1. INTRODUCTION
Neighbor Discovery Protocol (RFC 4861 and RFC 4862) does not have a built-in security mechanism to enable nodes to authenticate each other. Therefore, NDP is prone to critical attacks [1]. NDP assumes that all nodes on the link trust each other, but this assumption cannot be guaranteed. Consequently, a malicious user can impersonate legitimate nodes by forging NDP messages to generate intentional serious attacks. Therefore, Internet Engineering Task Force (IETF) working group, IETF SEND, proposes the Secure Neighbor Discovery (SEND) [2] as an extension to NDP. SEND uses RSA key pairs, Cryptographically Generated Addresses CGA [3], digital signature and X.509 certification to offer significant protection to NDP.

SIN'11, November 14–19, 2011, Sydney, Australia.

Unfortunately, there is no major operating system (OS) providing an effective level of support for SEND. The current SEND implementations for specific OS distributions such as Linux and FreeBSD are basically experimental codes rather than production ready software. SEND is not supported yet in Windows XP/Vista/7 [4]. This means that the majority of hosts are not secured by SEND, since the Windows family is the most popular operating system and accounts for more than 80% of usage compared to other operating systems [5]. This lack of SEND support leaves IPv6 local networks vulnerable to attacks and consequently limits IPv6 deployment.

Thus, we decided to implement SEND from scratch and offer it as a service for Windows families. Our implementation (WinSEND) is a user-space implementation which is developed in Microsoft .NET. WinSEND works as service for Windows families with easy user interface to set security parameters for the proper Network Interface Card (NIC). To the best of our knowledge, WinSEND is the first SEND implementation for Windows families. This paper describes the design and the implementation of WinSEND.

The paper is structured as follow. An overview of NDP and possible attacks against it is presented in Section 2. Section 3 shows how SEND can protect NDP. In Section 4, we list the existing SEND implementations. Section 5 discusses the design choice of WinSEND implementation. Section 6 shows the WinSEND implementation. The last section concludes the work.

2. NEIGHBOR DISCOVERY PROTOCOL

2.1 NDP Messages and Functionalities
Neighbor Discovery (ND) for IPv6 [6], and IPv6 StateLess Address Autoconfiguration (SLAAC) [7], together are referred to as IPv6 Neighbor Discovery Protocol (NDP). NDP is one of the main protocols in IPv6 suite. NDP greatly improves the efficiency and the network management. It is also heavily used for several critical functionalities, such as discovering other existing nodes on the same link, determining others' link layer addresses, detecting duplicate addresses, finding routers and maintaining reachability information about paths to active neighbor. Also, NDP plays a crucial role in mobile IPv6 (MIPv6) networks [8].

NDP functionalities are based on five ICMPv6 messages: Router solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS), Neighbor Advertisement (NA), and Redirect. RS is sent by hosts to request for RA. Routers advertise the link

local prefixes and other options by RA. RA is sent periodically or as a response to RS. NS is sent by IPv6 hosts to find neighbors on the link or to verify the reachability of nodes. NA is sent by IPv6 host as a response to NS or to propagate the link layer address. Redirect is sent by routers to inform hosts about the preferred first-hop destination.

2.2 NDP Security and Privacy Implications

NDP offers only selected basic protective mechanisms to ensure that packets come from nodes that directly connect to a local link. However, this protection shield is not enough to protect IPv6 local networks. If NDP is not secure, it is vulnerable to a set of attacks. IPv6 Neighbor Discovery (ND) Trust Models and threats [1], describes these attacks. The attacker can carry out several attacks based on address resolution, redirect, Duplicate Address Detection (DAD), Router Advertisement (RA), and address configuration. Besides, the SLAAC leads to privacy problems. Since generating the interface identifier from the MAC address, (which remains constant over time), this makes it possible to track a node over the Internet. Also, SLAAC make it easy to correlate the traffic patterns and the activities to the certain user [9].

3. SECURE NEIGHBOR DISCOVERY

SEcure Neighbor Discovery (SEND) [2] offers three additional features to NDP: Address ownership proof, message protection and router authorization mechanism. To achieve these enhancements, SEND comes with four new options: *CGA*, *RSA Signature*, *Timestamp*, and *Nonce*.

- *CGA Option*: carries the associated parameters to enable the receiver to validate the proper binding between the public key (used to verify the signature) and the Cryptographically Generated Address (CGA). Also, CGA depends on a security parameter (*Sec*) which is chosen by the user to determine the desired security level against brute force attack. Sec value is not carried by CGA option. It is encoded in the IP address bits.
- *RSA Signature Option*: this option authenticates the identity of the sender. The sender sign messages with the private key which is related to the public key has been used in CGA's generation algorithm. This signature prevents an attacker from spoofing CGA addresses.
- *Nonce Options*: is used to protect messages from replay attacks, and to ensure that an advertisement is a fresh response to a solicitation sent earlier by the node.
- *Timestamp Option:* is used to ensure replay protection against unsolicited advertisements, such as periodic RA and Redirect messages.

SEND uses an Authorization Delegation Discovery (ADD) process to validate and authorize IPv6 routers to act as default gateways, and specifies the IPv6 prefixes that a router is authorized to announce on the link. ADD relies on an electronic certificate issued by a trusted third party. Before any node can accept a router as its default router, the node must be configured with a trust anchor(s) that can certify the router via certificate paths. So, the node requests the "router" to provide its X.509 certificate path to a Trust Anchor (TA) which is preconfigured on the node. The "router" should not be trusted if it fails to provide the path to TA.

Two new ICMPv6 discovery messages are offered for identifying the router authorization process: the *Certificate Path Solicitation (CPS)*, and the *Certificate Path Advertisement (CPA)*. A CPS message is sent by hosts during the ADD process to request a certification path between a router and one of the host's trust anchors. The CPA message is sent in reply to the CPS message and contains the router certificate.

4. SEND IMPLEMENTATIONS

There are some experimental implementations for SEND which are done for Linux and *BSD. Some of these implementations are done in user-space and others are done at the kernel. NTT DoCoMo USA Labs [10] implemented the first open source user-space implementation of SEND. Their implementation (send-0.2) works on FreeBSD. However, DoCoMo USA Labs is no longer maintaining SEND project and source code is no longer available for downloading in the web site and the support has been canceled. NDprotector [11] is another user-space implementation of CGA and SEND for Linux based on *Scapy6* and it is limited to Linux platform due to its dependency on *iproute2*, *ip6table*, and *netfilter* queue. Easy-SEND [12] is a further Linux user-space implementation of SEND developed in Java. Easy-SEND is an open source projects which is developed for educational purpose.

Other implementations try to integrate the SEND with the NDP code at the kernel. In Native SeND kernel API for *BSD (send-0.3) implementation [13], a new kernel module (*send.ko*) is implemented to act as a gateway between the network stack and the user-space interface. Huawei and BUPT (Beijing University of Post and Telecommunications) introduced an implementation for SEND [14] within the Linux kernel IPv6 module. This work is a research prototype under development, which still lacks the interoperability testing. Bugs that could even cause the kernel to crash are expected.

From our literature review, we did not find any SEND implementation for Windows families. Therefore, we decided to implement SEND for Windows. More details about the design and architecture of this implementation (WinSEND) appear in the following sections.

5. WinSEND DESIGN

5.1 WinSEND architecture

For WinSEND implementation, *Winsock* or *Winpcap* can be used to transfer data between Network Interface Card (NIC) to the upper layers and vice versa. *Winsock* API is implemented in Windows to allow the access to network services, especially TCP/IP. It is designed and implemented based on BSD sockets with some new functionality that enables API to support windows standard programming models. *Winpcap* project started about 10 years ago as a need to run network as an analyzing and *tcpdump* tool on Windows machines, which later became known as *libpcap* and *tcpdump* ported on Microsoft OS [15].

WinSEND uses *Winpcap* library which has direct access to the raw sockets. *Winpcap* offers the possibility for an application to receive/send network traffic before processing it by the OS. Thus, by using *Winpcap*, the network traffic can bypasses the OS and give an application the direct access to link layer. Moreover, *Winpcap* has the following advantages over *Winsock* API:

- *SIO_RCVALL* [16] control code (in *Winsock*), enables a socket to receive all raw packets. However, WinSEND is just interested in ICMPv6 messages that are related to NDP

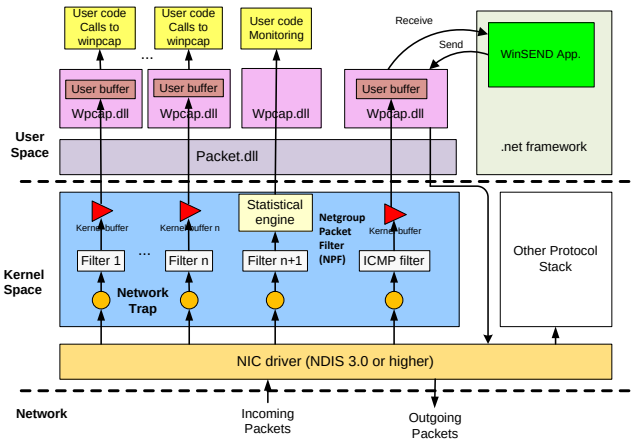

Figure 1. WinSEND architecture for sending and receiving packets via `Winpcap` library

messages and left the normal TCP/IP stack to handle the other traffic. Thus, all other traffics that are not related to NDP should be filtered out. In *Winsock*, this filtering is done in user-space and it takes a large amount of CPU power, and the kernel buffer will be out of space and force to drop packets [17]. So one solution is to bypass *Winsock* and enable the application to interact to either the Transport Data Interface (TDI) layer or to Network Driver Interface Specification (NDIS) [18].

- Overall performance of *Winpcap* in capturing and transmitting data is better than *Winsock*. *Winpcap* is located below *tcp.sys* in NDIS that can bypass windows protocol stack. To speed up data processing in *Winsock*, some of network services, such as firewall and network list services can be disabled. However, the memory usage and response time is still higher than *Winpcap*. Also, some of critical windows updates that are based on those services will fail. [19]

Figure 1 shows the Architecture of WinSEND and the involving libraries in sending and receiving traffic. *Winpcap* is a main external API in WinSEND application. It is subdivided into three components: a packet capture device driver, a low level dynamic library which is called Netgroup Packet Filter (NPF), and a high level static library. When traffic arrives to a network adapter, the network adapter invokes the "Network Trap" which copies packet to NPF. Then, NPF applies user defined filter, i.e. *icmpv6*, and sends the captured packets to user-space. *Wpcap.dll* compiles user defined filters. Also, *Wpcap.dll* contains some user mode functions that enable user application to receive and send packets. To limit the packets lost', due to processing time by application, NPF buffers the incoming packets. *Packet.dll* provides common interfaces to packet capture device driver among different versions of "win32". Therefore, the user code can run independently of "win32" API version.

5.2 WinSEND Components

WinSEND is subdivided into three main components: WinSEND user interface, WinSEND service and WinSEND main classes. Figure 2 shows these components.

- WinSEND User Interface offers the possibility for the user to generate CGA address and save SEND parameters which is required for WinSEND service. The user can set the desired

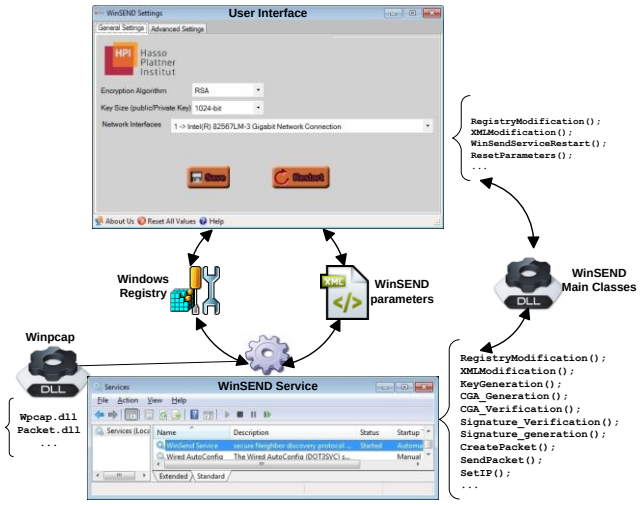

Figure 2. WinSEND main components and its relationships

security parameter "Sec" and determine which interface will be secured by WinSEND.

- WinSEND Service is the main part of this application. It does the functionalities of SEND in Windows. For instance, it implements CGA generation and verification and signature generation and verification. It also offers the other SEND options.
- WinSEND Main Classes (class library), contain shared classes that are called by both WinSEND user interface and WinSEND Service.

WinSEND parameters are stored in XML file format. This XML file contains SEND options such as the "subnet prefix", "Sec" value and key pairs (public/private keys). The reason for storing CGA parameters is to avoid CGA generation delays and to skip the CGA generation process while a node is connected to the same network. However, the address is not valid forever. It changes once the node joins a new subnet. For generating new CGA addresses, WinSEND reruns all CGA generation processes automatically. When the WinSEND service runs on a node for the first time, it calls "`KeyGeneration`", a function that generates key pairs in user-defined key size automatically. By default, the WinSEND uses RSA public key with 1024-bit length.

6. WinSEND IMPLEMENTATION AND TESTING

WinSEND integrates the security options to NDP messages. It generates CGA address for the node and verifies the received CGA addresses. For generating CGA, a security level "Sec" value should be specified through the WinSEND user interface. "Sec" value determines how it is difficult to break CGA by the brute-force attack. "Sec" value also affects the CGA generation time. A higher "Sec" value means longer time to generate CGA.

WinSEND implementation is tested in several experiments. One of these experiments is described below. The experiment is done in a test-bed which contains one router and two clients: A and B. Both A and B run WinSEND application. As soon as B is connected to the network, WinSEND service generates a CGA local link address and assigns it for the network adapter. Then it creates a Router Solicitation (RS) packet and sends it on the link. Once it receives the Router Advertisement (RA) message from the

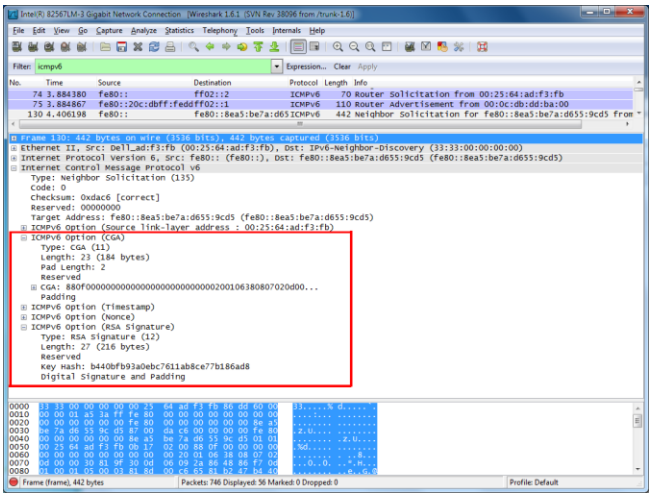

Figure 3. Wireshark screenshot of NS packet

router, WinSEND extract RA message to retrieve the needed data, such as "subnet prefix" to generate CGA global address and secure the response message with WinSEND.

Once the node generates its IPv6 address in a secure way by WinSEND, SEND options are attached to all Neighbor Discovery Protocol (NDP) messages. Figure 3 shows *Wireshark* screenshot of Neighbor Solicitation (NS) packet. All SEND options: *CGA*, *timestamp*, *Nonce*, and *RSA signature* are attached to the sent packet.

The receiver node should verify the CGA and the signature for the incoming packets. When client "A" receives NS message from client B, it verifies CGA. If the verification process successes, it accepts the traffic from "B". In addition to CGA verification, the receiver node needs to verify the signature which is carried by *RSA Signature Option*. If the signature verification fails, the node discards the packet silently and considers that it comes from a malicious node. In case the network has non-CGA nodes, the WinSEND can be configured to discard all packets sent by non-CGA nodes silently to avoid any possibility to generate Denial of service (DoS) attacks. Therefore, it is recommended to force all nodes to use CGA, otherwise it is not easy to distinguish between real and fake addresses.

7. CONCLUSION AND FUTURE WORK

Neighbor Discovery Protocol (NDP) is one of the most novel features in IPv6. In non-trusted environments, NDP is vulnerable to several attacks. The Secure Neighbor Discovery (SEND) protocol was proposed to counter most of the threats against NDP. However, SEND is not widely implemented. There are some SEND implementations in Linux and BSD, but no implantation for Windows. We implement WinSEND for Windows family, the most popular operating systems. WinSEND is developed in Microsoft .NET. It can be installed and integrated to Windows as a service to provide the functionalities of SEND. It has an easy user interface to enable the user to set the desired security parameters. WinSEND uses *Winpcap API* to have a direct access to raw sockets and bypass normal TCP/IP stack. To avoid the delay due to CGA generation algorithm, WinSEND stores CGA parameters in an XML file. These parameters are update when the node joins to a new subnet.

In this paper we put the basic principles for designing the implementation of SEND for Windows operating systems. As future work, we will continue the work to optimize the code. Currently, WinSEND is implemented to do the brute-force search to satisfy Hash2 condition [3] in CGA algorithm sequentially. In sequential computation, one instruction is executed per unit of time and WinSEND application uses only one CPU core of the computing device. If computing device has more than one core, the WinSEND is not able to use all the CPU capacity. However, the CGA is computationally heavy and it can take long time especially for high "Sec" value. In some cases, it is better to invest all the CPU capacity to finish the CGA computation as fast as possible. Consequently, we are working to offer WinSEND with parallel computational mode.

8. REFERENCES

[1] Nikander, P., Kempf, J., and Nordmark E., "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[2] Arkko, J., Kempf, J., Zill, B., and Nikander, P., "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[3] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005. Updated by RFCs 4581, 4982.

[4] Microsoft TechNet, IPv6 Security Considerations and Recommendations, http://technet.microsoft.com/en-us/library/bb726956, 2011.

[5] OS Platform Statistics, http://www.w3schools.com/browsers/browsers_os.asp, 2011.

[6] Narten, T., Nordmark, E., Simpson, W., and Soliman, H., "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[7] Thomson, S., Narten, T., and Jinmei, C., "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[8] Koodli, R., Ed., "Mobile IPv6 Fast Handovers", RFC 5568, July 2009.

[9] Narten, T., Draves, R., and Krishnan, S., "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[10] DoCoMo USA labs, http://www.docomolabs-usa.com/lab_opensource.html

[11] NDprotector, http://amnesiak.org/NDprotector/

[12] Chiu, S. and Gamess, E., "Easy-SEND: A Didactic Implementation of the Secure Neighbor Discovery Protocol for IPv6", Proceedings of the World Congress on Engineering and Computer Science, volume 1, 2009.

[13] Kukec, A. and Zeeb, B.A., "Native SeND kernel API for* BSD", 2010. http://people.freebsd.org/~anchie/SeND_AsiaBSDCon_2010.pdf

[14] ipv6-send-cga, http://code.google.com/p/ipv6-send-cga/

[15] Winpcap documentation, http://www.winpcap.org

[16] SIO_RCVALL Control Code, Build date: 21. 07. 2011, http://msdn.microsoft.com/enus/library/ee309610

[17] Windows Filtering Platform, http://msdn.microsoft.com/en-us/windows/hardware/gg463267.aspx

[18] Transport Driver Interface (TDI), 2011, http://msdn.microsoft.com/en-us/library/ms819740.aspx

[19] Smith, M., and Loguinov, D., "Enabling high-performance internet-wide measurements on windows". In PAM'10: Proc. of Passive and Active Measurement Conference, pages 121–130, Zurich, Switzerland, 2010.