Chapter 7
Necessary Standard for Providing Privacy and Security in IPv6 Networks

Hosnieh Rafiee
University of Potsdam, Germany

Christoph Meinel
University of Potsdam, Germany

ABSTRACT

Security and privacy have become important issues when dealing with Internet Protocol version 6 (IPv6) networks. On one hand, anonymity, which is related to privacy, makes it hard for current security systems to differentiate between legitimate users and illegitimate users, especially when the users need to be authenticated by those systems whose services they require. On the other hand, a lack of privacy exposes legitimate users to abuse, which can result from the information gained from privacy-related attacks. The current problems inherent within IPv6-enabled networks are due, in part, to the fact that there is no standard available telling companies about the current deficiencies that exist within IPv6 networks. The purpose of this chapter is to show a balance between the use of privacy and security, and to describe a framework that can offer the minimum standard requirement needed for providing security and privacy to IPv6 networks.

INTRODUCTION

IPv6 (Deering & Hinden, 1998) is the successor version of Internet Protocol version 4 (IPv4). These protocols are what makes communication possible across the Internet. Without the use of an IP address, it would not be possible to access the information located on many distributed repositories in many different locations across the world.

When the Internet Engineering Task Force (IETF) first proposed IPv6, the main assumption was to have a highly secure protocol which could support Internet Protocol Security (IPsec) (Kent, & Seo, 2005) natively, thus solving any problems dealing with end-to-end communications. However, this assumption proved to be unsupported. Unfortunately many implementers, and vendors alike, have not supported or have not

DOI: 10.4018/978-1-4666-6158-5.ch007

activated IPsec. One reason for this is because of the complexity that is involved in configuring IPsec and also the key management involved. This protocol thus remains with only the basic protection mechanisms in play so companies and governments are thus unwilling to widely deploy IPv6 or to replace their current IPv6 backbones with one making use of the IPsec protocol.

But the story does not end here. The development of a new address scheme for IPv6, which was necessitated by the IPv4 address space exhaustion, has led to new technology waves in which several volunteer experts and companies have become involved in looking for the flaws in IPv6. They are doing this in order to provide new security protocols to the users' of IPv6-enabled networks so that they can have the same security that exists when using IPv4, or maybe even higher.

The increased use of clouds and other repositories on the internet, in order to service and support many users at the same time, exposes this data to the vulnerabilities exploited by several types of privacy and security attacks. Today, we are living in an information technology world where governments and companies try to collect as much information as possible about their competitors. They do this so that they can overcome any possible threats from other governments or companies. Along with the risk of competitors, there are also individual/groups of attackers who are interested in obtaining the information available in those repositories so that they can misuse this information for their own criminal purposes. This is why, today, cyber attacks (attacks that are accomplished using cyber methods) are one of the main concerns of both societies and governments. Over the past 10 to 15 years multiple cyber attacks have occurred targeting both governmental agencies and private companies. The damage estimates are placed at more than $1 billion. Based on an official report from the United States Homeland Security Agency, in 2011 there were more than 106,000 incidents reported of which 5000 needed an urgent response.

Unfortunately, nobody knows yet, whether or not through the use of all security protocols, if IPv6 nodes will have the security level that is necessary to prevent several types of attacks or whether there are still many uncoverd flaws that might prevent the current services from being available to users or that can leak confidential information. What if there is a system that helps to show the current flaws in IPv6 networks and enables the user to do further tests to discover uncovered flaws in order to find a solution before these vulnerabilities can be used, like a tool in the hand of criminals? The remaining sections of this chapter are organized as follows:

Section 2: Introduce different meanings for privacy, anonymity and security in general and survey the current approaches in use in the application and network layers by explaining their issues and by introducing any available solutions.
Section 3: Explains some of the possible attacks in use across the internet.
Section 4: Surveys the currently available tools
Section 5: Introduces a system which can be used as a basic consultant system
Section 6: Explains the security recommendations for use in IPv6 networks and gives a basic security requirement.
Section 7: Summarizes this chapter.

PRIVACY, SECURITY, ANONYMITY, AND THEIR CONFLICTS

It is not easy to come up with a unique definition of privacy, security and anonymity. This is because some entities (such as users' data that can be an IP address, bank information, a post on social networking website, medical data, etc.) that are considered confidential for one group of people may not be considered confidential for another group of people. In other words, people have different interpretations of what constitutes privacy.

This is why they are easily confused between the meaning of privacy, anonymity and security and they perceive them to be the same while in fact each is an entity unto itself. Privacy is the act of allowing users to choose what entities they want to share with others and what entities they want to keep from others. Security, on the other hand, is the ability to protect these entities and this is usually possible by the use of cryptographic approaches. Of course encryption of the users' information also protects their privacy. But this approach does not allow a user to choose what data he wants to share with others and what data he wants to keep private. Moreover, the gathering of a user's location information via their IP address, for security purposes, might infringe on their privacy. Anonymity is the act of hiding a node's real identity so that its IP address or his other identities will be unseen by the other nodes on the Internet. For example, to remain anonymous, a user can use a Virtual Private Network (VPN) or proxy software, such as Tor, so that his real IP address would remain hidden behind several other groups of trusted computers. In other words, the node would hide its identity by making use of intermediate groups of trusted nodes thus masking its real identity. But hiding the IP address, or the identity of a user, does not necessarily protect the users' information from prying eyes. This means that, if the data is not encrypted, an attacker is capable of obtaining users' information, and then backtracking, using this information to further invade a user's privacy.

In 1995, the European Union (EU) (EU DPD, 1995) attempted to define privacy and come up with the first Data Protection Directive (DPD). This was the first attempt made by governments to officially define privacy so that they could pass laws to protect it. Unfortunately, later, European countries had a different interpretation of this DPD and used their own interpretation as a base for their laws as applied to a user's privacy. So, some countries decided only to concern themselves with location based tracking, while other countries only worried about some of their users' data while still others looked for absolute privacy so they think that everything should be kept confidential. So again, in 2012, the EU proposed another DPD (EU DPD, 2012) that tried to unify the meaning of privacy. According to this proposal, "privacy consists of personal data that concerns any information relating to an individual, whether it relates to his or her private, professional or public life. It can be anything from a name, a photo, an email address, bank details, posts on social networking websites, medical information, or a computer's IP address".

Another attempt at defining privacy also took place in 2012 when a group of people at the IETF proposed a standard (Cooper et. al., 2013) in which they tried to unify the meaning of privacy and anonymity so that all people, wishing to make implementations while observing privacy, would have a document that they could use as a reference for insight while doing their work.

As explained earlier, privacy is a user's entity so it is not bound to any particular Open Systems Interconnections (OSI) layer but some layers might have a stronger impact on a user's privacy than others. Two examples of these layers are the application layer and the network layer. In the next subsections we will explain what constitutes privacy in these two layers

Privacy in the Application Layer

Most of the user's confidential data, that is used by particular applications or is stored in some databases, reside then in the application layer. This data then might be accessible, over the internet, through the use of the user friendly name of your computer, called the Domain Name Service (DNS) name (Mockapetris, 1987). Since people are willing to use the online storage for easy accessibility to their data from everywhere, attackers are also willing to try to find ways to gain access to this confidential data. The data transferred across the internet can be in plaintext or encrypted. So when

the data is not encrypted and the user uses unsafe networks, or when it is easy to obtain a user's computer identity (that can be the DNS name of his computer), then there is a possibility that an attacker will eavesdrop on the network in order to obtain the information that the user is not willing to share with him. There are currently several approaches that can be used in the application layer to make data anonymous. These approaches are more focused on processing data in a way that removes any confidential data from them, thereby making it harder for an attacker to identify the owner of this data. The confidential information is removed from the data through the use of different algorithms. The work of Truta and Vinay (2006) provides an example of privacy protection used to protect medical data. They use k-anonymity algorithms where their algorithm categorizes and sub categorizes data into different classes based on similar properties, or based on the level of damage they might inflict on a user's health, etc. Then they try to group each category through the use of symbols, such as greater or smaller. They also remove any addresses or names from the data that would aid the attacker in identifying the data's owner. From the output of their algorithm you might obtain something like x number of people who have an age greater than 20 and less than 35 in the area of *secret_area, and* are under the category of *secret_letters_for-group-category*. In this case, this information might only be useful for statistical purposes.

The other attempt at providing privacy in the application layer has to do with giving users the ability to choose or hide their data on their public website, such as their profile. Google and Facebook are two big companies that observe user privacy using this approach. This allows users to also share their data with a group of confidential users and not the public.

However useful this might prove to be, there are, unfortunately, some companies, where people store their data, that can be easily hacked or that don't provide the high degree of protection that is necessary to prevent this data from intentionally being exposed to governments or other competing companies. Some companies might even store their confidential data on some public websites

Privacy in the Network Layer

One of the important protocols used in the network layer is the IP. IP addresses form the core of the addressing scheme used across computer networks to identify a user with his computer or to identify different devices. As explained in a prior section, a user might use the DNS name of a computer to access an entity. But this DNS name does not have any meaning without its mapping to an IP address. This means that any DNS name is mapped to one or several IP addresses. The IP address can thus provide an attacker with the means of finding the location of a user and then to track his movements over the internet. By knowing the user's location, he might be able to accomplish other criminal acts. If he knows when the user is at work, he can plan to rob his house when he isn't there. Another example would be to track very important politicians in order to carry out some form of terrorism. However, location tracking is not always done for criminal purposes as it may be done solely for advertising or security purposes. For instance, when you take a trip to an unfamiliar place, it is nice to have Google Maps available to show you where restaurants and shopping centers are located near where you are staying. It does this by storing your home IP address (the IP address of where you usually use to connect to the Google server) and comparing it to the new IP address that you just obtained from the server where you are staying. By doing this, they can also prevent unauthorized access to your account on their server by trying to verify you.

Knowing a user's IP address can lead to obtaining a user's confidential information. Especially if the user always uses the same IP address, or does not encrypt his data, or connects using an unsecure network. So this can allow an attacker

to gain access to this user information during its transfer across the network, which could result in harm to a user's privacy. It was for this reason that some research work (Silva, Dias, & Ricardo) was conducted to address the issue of anonymity in the network layer by hiding the user's identity, his IP address. This would give users the ability to anonymously surf the Internet or anonymously access websites. One example might be by encrypting the payload and then submitting it through many different intermediate nodes. The drawback to using this process is that the receiver is known to all communicating nodes and may also be well known to the attacker. Another example would be the use of a VPN, or other single entity controlled solution, i.e., using servers that are owned by entities, such as companies, to provide encryption and protection for the nodes. However, if this single node is compromised, then this will have an effect on the privacy of all nodes that used that node to start their VPN connection. Generally, the approaches just explained will cause delays or will end up being a single point of failure.

There are currently some countries, like Germany, that enforce their Internet Service Providers (ISPs) to change the range of IPv4 addresses in use once per week in order to protect their users' privacy by disallowing an attacker to track their users or to learn of their exact location. This range of IP address can be from a different location of the country.

The Privacy Extension standard RFC (Narten, Draves, & Krishnan, 2007) is the first attempt in IPv6 enabled network to observe privacy by generating a temporary Interface ID (IID). An IID consists of the 64 rightmost bits of the 128 bit IPv6 address. The 64 leftmost bits of an IPv6 address represent the subnet prefix and the user's network. The IPv6 address is in hexadecimal format. For example, if the IPv6 address of a computer is *2020:126:897:abc:ef2:8bc:683:a741* then the subnet prefix is *2020:126:897:abc* and the IID is *ef2:8bc:683:a741*. The Privacy Extension uses two different approaches for the generation of the IID portion of an IPv6 address, which is not based on a Media Access Control (MAC) address. A MAC address is a 48- bit hexadecimal number used by network adapters to uniquely identify themselves on a Local Area Network (LAN). Narten et al. (2007) assign a short lifetime to these temporary IIDs so they are not valid for long periods of time. Unfortunately this approach does have some drawbacks that afford attackers the ability to track users across networks. The list of these issues is as follows (Rafiee & Meinel, 2013):

- When a node joins a new network with a different subnet prefix, if the option in the router advertisement tells the node to extend the lifetime of its address, and if the maximum lifetime of that address has not been reached, then the node will keep its current IID without generating a new one.
- The node may still respond to requests from other nodes using the IID that was generated based on the MAC address. This can happen because this mechanism prompts the node to generate its public address based on a MAC address. There are different types of subnet prefixes: link-local subnet prefix, i.e.,the value of fe08:: and global subnet prefix, i.e., any subnet prefixes that are obtained from a router or make use of other servers, such as a Dynamic Host Configuration Protocol version 6 (DHCPv6) (Droms et al., 2003). The public addresses are the addresses that make use of a global subnet prefix. These addresses might also be defined on DNS servers or have associated DNS records.
- Another problem can occur when the node cuts its current connections with other nodes because the maximum lifetime for this IID has expired. In general the preferred lifetime is 1 day and the maximum lifetime is one week.
- Nodes may require a stable storage area in which to store both the history and the currently generated IID. This is done to preclude the use of an already used value.

If there is no stable storage area available, and the node does not use a good randomization algorithm, then the node may be unable to make use of a greatly randomized IID.

Stable Privacy Enhanced IID Generation (Gont, 2013) was proposed in order to address some issues that exist with the current IID generation mechanisms in use today. It uses a pseudo random function *F()*, and also some other parameters, as an input to this function thus enabling a node to generate a unique IID, which would be the same for the same subnet prefix, but will change with different subnet prefixes. This approach can significantly decrease the possibility of scanning attacks making it dissimilar to the approach that is based on MAC as it will not have the same IID in different networks. A problem with this approach, though, is that the node generates the same IID for the same subnet prefix and keeps it for as long as the subnet prefix is valid. This means that once the attacker finds the node's IP address, and if the node is fixed in one network, then the attacker will have enough time to try to gather as much confidential information as possible during the time that the subnet prefix is valid. In real life, the subnet prefixes are not frequently changed and they can be valid for several months to years. This is true for all countries except Germany, as was mentioned earlier with regard to IPv4 networks, which I assume will be same for IPv6 networks.

Router Advertisement (RA) based Privacy (Rafiee & Meinel, 2013) is a possible solution to the problem of privacy in the network layer. It tries to address the existing problems with the Privacy Extension and Stable Privacy Enhanced approaches make use of a randomized algorithm which is not based on the MAC address and they also force the node to change its IID whenever it receives a new router advertisement message. Dissimilar to Stable Privacy Enhanced IID, it will keep its IID for a short period of time for each subnet prefix. Therefore, if the user's identity is exposed to an attacker, the attacker probably won't have enough time to track this node or to obtain confidential information.

MECHANISMS USED TO ATTACK IPV6 ENABLED NETWORKS

There are many possibilities for attacks' classifications. Many attacks may make use of similar techniques to attain their goals. This means their way of doing it almost is similar, but their purpose might be different, or they might have a similar purpose as well. For accomplishing these attacks, one might need to misuse a/many intermediate services. The attacks might have an effect on both the privacy and security of the users or may only harm the privacy or security of users. The attack might be easy to detect or hard to detect depending on the nature of this attack. There is currently some work being done to try to detect attacks and then drawing a graph that compares attack patterns to similar attacking scenarios that are available in their database (Roschke, Cheng & Meinel, 2010). Here we try to categorize these attacks based on the impact that the attack has on the entities' privacy and security.

User's Identity Detection

The attacks in this category can be divided into two subcategories: passive and active.

Passive Detection

A passive attack is one where the attacker does not perform any process that involves the sending of a lot of packets to the target network, which could enable the security system, in target networks, to detect his malicious behavior. One of these passive approaches is the reconnaissance of target networks in order to obtain general information about the network and the nodes (any host in this network such a router a computer, etc.) in

that network. This is usually the first step that an attacker would execute. Some information that might be obtained through the use of this step is the information about firewalls, subnet prefixes used in this network, and general information about the type of organization/company or the place he wants to attack. If during this phase the attacker is able to obtain any IP addresses or nodes' information, such as the OS being used, servers, routers, etc., then as we explained earlier, finding an IP address of a node might lead to location tracking or leakage of the confidential information. This is because an attacker might be able to eavesdrop (silently listening to the network) all/part of the packets sent/received to/from a victim node in the target network. This might give this attacker the ability to gather any confidential information about the user of this node. Usually this attack is successful if there is no security mechanism in place for protecting a user's data. This is why, this phase of an attack is more concerned with a user's privacy. If the attacker is in the same network as other nodes, then he is able to change his network adapter status to promiscuous mode and eavesdrop on all packets being transferred in this network. This allows him also to obtain the other nodes' IP addresses, especially in an IPv6 network, where by default, Neighbor Discovery Protocol (NDP) (Narten et al., 2007) is active. The use of NDP is proposed for IPv6 networks to ease the IP address configuration process. Nodes in the network use 5 different types of Internet Control Message Protocol version 6 (ICMPv6) for the purpose of configuring their IP address and detecting neighboring nodes in this network.

Active Detection

The attacks in this category are involved with actively initiating processes in order to obtain a user's data. This means that these attacks are also involved with invading a user's privacy. In order to expand the information obtain from the Reconnaissance phase, one might need to scan the target network. This allows an attacker to obtain the IP addresses of the nodes. This step concerns privacy more than security. In IPv6 networks, because of the large address space (2^{128} unique IP addresses), scanning attacks are not really feasible but the attacker does have the possibility of sending and receiving multicast ICMPv6 echo messages in order to obtain the node's IP addresses. It is also possible to obtain node's IP addresses from DNS servers. One way to do this is by querying the DNS server for different domains. Another way is to obtain a copy of the DNS zone file during a zone transfer from a master to a slave DNS server. A zone is a portion of domain space that is authorized and administered by a primary name server and one or more secondary name servers. A name server can be a master or a slave. Master or primary name servers are the ones from which other name servers can transfer zone files. So attacking DNS servers is a means of obtaining IP addresses. After obtaining IP addresses, the attacker needs to detect the running services on each node in this network. Port scanning is one of the popular ways to detect the services running on each node in this network. This step is concerned more with a user's security than his privacy. This step is performed to enable an attacker to later look for security flaws in target services so that, later, he will be able to compromise the node and access the user's data.

Active Data Collection

Attacks in this category are divided into two subgroups; bugs in implementations or during configurations, and other attacks.

Bugs in Implementations or During Configurations

The standard documents defining Internet procedures, i.e, RFCs, usually explain what to implement but in most cases give the implementers the

choice of how to go about implementing. Human errors are a well-known problem in all computer systems. It is impossible for programmers to consider all of the possible conditions that their code must handle, so their code could cause problems during run-time. This is why implementers put so much effort into stress testing their implementations. One stress testing approach is fuzz testing. With this test random and unexpected data are used as inputs to the system being tested to find the conditions under which the system crashes or leaks information. Unfortunately attackers also make use of this approach to attack target protocols and services in IPv6 networks. They send packets containing invalid data, but with the correct checksum. By doing this they can crash the victim's system or force the protocol to leak information. The chance of this attack being successful increases when the protocol or service is not configured correctly. This too can lead to the leakage of information.

Other Attacks

Most attacks in this subgroup are a result of spoofing or the combination of spoofing and Distributed Denial of Service (DDoS) Attacks. Spoofing is the act of modifying data. DoS is the act of denying a node or a group of nodes access to their desired services. These attacks appear in different layers of the OSI model: network, application, etc. They occur using different formats and attack both privacy and security. Nikander, Kempf & Nordmark (2004) explained a list of NDP vulnerabilities. Hogg & Vyncke (2009) explain the attacks used against network layer protocols in IPv6 networks.

One example of these attacks is a smurfing attack where the attacker sends a lot of ICMPv6 messages using a spoofed source IP address, i.e., uses a victims' IP address as a source IP address and then sends it to a broadcast IP address. This results in all nodes in the network responding to that message by sending messages to the victim's node.

The Man in the Middle (MITM) attack is an example of a spoofing attack where the attacker tries to interrupt communications between two nodes. This is accomplished by presenting itself in a way that the two communicating nodes think that they are communicating directly with each other while their communication is actually via the attacker's node. This is a form of information leakage. Another spoofing attack occurs during the authentication process between one node and another node. The authentication is usually based on the source IP address or some other identity number existent in the packets. The attacker can spoof this data and easily impersonate itself as the other node to obtain unauthorized access to data or to make unauthorized modifications. Spoofing attacks might not be possible if the node where to use a security mechanism, but the chance for DDoS attacks actually increases when the node uses a security mechanism to protect its data. This is because the verification process takes more time than when not using a security mechanism and this time factor makes the node to susceptible to DDoS attacks.

AVAILABLE PENETRATION TESTING TOOLS

Currently there are several research groups and individual researchers actively involved in testing IPv6 networks for vulnerabilities in a variety of IPv6 protocol suites. These include addressing mechanisms, extension headers, fragmentations, tunneling or the dual stack networks (using IPv4 and IPv6 at the same time).

Single and Multi Function Tools

This section introduces some tools that are available for use in IPv6 networks. An example of such a tool for this endeavor is The Hacker Choice (THC)[1] Attack Suite, which was created by and is supported by some individual researchers. In

spite of being considered one of the most complete tools available on the Internet with which to initiate a large variety of attacks in IPv6 networks, one needs to call each attack separately, one by one, using the required parameters in order to execute these attacks. Another shortcoming of this tool is that no reports are prepared indicating whether or not the attacks were effective on the network and what hosts were vulnerable to what attacks. A third problem is that there is no good documentation for developers to use in writing additional code for their testing purposes. This is due to the fact that each tool might be developed by a different individual developer and so there is no consistency in the code. This means that the code might not be able to be used as a library.

There are other tools available, but they only have a single functionality and they thus can be used only for a specific purpose. For example, nmap (Lyon, 2009), halfscan6, etc. are used for scanning or for locating live nodes in IPv6 networks. SendIP[2], Scorpio[3], isic6[4], etc. are used to generate and manipulate IPv6 packets. Lecigne & Neville-Neil (2006) developed tools to test the IPv6 stack of FreeBSD during Google summer code. Later they extend their code to also evaluate IPv6 stack protocol in other Operating System (OS) such as Windows and Linux[5]. Some other tools are used just for testing a particular service. Web applications are one example of important target services in IPv6 networks. Ottow et al. (2012) explain the modifications required in order to be considered a valid current tool to be used for penetration testing of web applications based on checklists containing the most prevalent security issues. One example of a modification is related to a node scanning mechanism that should consider the IPv6 large address space and modify its scanning mechanism accordingly. Another issue of concern is that none of these tools can be used for scanning a network when DNS Security Extension (DNSSEC) (Arends et al., 2005) and NSEC3 are used. DNSSEC is an extension to the DNS used in the validation of DNS query operations. It verifies the authenticity and integrity of query results obtained from a signed zone. It introduces new Resource Records (RRs) (DNS servers store data in a specific structure called RR) for the authentication and verification purpose. One of these important RRs is the NSEC/3 (Laurie et al., 2008) which is used for NXDOMAIN responses. A NSEC RR includes the names of the successor and predecessor to a query response for the purpose of preventing replay attacks. All query responses in NSEC3 are the result of a SHA1 hash of the names so that it is not possible for the attacker to obtain the identities of the nodes by querying a DNS server. This is because the identities are not in plain text.

A FLEXIBLE FRAMEWORK

The tools currently available for penetration testing in IPv6 networks lack the ability to generate reports and do not focus on all available protocols used in IPv6 networks. They are also not capable of checking to see whether or not a combination of two or more protocols can be used to generate new attacks. One example relates to the vulnerability of the DNS server. Today, the nature of attacks used against DNS has changed to the point where attackers use this protocol, as a tool, to perform attacks against various other services available in the network. To address this problem we developed a new, flexible framework[6]. This framework not only detects the services in use in a network automatically but also performs attacks. It also generates vulnerability reports which administrators can use in resolving the detected security flaws in their network. This also provides for extensions from three perspectives: service discovery, attacks, and reports. This is a user space with an easy to use framework. It consists of two main components: attack and report/monitoring. They can be installed for use in two different linux-based nodes in IPv6 networks. Each of these components consists of a web interface and a backend library. The flex-

ibility of the framework gives users the ability of dragging and dropping their own scripts or to extend the framework by using other external scripts or codes that are available on the internet via the provided web interface. They can add and save their commands, which contain the required parameters, via this interface so that, later, the backend component can compile the external scripts and codes and save them in a directory and then save the path to the compiled code in its database. The framework will consider and run these scripts and codes during the next user-triggered network service execution. During the execution phase, the commands will pass through to the backend component in order to trigger a standard console command. The web interface is written in PHP.

New Node Discovery Algorithm

Attack components make use of two phases to process any attack: service discovery and execution of attacks. For service discovery, first it employs ICMPv6 multicast echo messages. In the case where it does not receive a response from the nodes in the targeted networks, it then uses a new algorithm, which uses the DNS to obtain information about the nodes having associated DNS records. To do this it executes the following steps:

1. If a zone transfer request is possible, then ask for the whole copy of the zone file. Otherwise go to step 2.
2. If the DNSSEC is not enabled, report a failure and stop scanning. Otherwise go to step 3.
3. If the DNS server uses NSEC, then start zone walking. Otherwise go to step 4. Zone walking is the process of sending different query requests to the DNS server on which DNSSEC is supported. This is done in order to retrieve the available legitimate DNS names of nodes, which then leads to being able to obtain the node's identities. Because in DNSSEC, the zone file is signed offline, it always responds to the query with the two closest legitimate names. If DNSSEC uses NSEC, then this response is in plain text. As an example an attacker might ask for an invalid name like "invalid.example.com". The DNS server would respond with two valid names, which are alphabetically close to that name, such as internal.example.com and nat.example.com.
4. If the DNS server uses NSEC3, start zone walking, gather hashed data, execute the dictionary attack offline, and then go to step 5.
5. If any records are left, do brute force attacks and repeat this step until no unknown record is available.

We gathered a list of popular domains and stored them in our database by using Alexa.com. We also tried to use a whole zone transfer request on these found domains. When our database was full of popular records, we ran the above algorithm for about 2 hours (Rafiee et al., 2013) on a computer with a 2.60 GHz CPU. We scanned one million domains. Among them, 55160 domains (5.5%) were not secure, which enabled us to obtain a copy of the zone file by using an AXFR query. This helped us to gather 1.43×10^6 RRs which we stored in our mysql database for later use in a dictionary attack against NSEC3. By using this approach, we were able to find the plain text for over 50% of the third level domains (example: www.google.com, www in this combination is the third level domain) in our target network on one public website.

The next phase of the attack component is to trigger all available attacking scripts for selected services for a certain period of time. This time was chosen by the user when this attack script was added to the framework. It also activates the report/monitoring component so that it is ready to monitor the network. The task of the report server

is to act as a basic consulting system that advises the user of any security and privacy flaws in his network. When the report/monitoring component receives a wake up message from the attack component, it will retrieve the list of attacks from the database of the attack component, and then starts monitoring the network to find out whether or not the attack was effective. This is done by sending probe messages or silently sniffing the network. For example, to see whether a node set its IP address based on a fake router advertisement message, the monitoring service sniffs the network to obtain the unsolicited neighbor advertisement message. To improve the reports, the user can easily add new scripts to detect the effectiveness of the attacks. For some types of attacks this phase may have problems in generating the report. This is because this framework does not install any external component or activate SNMP on any node in the target network. This does not mean that the ability to use this protocol does not exist. If the user wants to manually enable SNMP on any nodes in his network, then he can configure this framework for the SNMP protocol.

Vulnerability Detection Algorithm

To better find the vulnerabilities that exist in the target network, we need to consider the cases where one service can use, as a means of attack, other services, so as to be able to enhance the system with this algorithm. We also need to use machine learning approaches in order to automate the process for adding new protocols. The steps of the vulnerability Detection algorithm are as follows:

Store the Current Root Vulnerability of Existing Trees

In this step, we need to manually (or use any online databases) feed the system with all the vulnerabilities of known protocols. This is necessary in order to have a consistency of keywords used for all protocols. For example, if your NDP protocol is vulnerable to IP spoofing and message spoofing, then the keyword can be spoofing. This means that one should then use the same keyword for a spoofing vulnerability in all other protocols. In this step all the dependencies should also be specified. This means there are two variable to be consider as inputs to the system, i.e., L and D, where L is the list of vulnerabilities for a specific protocol, V is all available vulnerabilities for all protocols and D consists of the dependency lists.

$$L = \{\forall l \in V, \exists D \therefore l \rightarrow D\} \quad (1)$$

For example, if l is spoofing (keyword), then D is the array of vulnerabilities that spoofing can lead to, such as cache poisoning, unauthorized access to the system, MITM, etc. These are all dependent on spoofing.

Definition 1: TV where T represents a tree structure. To prove that it is a tree, we assume that T has n members and we define a matrix $M(T)_{n \times n}$ that specifies whether or not there is an edge (dependency) between two vulnerabilities in T. We initially set M(T) to zero. So,

$$\forall t_i \in T, \exists t_j, t_k \in T \text{ where } t_j, t_k \text{ are adjacents of } t_i \therefore M[t_j, t_k] \neq 0 \quad (2)$$

Adjacent vertexes are the vertexes that are directly connected to each other with an edge. This means neighboring vulnerabilities are dependent on t_i.

From Definition 1 we can generate a tree structure containing l as a root and many Ds as the leaves of this tree. The leaves might directly or indirectly connect to the l. The tree will be useful later as a report component that gives users more precise information about the vulnerabilities in the system. But for finding out whether or not the vulnerability of a service might be used as a

means of attacking other service, then only the root level vulnerabilities play an important role in determining that.

Execute an Attack on Other Services Using the Root Vulnerabilities of a Particular Service

For example, spoofing is the root attack when an attacker spoofs the source IP address in a DNS query message and then uses the victim node's IP address instead of his own IP address. Then he asks for a query that requires a large response. He can also ask for different queries. The DNS server will then respond to those queries. This means that the victim node will then, as a result of this, receive several large packets that were never requested. Thus the victim node will be kept busy processing those packets. This attack is applicable if there is no authentication during the DNS query process involving the DNS server and clients. DoS attacks are thought of as both an attack against privacy and security. It is considered as such because the user is not able to share his chosen information with others due to the fact that the attacker has prevented the running of any service on his node.

Evaluation of the Framework

We evaluated our framework by considering some factors such as the time it takes to uncover what services are in use in the network and what attacks are being perpetrated along with the time needed to generate the report describing said services and attacks. We ran our framework using a testbed of 17 nodes. As we were in the same network as the other nodes, we did not need to use our DNS based scanning algorithm. We could have easily used ICMPv6 echo messages. After we found the list of available nodes, we tried to call one of the external codes which can be used to give us the general information about the available services in the network. Our results showed that it takes the total of 5.99 seconds to find the list of nodes and to do the port scan for ssh and http.

The duration of time for the attacks is a variable time to be determined by the user and input using the framework's web interface. If the user chose a value of 40 seconds, then the duration of the attack would be 40 seconds. To enhance the framework by using more attacks, we implemented new attacks with which to evaluate the IPv6 protocols. These attacks entailed the use of fuzz mechanisms, evaluating DNS protocols such as multicast DNS (mDNS) (Cheshire, & Krochmal, 2013) and the evaluation of Mobile IPv6. Some instances of the new attacks are as follows:

Attacks Against Multicast DNS (mDNS)

DNS is one of the application layer services that uses an IP address. This is why the existing attack tools for DNS need some modification to support IPv6. mDNS is one of the new operations of DNS used in a local link in the absence of a unicast DNS server. Domain names using the multicast DNS end with *.local*. mDNS is used for finding printers or other shared folders of different OSs, or for checking the uniqueness of names used in local links. When a host joins a network that supports multicast DNS, it tries to set its local hostname, like *mydomain.local*, and then it sends a multicast DNS message to all nodes on that local link to see whether or not the chosen name is unique. We implemented a Man In The Middle (MITM) attack using C++ by extending the packet-manipulation-library[7]. Using a *Sniffer* object, provided by the packet-manipulation-library, the component analyzed all traffic on UDP port 5353. Each host name query (AAAA, i.e. IPv6 RRs) was stored in a map laid out to remember the questioner and the host name in question. Every time a DNS response was received by our framework, the map was checked for a matching answer. If found, the attack was carried out against the original questioner. For example, if nodes A and B want to communicate together, then node B

will ask the name of node A using mDNS in order to connect via its name. Node A picks up a name and tries to check the uniqueness of this name by sending a mDNS message. Our MITM components (in our framework) receive this message as well as does Node B. Our component can then spoof that message and claim to own that name. There is no security proposed for mDNS, so it is easy for this to happen. Or our component could wait for node A and B to start their connection. First the MITM component will send out unicast goodbye packets to B indicating that the original host name holder gave up his authority over the name (node A will not hold onto A's name anymore). Then the MITM component sends another spoof unicast mDNS message to node B claiming to be allowed to use name A and continues the communication and then redirects the communication to the real A. In this case it plays a MITM attack. According to the mDNS RFC, the node should not accept unicast messages. However, we discovered that all current mDNS implementations accept unicast messages, which meant that we were able to successfully execute our attacks.

Fuzz Attacks

Fuzz testing is one of the popular testing approaches used by industries. To better test an IPv6 protocol stack, we implemented fuzz attacks. The code is called by our framework, which is then responsible for the creation and sending of fuzzy IPv6 packets as well as monitoring the target machines. Our first approach falls into the category of random input generation. We used scapy to generate our own packets, as it allows for the generation of invalid packets, which is useful for the fuzz approach. Our experimental results show that many packets are being rejected in the early stages because of invalid formats. In some cases the victim node's crashed and we had to reboot the system. In order to improve our results we used grammar based fuzz mechanisms. Grammar based fuzz mechanisms are more precise. To facilitate the process of grammar based fuzzing we used the Peach Fuzzer framework[8]. On the day that we ran our code, we could not find any malfunction in nodes that accepted our IPv6 plain packets. However, as we did not have access to the IPv6 implementation stack, it was not easy to evaluate the target nodes' completely.

The last step, and the most important step for the framework, is the generation of reports. Based on our experiments, it takes an average of 9.77 seconds to generate a report about one protocol, like the NDP. This report varies from one protocol to another as it is dependent on many factors such as how fast the victim nodes respond to attacks, the duration of the attacks, and whether the report component needs to send a probe message to gather data from the nodes in the network, or whether that just sniffing will suffice.

SECURITY AND PRIVACY RECOMMENDATIONS

Enabling IPv6 might be a nightmare for many governments, companies or places who are still not sure about how to deploy this protocol without interrupting their running services or without the fear of privacy and security attacks. This is why many people felt that there was a need for a deployment guide for this protocol. To address this issue,the National Institute of Standards and Technology (NIST) (Frankel et al., 2010), with the help of many individual experts or companies, gathered a guideline, which supports most of the protocols used in IPv6 networks. However, even though this work is so promising, it unfortunately, despite the attempt to prepare a good document, contains several mistakes in descriptions of protocols and their protective capabilities. This is why, it cannot be considered a reliable source for people who are new to IPv6 and do not have any experience in this area. Usually these guidelines

are used by non-experienced people more than experienced ones. One example occurs in the security section and concerns the description of Cryptographically Generated Addresses (CGA) (Aura, 2005).

To address the concerns surrounding IPv6 deployment and to offer a minimum standard, we will cover some of the protocols used in IPv6 networks. Our coverage includes the DNS, Mobile IPv6, and the IPv6 protocol itself, such as IPv6 addressing schemes and the IPv6 dual stack mode.

DNS Security and Privacy Recommendations

The DNS is one of the fundamental protocols used by the internet. It allows for the translation of the user friendly names of nodes in the network to their IP addresses. This protocol only provides protection by use of a basic security mechanism which is authenticated based on the source IP address. DNS is vulnerable to many security flaws such as cache poisoning, spoofing, etc. (Atkins & Austein, 2004). Because the DNS is usually a public node, which must be accessible over the internet, privacy for this protocol does not make sense. But ignorance of security flaws might lead to exposing the identity of nodes in the targeted network, as was through the use of our framework in a prior section. In other word, the lack of security on this node could lead to serious privacy and security issues for other nodes. Some of the things that need to be considered for this protocol are as follows:

- Secure Authentication during zone transfer, DNS update (updating a/ record/s by a client to a DNS server) or resolver authentication (authentication of a DNS resolver during query response with clients or other resolvers). There are two solutions here; using DNSSEC with NSEC3 or using CGA-TSIG (Rafiee, Loewis & Meinel, 2013):
  - Use DNSSEC with NSEC3 to authorize zone transfers. To decrease the chance of a dictionary attack, the use of popular names is not recommended. The protocol also needs to pick up a random signed name as a response to an invalid name that is not available in the DNS server. This will decrease the chance of guessing the third level domains, as the two letters will not be close in alphabetical order.
  - Use CGA-TSIG when Secure Neighbor Discovery (SeND) (Arkko et al., 2005) is available in the network. This decreases the complexity of the use of DNSSEC and will also exhibit the same problem that exists in NSEC3.

Mobile IPv6 (MIPv6)

Mobile IPv6 (MIPv6) (Perkins, Johnson, & Arkko, 2011) was designed to allow users to move from one network to the other network without any interruption in their connections. There are only a few implementations of MIPv6. One of these implementations is UMIP[9], which is not a stable version (Based on our experiments carried on three Computers which support Debian). Like some other protocols that are susceptible to spoofing attacks, MIPv6 is also vulnerable to spoofing attacks. This is the reason why it is not recommended to use this protocol without having IPsec enabled or using some other security mechanism in its stead. The following recommendations made concerning the use of MIPv6:

Use SeND or some other security approach to configure IPv6 for a Mobile Node (MN). MN is a node, such as a laptop, etc. that can move from one place to another. UMIP does not support SeND. UMIP only supports radvd[10], which is a daemon used to send Router Advertisement (RA) messages, which allows nodes, that support NDP, to configure their IP addresses. The other

option for configuring the IP address is to use DHCPv6. However, it is not secure either. If an attacker is in a Home Network (the network where the MN first made its connection and started its communication with the other nodes, known as Correspondent Nodes. The Home Agent (HA) in this network maintains the status of the MN) or the Foreign Network (the network to where MN moved and expects to continue his communication with other nodes without interruption. The Foreign Agent (FA) in this network maintains the status of MN) so that he can then use a spoofing attack and play a MITM role. However, the chance of a MITM attack decreases when the node uses IPsec. This is because, by using IPsec, a secure channel is established between the MN and his communicating node.

In general, key management may pose a problem when using IPsec. This is because the administrator might need to manually exchange the keys between MN, HA, FA and/or CNs.

IPv6 Addressing Scheme

As explained earlier, NDP and DHCPv6 are two popular approaches used to configure the IPv6 address of nodes. Unfortunately DHCPv6 does not support any mechanism that would enable nodes to authenticate the DHCP servers or clients. This means it is vulnerable to spoofing attacks. For example, if a malicious node is inside a network, then it can claim to be a DHCP server. Nodes in this network will accept this DHCP server since they cannot distinguish between the legitimate and illegitimate DHCP servers. Some administrators try to limit the access to their DHCP-enabled network by including the MAC address of all legitimate clients on their DHCP access list. However, the attacker can spoof a MAC address of one of legitimate node and then initiate his attacks. NDP has the same problem. A malicious node in the network can claim to be a router and redirect all the traffic to his desired place. These are some recommendations for generating IP addresses:

- It is recommended that SeND be used, including the use of the CGA or a Simple Secure Addressing Scheme (SSAS) (Rafiee & Meinel, 2013) options. SSAS is faster than CGA and provides the node with a good level of security that is even higher than that of CGA.
- For privacy reasons, the IP addresses should only be valid for a short period of time.
- Using DHCPv6 is not recommended since there is no way for authorizing nodes. If this is a local network that no external users can access, the local network might be safe. In this case then, for privacy, DHCPv6 can make use of any random approach for the generation of IIDs.

Dual Stack

Dual stack is the state of concurrently using both IPv6 and IPv4 in parallel. It is the most preferable scenario during the coexistence of both IPv4 and IPv6. This means the nodes can process packets send/receive to/from other nodes with IPv4 or IPv6 contents simultaneously. This is done due to the fact that we are in a migration period from IPv4 to IPv6. This means some networks might support only IPv4 and some both. One problem with dual stack is the fact that we need to configure all services with both IPv4 and IPv6 addresses, but unfortunately IPv4 addresses are nearly exhausted. One solution is to use Network Address Translation (NAT) in IPv4 networks. According to the IETF definition "NATs are used to interconnect a private network consisting of unregistered IP addresses with a global IP network using limited number of registered IP addresses". However, NAT creates a problem for end-to-end communication. Recommendations for the use of the dual stack relate to the consideration of security issues within IPv4 and IPv6 enabled networks. This means that if one wants to have a secure network, and to partially support IPv6, then it needs to ensure

that all the security protocols are installed in the target network. This can be done by using one of the tools that penetrate IPv6 networks, and then using the tools that detect IPv4 vulnerabilities. The problem that many people ignore is the case where the network only supports IPv4. They are not aware that, by default, the current OS supports IPv6 and in most cases they are activated automatically during the installation of the OS. One example of these protocols, in IPv6 suites, is NDP. In the Windows OS, this feature, by default, is activated. This is why, in dual stack mode, or in IPv6 mode only, tools for both protocols should be used to penetrate the networks and uncover possible flaws.

CONCLUSION

In this chapter we explained the different meaning given to privacy, security, and anonymity from different points of view. We also covered the problems affecting both security and privacy in the network layer and application layer. We then categorized attacks based on their impact on privacy and security. We evaluated different available IPv6 tools used for penetration testing, we introduced our own flexible framework that can make use of all the external tools and finally we enhanced this framework with new attacks. We evaluated our framework by considering the time required to scan the network, execute attacks, and generate reports. This framework can be used as a basic consulting system, which helps people learn of the flaws in their network and then shows them what to install as a security protocol to safeguard it. Our framework makes use of a new vulnerability algorithm. We also introduced a new scanning algorithm which makes use of DNSSEC and NSEC3. Finally we explain our recommendation for some of the protocols in IPv6 networks to show how one can observe privacy and security in this areana.

REFERENCES

Arends, R., Austein, R., Larson, M., Massey, D., & Rose, S. (2005, March). DNS Security Introduction and Requirements. *RFC*. Retrieved March 2005, from http://www.ietf.org/rfc/rfc4033.txt

Arkko, J., Kempf, J., Zill, B., & Nikander, P. (2005, March). *SEcure Neighbor Discovery (SEND)*. Retrieved from http://tools.ietf.org/html/rfc3971

Atkins, D., & Austein, R. (2004, August). Threat Analysis of the Domain Name System (DNS). *RFC*. Retrieved August 2004, from http://www.ietf.org/rfc/rfc3833.txt

Aura, T. (2005, March). *Cryptographically Generated Addresses (CGA)*. Retrieved from http://www.ietf.org/rfc/rfc3972.txt

Cheshire, S., & Krochmal, M. (2013, February). *Multicast DNS*. Retrieved from http://tools.ietf.org/html/rfc6762

Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., & Smith, R. (2013, July). *Privacy Considerations for Internet Protocols*. Retrieved from http://tools.ietf.org/html/rfc6973

Deering, S., & Hinden, R. (1998, December). Internet Protocol, Version 6 (IPv6) Specification. *RFC*. Retrieved from http://www.ietf.org/rfc/rfc2460.txt

Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., & Carney, M. (2003, July). Dynamic Host Configuration Protocol for IPv6 (DHCPv6). *RFC*. Retrieved July 2003, from http://www.ietf.org/rfc/rfc3315.txt

EU DPD. (1995). *European Union Data Protection Directive: Processing of Personal Data and on the free movement of such data*. Retrieved from http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML

EU DPD. (2012). *European Union Data Protection Directive*. Retrieved from http://europa.eu/rapid/press-release_IP-12-46_en.htm?locale=en

Frankel, S., Graveman, R., Pearce, J., & Rooks, M. (2010, December). *Guidelines for the Secure Deployment of IPv6*. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-119/sp800-119.pdf

Gont, F. (2013). *A method for Generating Stable Privacy-Enhanced Addresses with IPv6 Stateless Address Autoconfiguration (SLAAC)*. Retrieved from http://tools.ietf.org/html/draft-ietf-6man-stable-privacy-addresses (Work In Progress)

Hogg, S., & Vyncke, E. (2009). *IPv6 Security*. Cisco Press.

Kent, S., & Seo, K. (2005, December). Security Architecture for the Internet Protocol. *RFC*. Retrieved December 2005, from http://www.ietf.org/rfc/rfc4301.txt

Laurie, D., Sisson, G., Arends, R., & Blacka, D. (2008, March). *DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. Retrieved from http://tools.ietf.org/html/rfc5155

Lecigne, C., & Neville-Neil, G. V. (2006, August). *Walking through FreeBSD IPv6 stack*. Retrieved from http://clem1.be/gimme/ipv6sec.pdf

Lyon, J. F. (2009). *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Academic Press.

Mockapetris, P. (1987, November). *Domain Names - Implementation and specification*. Retrieved from http://tools.ietf.org/html/rfc1035

Narten, T., Draves, R., & Krishnan, S. (2007, September). *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. Retrieved from http://tools.ietf.org/html/rfc4941

Narten, T., Nordmark, E., Simpson, W., & Soliman, H. (2007, September). Neighbor Discovery for IP version 6 (IPv6). *RFC*. Retrieved September 2007, from http://www.ietf.org/rfc/rfc4861.txt

Nikander, R., Kempf, J., & Nordmark, E. (2004, May). *IPv6 Neighbor Discovery (ND) Trust Models and Threats*. Retrieved from http://tools.ietf.org/html/rfc3756

Ottow, C., Vliet, F. V., Boer, P. D., & Pras, A. (2012). The Impact of Ipv6 on Penetration Testing. Springer.

Perkins, C., Johnson, D., & Arkko, J. (2011, July). *Mobility Support in IPv6*. Retrieved from http://tools.ietf.org/html/rfc6275

Rafiee, H., Loewis, M. V., & Meinel, C. (2013). Challenges and Solutions for DNS Security in IPv6. In Architectures and Protocols for Secure Information Technology Infrastructures. Hershey, PA: IGI Global. DOI: doi:10.4018/978-1-4666-4514-1.ch006

Rafiee, H., & Meinel, C. (2013). *Router Advertisement based privacy extension in IPv6 autoconfiguration*. Retrieved from http://tools.ietf.org/html/draft-rafiee-6man-ra-privacy

Rafiee, H., & Meinel, C. (2013). SSAS: A Simple Secure Addressing Scheme for IPv6 AutoConfiguration. In *Proceedings of the 11th IEEE International Conference on Privacy, Security and Trust (PST)*. IEEE.

Rafiee, H., Mueller, C., Niemeier, L., Streek, J., Sterz, C., & Meinel, C. (2013). *A Flexible Framework For Detecting IPv6 Vulnerabilities*. Submitted to ACM Conference.

Roschke, S., Cheng, F., & Meinel, C. (2010). Using Vulnerability Information and Attack Graphs for Intrusion Detection. In *Proceedings of the 6th International Conference on Information Assurance and Security (IEEE)*, (pp. 68 - 73). IEEE. doi: doi:10.1109/ISIAS.2010.5604041

Silva, P. M., Dias, J., & Ricardo, M. (n.d.). *Survey on Privacy Solutions at the Network Layer: Terminology, Fundamentals and Classification*. Retrieved from http://paginas.fe.up.pt/~prodei/dsie11/images/pdfs/s6-4.pdf

Truta, T. M., & Vinay, B. (2006). Privacy Protection: p-Sensitive k-Anonymity Property. In *Proceedings of the 22nd International Conference on Data Engineering workshops*. IEEE.

ENDNOTES

1 http://www.thc.org/thc-ipv6/
2 http://snad.ncsl.nist.gov/ipv6/sendip.html
3 http://www.secdev.org/projects/scapy/
4 http://isic.sourceforge.net/
5 http://clem1.be/ipv6-attacks/
6 http://www.hpi.uni-potsdam.de/meinel/security_tech/ipv6_security/ipv6ssl.html
7 https://code.google.com/p/packet-manipulation-library
8 http://peachfuzzer.com
9 http://www.umip.org/
10 http://www.litech.org/radvd/