Published as : Hosnieh Rafiee and Christoph Meinel. "Privacy and Security in IPv6 Networks: Challenges and Possible Solution". The 6th International Conference on Security of Information and Networks (SIN2013), ACM, November 26 - 28, 2013 Aksaray, Turkey

# Privacy and Security in IPv6 Networks: Challenges and Possible Solutions

Hosnieh Rafiee
Rafiee{at}hpi.uni-potsdam.de

Christoph Meinel
Meinel{at}hpi.uni-potsdam.de

Hasso Plattner Institute, University of Potsdam
P.O. Box 900460, 14440 Potsdam, Germany

## ABSTRACT

Privacy is a very important element in every one's everyday life. Most users would not like to have their data exposed to other people on the Internet. The initial approach used for attacking a user's privacy and security is done by scanning the nodes on a network. This gives an attacker the ability to obtain the IP addresses in use by this node so that this information can then be used to initiate further attacks against this node, such as tracking them via their IP address across the networks, and then, later correlating the user's activities with his IP address. The first attempt by the Internet Engineering Task Force (IETF) to protect a user's privacy was defined in the Privacy Extension RFC [13]. Unfortunately this RFC has some deficiencies which makes its use vulnerable to privacy related attacks. To address this problem, and solve the deficiencies that exist with the use of this RFC, we introduce our new algorithm, which not only maintains a node's lifetime, but also provides a user with a method for randomized Interface ID (IID) generations.

## Categories and Subject Descriptors

H.4 [**Privacy and Security**]: Authentication and Reliability

## Keywords

privacy, lifetime of IID, IID generation, Privacy model in network layer

## 1. INTRODUCTION

Today many people talk about Internet privacy and security, particularly since the deployment of Internet Protocol version 6 (IPv6), which is the next generation of Internet Protocol replacing IPv4. But trying to come up with the ways and means of actually providing for privacy is a daunting task. Just trying to define the difference between privacy and security becomes a difficult task as they are so closely related. This is why the meaning of privacy is not definitive among the experts because there is no fixed dividing line (time-frame) between privacy and security.

The implementation of IPv6 has created many more issues than it has solved. This privacy issue is even more complicated than it was before because of the use of many new techniques for storing data or sharing data among many people at the same time. Much of this data is confidential, and the users may just want to share it with a select few, but in today's environment they are leaving themselves open to privacy attacks that would result in a wider dissemination of their data than they would like. Places such as those offering cloud computing, social websites, and completing transactions on-line are just a few of the areas vulnerable to privacy attacks.

The current mechanism used to prevent node tracking, and as a result to protect a user's privacy in the network layer, is to change the node's Interface ID (IID) frequently. The IID consists of the 64 rightmost bits of the 128 bit portion of their IPv6 address (then call it a temporary IP address). When this is done nodes will avoid the leakage of confidential information about themselves. Even though this appears to be a promising approach, the generation of an IID based on the MAC (hardware identities) address, and using this along with the temporary IP addresses [13], increases the chances of incurring a privacy related attack.

In this paper we introduce our algorithms that address the problem inherent in the use of the privacy extension RFC. The remainder of this paper is organized as follows: Section 2 explains the definition of Privacy and Security. Section 3 explains privacy and its Correlation to IPv6. It also explains the deficiencies inherent in the current IID generation mechanisms. Section 4 introduces our algorithm for maintaining privacy. Section 5 evaluates our proposed algorithm. Section 6 explains future works. Section 7 summarizes our conclusions.

## 2. WHAT IS PRIVACY? WHAT IS SECURITY?

Privacy and security have a close relationship. Privacy, simply stated, is the act of allowing a user to choose which data he wants to make available to others, or which data he wants to keep from others. Security, on the other hand, is the ability to protect a users' data, or to keep a users' data confidential. There are times, however, where one will have to be sacrificed for the sake of the other. One example could be where a company wants to control who logs into its server. A company does this to disallow an unauthorized IP access to their customers' data. This company will thus need to gather the location information, based on the IP

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIN'13, November 26-28, 2013, Aksaray, Turkey. Copyright 2013 ACM 978-1-4503-2498-4/00/10 ...$15.00.

Published as : Hosnieh Rafiee and Christoph Meinel. "Privacy and Security in IPv6 Networks: Challenges and Possible Solution". The 6th International Conference on Security of Information and Networks (SIN2013), ACM, November 26 - 28, 2013 Aksaray, Turkey

address, of the people who access their server, even though the user might not like this, and this might prove harmful to their privacy. But in many cases, when cryptography or other approaches are used to protect the content of the data, it is not only securing the data but also providing privacy. This shows that privacy and security are moving in the same direction. There always seems to be a problem in trying to define privacy. This is because people have different interpretations for what constitutes privacy. In an attempt to try to provide a universal meaning for what constitutes privacy, the European Union (EU) developed, in 1995, the first Data Protection Directive (DPD) [10]. In spite of this attempt for a universal directive, European countries decided to enact their own national laws dealing with privacy based on various views, such as disallowing location based tracking, or for providing only partial protection for a customers' data, or for insuring everything that seems to be personal in nature with thorough protection. This is why the European Commission has tried to improve and integrate the implementation of a new privacy law. Based on the recent EU Commission proposal [17], 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.

## 3. PRIVACY AND ITS CORRELATION TO IPV6

### 3.1 What is Internet Protocol version 6 (IPv6)

IPv4 is being replaced by IPv6 [7], the next generation of Internet Protocol. The lack of available IP addresses in IPv4 necessitated the inception of a new protocol which would address this issue and this new protocol was named IPv6. IPv6 was designed with a much larger address space thus accommodating a horrifically large number of addresses. The addresses in IPv6 use a hexadecimal format in contrast to the numeric format of IPv4. Autoconfiguration is the mechanism used that makes nodes capable of generating their IP address as soon as they join a new network. There are two different mechanisms used in IPv6 autoconfiguration: Dynamic Host Configuration Protocol version 6 (DHCPv6)[8] and Neighbor Discovery Protocol (NDP)[14, 18].

#### Dynamic Host Configuration Protocol version 6 (DHCPv6)

DHCPv6 is used in IPv6 for what DHCP is used for in IPv4. This protocol requires a certain amount of human intervention with regard to the installation and administration of the DHCPv6 servers.

#### Neighbor Discovery Protocol (NDP)

NDP refers to a combination of two mechanisms in IPv6: StateLess Address AutoConfiguration (SLAAC)[18] and Neighbor Discovery[14]. Together they enhance the node with several features, one of which is allowing the node to discover its neighboring nodes within a network. This mechanism is new to IPv6 and did not exist with IPv4. With this mechanism, in contrast to DHCPv6, there is no need for human intervention in the IP address configuration process. But there does exist a large security issue when this mechanism is used. The problem is that the default assumption made, when using NDP, is that all nodes in local networks are trusted. It is for this reason that this mechanism does not provide security protection to the nodes. To address this security issue a new feature was proposed, Secure Neighbor Discovery (SeND)[2], which provides the necessary protection by adding 4 options to NDP messages. These options are timestamp, nonce, RSA signature, and Cryptographically Generated Addresses (CGA)[4]

### 3.2 Current Problems Concerning IPv6 Privacy

#### Privacy Extension and its drawbacks

IPv6 addresses consist of two parts; the subnet prefix, which is the 64 leftmost bits of the IPv6 address, and the Interface ID (IID), which is the 64 rightmost bits of the IPv6 address[12]. A standard used in the generation of IPv6 IIDs is called the Extended Unique Identifier (EUI-64). EUI-64s are generated by the concatenation of an Organizationally Unique Identifier (OUI), assigned by the IEEE Registration Authority (IEEE RA), with the Extension Identifier, assigned by the hardware manufacturer. Then bits u and g, or bits 7 and 8, from the leftmost bytes of the EUI are set to one and the entity that results is the IID. If the OUI is 24 bits, and the extension identifier is 24 bits, then a hexadecimal value of 0xFFFE will be inserted between these two values.
The drawback to the generation of an IID based on the MAC address, or EUI-64, is that the node will generate the same IID whenever it joins a new network. This fact makes it vulnerable to privacy related attacks. One problem with this approach is that it makes it easy for someone to track the node's location based on its IP address. Another problem with this approach is that it gives an attacker enough time to gather confidential information about the user because he can follow this node across several networks.
The first attempt at protecting privacy was defined in RFC 4941. This RFC explains two possible approaches for IID generation. The first approach makes use of available stable storage. In this approach, first, the node chooses the last IID value from history. If the history is empty, or there is no stable storage available, the node will choose a random value. Then the IID retrieved from history, or the random value, will be concatenated with the EUI generated as explained in RFC 4941. Then the node will execute MD5 on the resulting value. It will then use the 64 leftmost bits from the MD5 digest and will compare this data to the list of reserved and currently assigned IIDs. If any matches are found, it will save the 64 rightmost bits of the digest, in history, and will again repeat the algorithm. If no matches are found, it will set bits u and g to one and use this as the final IID. The following items explain the drawbacks to using this privacy extension mechanism:

- 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

Published as : Hosnieh Rafiee and Christoph Meinel. "Privacy and Security in IPv6 Networks: Challenges and Possible Solution". The 6th International Conference on Security of Information and Networks (SIN2013), ACM, November 26 - 28, 2013 Aksaray, Turkey

MAC address. This can happen because this mechanism does not force a node to only use the IID generated by use of this approach. It also promotes the node to generate its public addresses based on the MAC address. There are two types of IP addresses; public/global and local. Public addresses are the addresses that are used for accessing other resources via the Internet or in another network. Local addresses are only valid in the local link and are not used for routing purposes. The local addresses start with the local subnet prefix.

- 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 Algorithm

Stable Privacy Enhanced IID Generation [11] 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. Using this mechanism a node first picks up a $secret_{key}$, which is a pseudo random number, and then executes the pseudo random function R=F(Prefix, $Net_{Iface}$, $Network_{ID}$, $DAD_{Counter}$, $secret_{key}$) on some inputs, where the prefix is the router advertisement message prefix and $Net_{Iface}$ is an implementation-dependent stable identifier associated with the network interface. $Network_{ID}$ consists of some network specific data that identifies the subnet to which this interface is attached. For example, one might use the Service Set Identifier (SSID). The $DAD_{Counter}$ will be incremented during the Duplicate Address Detection (DAD) process after an occurrence of a collision. The node will then take the 64 leftmost bits of $R$, the result of $F()$, and this will become the IID.

This approach can significantly decrease the possibility of scanning attacks and is also useful for nodes, working as servers, that need a stable address but do not want to expose their hardware information (not generating an IID based on MAC). 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 period of time that the subnet prefix is valid. In real life, the subnet prefixes are not frequently changed. It can be valid for several months to years, except for one country, i.e. Germany [9], where the subnet prefix is valid for only a week. So the main drawbacks to the use of this approach is that the node must always wait to see when the prefix changes in order to generate a new IID. Otherwise this IID will be kept forever. This is also true when nodes change, for any reason, their network adapter card (hardware problem). This mechanism recommends that the node still configure the same IID as it had for this prefix.

The scenario that leads to an increase in the possibility of this attack is that, in IPv6, the node is supposed to use its public address to connect to the Internet, or other networks, and not use Network Address Translation (NAT) [3] as is used in IPv4. This is because it can cause problems with end-to-end communications. Also, one of the features used to promote IPv6 was the large address space that was advertised as being able to support unique addresses for each individual device in use. This is not possible in IPv4, as it has already exhausted its supply of addresses. So this means that attackers can only rely on fake websites or attack DNS servers to gather a node's IP addresses that he can later use to perform attacks against this node, which will harm the privacy or security of this node.

## CGA and its drawbacks

CGA [4] is an important option within SeND. It gives a node the ability to generate the IID portion of its IP address, securely, by the use of one way hashing. It not only provides for the generation of a randomized IID, which is not based on the MAC address, but also provides the node with proof of address ownership. To generate an IID using CGA, a node will first need to generate a random number, which will be called the modifier. It will concatenate this number with a zero valued subnet prefix and a zero valued collision count and then, with the public key. SHA1 will then be executed against the value that results from this concatenation. The 16 by sec result, obtained from the execution of SHA1, is compared to zero. The sec is a value between 0 and 7. The higher the sec value the better will be the protection afforded a node against brute force attacks. When the condition is met, the node will concatenate the modifier with the real subnet prefix, the public key, and the collision count and will then execute SHA1 again. It then will take the 64 leftmost bits of the SHA1 digest and set bits u and g to one. If the condition is not met, then the node will increment the modifier and repeat the process. When a sec value higher than 0 is chosen, there is no guarantee that the 16 by sec value equal to zero condition will ever be met. So the problem with the use of the CGA algorithm is the fact that it is compute intensive. It is for this reason that when the node generates the IP address using CGA, it does not change it and continues to use it forever, thus making this node vulnerable to privacy related attacks. There is a recent work that explains how to use CGA with regard to privacy[1]. In this approach a new temporary CGA address should be generated, when a host joins a new subnet, or before the lifetime for the in-use CGA address has expired, or when the subnet prefix lifetime has expired, or when the user needs to override the default values of CGA. However, as this approach only tried to address the privacy issues with respect ot the use of CGA, it does not solve the heavy computation, or those looking for a faster way to protect their privacy.

| Privacy Level | Algorithm |
|---|---|
| Extreme | Combination of our algorithm with cryptographic approaches |
| High Moderate | Combination our algorithm with randomized data sending approaches |
| Moderate | our algorithm |
| Low Moderate | Stable Privacy-Enhanced IID Generation Algorithm |
| Low | Privacy Extension |
| No Privacy | IID generation algorithm based on EUI-64 or MAC address |

**Table 1: Privacy Protection Afforded by Different Algorithms**

## Other Privacy Algorithms

[6] surveyed some network layer algorithms that can be used to protect a user's privacy. For example, one might encrypt the payload and submit it through many different intermediate nodes. The drawback in this case 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 Virtual Private Network(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 which used that node to start their VPN connection. Generally, the approaches explained in [6] can cause delays or be a single point of failure. There are some other privacy models that can be applied to the application layer where an attempt is made to apply some type of masks to data in order to remove critical confidential data from it. The non critical data are then exposed to others [19]. One problem with the use of these approaches is that it is assumed that exposing this data to the public is a non issue and that they do not expect that the receiving node would want to have the original data and not the data modified by the algorithm. This means that they might provide a level of anonymity, but they fail to give the receiver a copy of the original data. So the result is only good for use with statistics or showing something online.

## 4. PROPOSED ALGORITHMS

Table 1 shows our categorization of IID generation mechanisms and the level of privacy we expect to offer nodes in the network who are using our algorithm.

### 4.1 Router Advertisement based privacy

As shown in table 1, many protocols currently in use for providing privacy actually do not provide the level of privacy assurance that is needed and expected. Our solution is based on the assumption that, by having a hard to guess IID, the attacker will be unable to track the node and thus will be unable to initiate any further attacks against its privacy or security. This is because node scanning is the initial step for further attacks against a target node or network. This section introduces our solution to the privacy problems that were explained earlier.

### IID generation Algorithm

In order to generate a more highly randomized IID, without the need for a stable storage area (as needed in Privacy Extension mechanism), a node needs to follow the following steps:

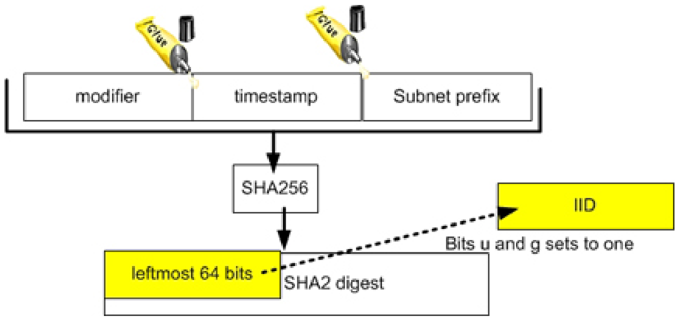

**Figure 1: RA-based Privacy Algorithm**

1. Generate a 16 byte random number called the modifier.
2. Obtain the router prefix from the Router Advertisement Message.
3. Obtain the node's current time and convert it to a timestamp. The timestamp is a 64-bit unsigned integer field containing a timestamp. This value indicates the number of milliseconds since January 1, 1970, 00:00 UTC, using a fixed point format.
4. Concatenate the modifier to the timestamp and the router prefix and call the resulting value R1. (see figure 1)
5. Execute SHA2 (256) against R1.
   digest=SHA256(R1)
   SHA2(256) is used because the chances of finding a collision are less than when using SHA1 and the generation time is in microseconds using a standard CPU.
6. Take the 64 leftmost bits from the digest generated in step 5 (the SHA2 digest) and call this the IID. The [5] has relaxed the use of the u and g bits (bits 7 and 8 from the leftmost byte of IID), which means that any new IID generation approach can decide for itself whether these bits can be meaningfully defined and used in their mechanisms (reserved for particular usage), or can be considered the same as the other bits in the IID. This algorithm has assigned no meaning to these bits.
7. Concatenate the IID with the local subnet prefix in order to set the local IP address.
8. Concatenate the IID with the router subnet prefix (Global subnet prefix) obtained from the RA message, and then set this as the tentative global privacy IP address. This IP address will become permanent after Duplicate Address Detection (DAD) processing has completed.
   After the DAD process, if the node finds collisions in the network then the modifier will be incremented and the DAD process will be repeated. If after 3 times, it receives the same result, it will consider this an attack and will start using that IP address.

Published as : Hosnieh Rafiee and Christoph Meinel. "Privacy and Security in IPv6 Networks: Challenges and Possible Solution". The 6th International Conference on Security of Information and Networks (SIN2013), ACM, November 26 - 28, 2013 Aksaray, Turkey

## Application layer based lifetime

There are two types of global IIDs: public and temporary. The public IID can also have a DNS record associated with it, which, in this case, means they need to be defined in the DNS. Temporary IIDs are those that do not have a DNS record and are only valid for a short period of time. For nodes where privacy is very important, using public addresses that have DNS records might prove problematic. This is because the attacker can find the location of this node based on its DNS records. This is true, especially in the case where the public IID never changes, or the IID is generated based on a MAC address.
There are two solutions available to nodes needing to use public addresses, which have associated DNS records. One is using an IID that was not generated based on a MAC address. The node can generate its IID using our algorithm as explained in the last section. This allows it to have a permanent lifetime. The public IID might also change whenever the node receives a new router advertisement message. Another solution would be to add the temporary IID to the DNS server. But in this case, the node would need to employ some type of security mechanism, when running the DNS Dynamic Update (DDNS) process, in order to allow for the updating of its own DNS records after it generates its new IID, like the mechanisms proposed in [16] and [15].
It is not easy to come up with a method by which privacy is observed, while at the same time not dissatisfying the user by cutting his connections because the lifetime for the IID has expired. One possible way of maintaining the lifetime of an IID is connection based (layer 4), although this way may prove problematic for FTP and other applications. Another possible way of maintaining the lifetime of an IID is application layer based. Having an application generate an IID for its connection and keeping it as long as this connection is valid, has some advantages and disadvantages. The advantages are that the IID is only valid for as long as it is used by that specific application, then it will be deprecated (expired). The disadvantage is that the number of valid IIDs will be out of control as there is no central process to control this. Another disadvantage is that the application should be "privacy aware" and should be able to generate a new IID.
The solution that we introduce here is to have a framework that assigns an application to a valid IID. In this case the application does not need to be "privacy aware" and also the number of valid IIDs can be controlled without extra effort since all IIDs are assigned by a centralized process in the node. This means that the application will open a connection using an IID and this connection will remain active for as long as the application uses it. If it is not used for about 5 minutes, and no other application is assigned to this IID, then this IID will be removed.

Figure 2 depicts a mechanism for assigning an IID to a new application where $app_i$ is a new application started by the node, $t\_app$ is the maximum number of applications per IID, $l$ is the maximum lifetime of an IID, $max\_IID$ is the maximum number of valid IIDs, $c\_IID$ is the current number of IIDs, $IID_i$ is a specific IID, $t\_app_i$ is the total number of applications per specific IID, $n_i$ is the current number of applications for a specific IID, $l$ is the maximum lifetime per IID (any newly generated IID sets its lifetime based on this value), and $l_i$ is the current lifetime of a specific IID. When a node wants to use a new application, it first checks to see whether or not it has also received a new router advertisement message. In the case where the node receives a new router advertisement message, it sets the total lifetime for the current valid IID to zero and resets the $c\_IID$ to zero. In this case all of the node's current IIDs will expired and the node should generate and use new IIDs for any upcoming applications. But the node can still use the expired IID as long as the current applications, which use them, are active. If the node does not receive any new router advertisement message, then it checks to see whether or not the current number of IIDs is less than the maximum number of allowed IIDs. If the condition is met, the node checks to see whether or not there are any IIDs where their current number of applications, or $n_i$, is less than the total number of applications, or $t\_app_i$. If the condition is met the node will sort these IIDs based on their current lifetime, or $l_i$, in descending order and then will assign $app_i$ to the IID with the larger $l_i$. Then the current number of applications, ($n_i$), for this specific IID will be incremented. If $n_i$ is equal to $t\_app_i$, then the node generates a new IID and assigns this application to this new IID.
In cases where the current number of IIDs is equal to $max\_IID$ and $n_i$ is equal to $t\_app$, then the node will be unable to generate a new IID for the application nor will it be able to assign a current IID to the application. In this case the node will find the IID with the longest lifetime and will increase the total number of applications, or $t\_app_i$, that can be assigned to it. Then the node will assign this IID to the new application. The advantage to using this approach, with regard to the IID lifetime, is that it allows for the control of the number of valid IIDs, while at the same time enabling users to keep their current application layer connections, which will lead to a user's satisfaction.
To consider the case where there is more than one router in a network, and so as not to generate thousands of IIDs each time the node receives a new RA message, it checks the list of routers in its neighboring caches. If the router is the same router, then it will not generate a new IID, as long as the current IIDs are valid. It will use the current IIDs for the new applications based on the application layer based algorithm. If a node receives two or more router advertisement messages, then it needs to check whether or not all these routers are in the same network. To do this it will send a Router Solicitation (RS) message and wait to receive the RA message. If, after a short interval of time, it receives the RA messages responding to its RS message, then it assumes that all routers are in its network. It will thus not generate another IID until the lifetime of the current IID has expired (if it already generated an IID for the current RA messages) and it will not reset the $c\_IID$ for each router. But it will divide the $max\_IID$ by the current number of routers in the network and then for each prefix, it will generate a certain number of IIDs if there are applications wanting to use them. This will also avoid the problem of having thousands of multicast groups in a network by dividing the $max\_IID$ by the number of routers. This means that the node will not have more than a certain number of valid IIDs for a certain period of time. This value is unrelated to the number of routers in the network but by increasing the numbers of routers in the same network, the number of IIDs per prefix decreases. This is because $max\_IID$ will never change for a particular network.

### *Automate the process for the lifetime.*

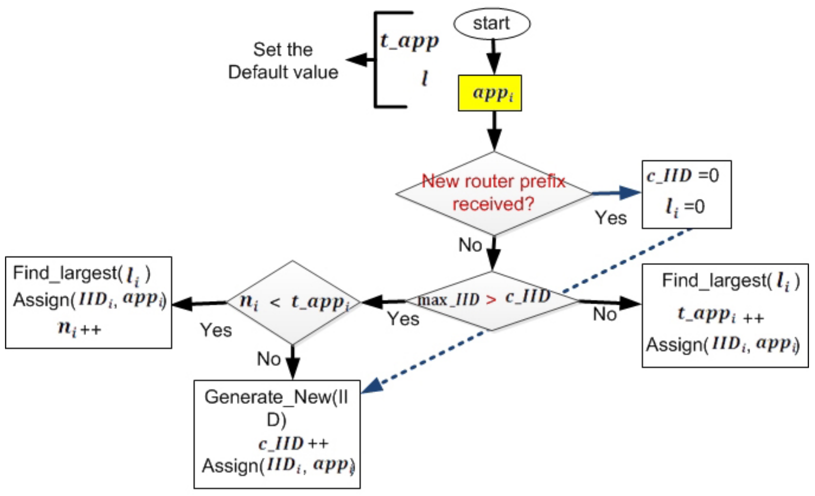

**Figure 2: Interface ID's lifetime algorithm**

| | | |
|---|---|---|
| Type = 26 (1 byte) | Length (1 byte) | Reserved (2 bytes) |
| Lifetime (4 bytes) | | |
| Max_IID (1 byte) | Max_Application (1 byte) | ... |

**Figure 3: Proposed data structure to Router Advertisement**

We propose to add a data structure to the optional section of RA messages where by, when this message is received and processed by nodes, all nodes will be enabled to, at the same time, set their default values, such as *t_app*, $max\_IID$ and $l$, as explained in figure 2. This will apply to all addresses that were generated using this method. This will eliminate the manual step that is needed to set these values, based on network policy, for any future IIDs generated using our approach. Figure 3 shows our proposed RA message data structure.

- *Type* is a 1 byte identifier.
- *Length* is the 1 byte total length for this structure starting with *Type* and including all options
- *Lifetime* is $l$ or the maximum lifetime of any newly generated IID,in seconds.
- *Max_ID* is $max\_IID$ or the maximum number of IIDs per Network Interface.
- *Max_Application* is *t_app* or the maximum number of applications per each valid IID.

## 5. IMPLEMENTATION AND EVALUATION

### 5.1 Implementation

The factor that we evaluated here is how much processing time the CPU will need to accomplish the steps in our algorithm. We implemented our approach using c#.net on a windows Operating System (OS) using a computer with 8 GB RAM and 2.70 GHz CPU. To simplify the implementation, we consider that all processes in the OS want to access the network adapter card and connect to the Internet. With this assumption, we had a thread that checked the list of processes in the system and obtained their dependent processes and assigned an IID to them. The total number of processes in the system at the start of the application was 125. Of that 125, 50 processes were dependent processes to the other processes in the system. The $max_IID$ was 10 and we assumed that there is one router in the network. $l$ was 5 seconds. We ran this application for 10 minutes and during this time we opened new applications to increase the number of processes in the system. The total number of deprecated IIDs (IIDs that were no longer valid) was 96. We noticed that our application, with all its threads, used an average of 10% of the CPU to run our algorithm.

### 5.2 Threat Analysis

#### Node Scanning

When a node keeps its IID for only a short period of time, and when it also changes its IID when the prefix changes, then it becomes very difficult for an attacker to find the IP address of the node. As the initial phase of any attack is scanning the network in order to find the IP address of the nodes, and then to run port scanning in order to find the available services running on those nodes in order to take advantage of the vulnerabilities of the services running on the node will result in a compromise of the node's security. So, by using our algorithm, not only do we provide the node with a level of privacy assurance but also we provide the node with security protection.

#### Location based tracking

This is the same as node scanning so it will be very difficult for an attacker to track the node across the network. The reason for this is because it is very difficult for an attacker to recognize that this node is the same node, but with a newly generated IID. This is especially true where there is an unlimited number of nodes in use on the same network.

### Obtaining confidential data

When a node frequently changes its IID within the network, and also among networks, then attackers probably won't have enough time to obtain the user's confidential data. It will also be difficult for an attacker to correlate the information that he does obtain to a specific user's IP address. This means that it will be difficult for the attacker to obtain more information about this user based on any correlation of data. An example would be when an attacker obtains a confidential document from a user, but he is unsure about the location of this user. If the attacker had the user's location he would be able to obtain much more information about this user, and then, he would be able to start other attacks against him. But changing the IID prevents an attacker from finding the location of this user and thus prevents further attacks.

## 6. FUTURE WORK

Our implementation does have some limitations because we had to make some assumptions, such as using all available processes of the system, etc. To test the performance of our algorithm in real life, we plan to improve the algorithm and install it in our lab where we can test it on several different nodes at the same time.

## 7. CONCLUSIONS

In this paper we introduced the mechanisms that are currently used for IID generation in IPv6 networks. We then explained the deficiencies that are present when these algorithms are used. To address those deficiencies we offered two algorithms, one for a highly randomized generation of the IID in order to make it difficult to guess what the next IID generated by the node will be, and the second algorithm which provides support for the application layer lifetime. We then implemented our work and evaluated it based on CPU usage. Our implementation results showed that our algorithm used, on average, 10% of the CPU resources.

## 8. REFERENCES

[1] A. AlSa'deh, H. Rafiee, and C. Meinel. Ipv6 stateless address autoconfiguration: Balancing between security, privacy and usability. In *Foundations and Practice of Security*, pages 149–161. Springer Berlin Heidelberg, 2013.

[2] J. Arkko, J. Kempf, B. Zill, and P. Nikander. Secure neighbor discovery (send). IETF, Mar. 2005. http://tools.ietf.org/html/rfc3971.

[3] F. Audet and C. Jennings. Network address translation (nat) behavioral requirements for unicast udp. IETF, Jan. 2007. http://tools.ietf.org/html/rfc4787.

[4] T. Aura. Cryptographically generated addresses (cga). IETF, Mar. 2005. http://tools.ietf.org/html/rfc3975.

[5] B. Carpenter and S. Jiang. Significance of ipv6 interface identifiers. Work in Progress, Aug. 2013. http://tools.ietf.org/html/draft-ietf-6man-ug-02.

[6] P. M. d. Silva, J. Dias, and M. Ricardo. Survey on privacy solutions at the network layer: Terminology, fundamentals and classification. http://paginas.fe.up.pt/ prodei/dsie11/images/pdfs/s6-4.pdf.

[7] S. Deering and R. Hinden. Internet protocol, version 6 (ipv6) specification. IETF, Dec. 1998. http://tools.ietf.org/html/rfc2460.

[8] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney. Dynamic host configuration protocol for ipv6 (dhcpv6). IETF, July 2003. http://tools.ietf.org/html/rfc3315.

[9] Expertengespraech zu ipv6, 2012. http://tinyurl.com/pne8hna.

[10] European Union Data Protection Directive: Processing of personal data and on the free movement of such data, 1995. http://tinyurl.com/6gpkrav.

[11] F. Gont. A method for generating stable privacy-enhanced addresses with ipv6 stateless address autoconfiguration (slaac). Work in Progress, June 2013. http://tools.ietf.org/html/draft-ietf-6man-stable-privacy-addresses.

[12] R. Hinden and S. Deering. Ip version 6 addressing architecture. IETF, Feb. 2006. http://tools.ietf.org/html/rfc4291.

[13] T. Narten, R. Draves, and S. Krishnan. Privacy extensions for stateless address autoconfiguration in ipv6. IETF, Sept. 2007. http://tools.ietf.org/html/rfc4941.

[14] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor discovery for ip version 6 (ipv6). IETF, Sept. 2007. http://tools.ietf.org/html/rfc4861.

[15] H. Rafiee, M. Loewis, and C. Meinel. Dns update extension to ipv6 secure addressing. IEEE, Proceedings of 27th International Conference on Advanced Information Networking and Applications workshops, 2013.

[16] H. Rafiee and C. Meinel. A secure, flexible framework for dns authentication in ipv6 autoconfiguration. IEEE, Proceedings of the 12th IEEE International Symposium on Network Computing and Applications (IEEE NCA13), 2013.

[17] European union data protection proposal. Online, 2012.

[18] S. Thomson, T. Narten, and T. Jinmei. Ipv6 stateless address autoconfiguration (slaac). IETF, Sept. 2007. http://tools.ietf.org/html/rfc4862.

[19] T. M. Truta and B. Vinay. Privacy protection: p-sensitive k-anonymity property. Proceedings of the 22nd International Conference on Data Engineering workshops, IEEE, Computer Society, 2006.