Defeating NIDS evasion in Mobile IPv6 networks

Michele Colajanni, Luca Dal Zotto, Mirco Marchetti, Michele Messori
Department of Information Engineering
University of Modena and Reggio Emilia
Modena, Italy
{*michele.colajanni, luca.dalzotto, mirco.marchetti, michele.messori*}*@unimore.it*

***Abstract*—**
The diffusion of mobile devices and technologies supporting transparent network mobility can have detrimental effects on network security. We describe how an attacker can leverage mobility in IPv6 networks to perpetrate known attacks while evading detection by state-of-the-art Network Intrusion Detection Systems (NIDSs). We then propose a new defense strategy based on the exchange of state information among distributed NIDSs. We demonstrate the effectiveness of the proposed solution through a prototype implementation, evaluated experimentally in a Mobile IPv6 network.

***Keywords*-network intrusion detection; NIDS state migration; mobility-based NIDS evasion; Mobile IPv6;**

I. Introduction

The number and the computational power of Internet-enabled mobile devices are increasing rapidly. Netbooks, smartphones and tablet PCs allow users to be always connected, thus encouraging the diffusion of new services as well as mobile versions of existing services, ranging from social networking to mobile banking. For these reasons, the adoption of network technologies that support transparent node mobility (i.e., the ability to roam between different networks without interrupting open connections) is increasing as well. One of the most promising solutions is the mobility extension of IPv6 (Mobile IPv6 [1]), which is expected to become the network layer of Internet and coming 4G networks (e.g. LTE-Advanced [2]).

While transparent node mobility offers new important opportunities, it also introduces new security risks. In particular, we have recently observed that an attacker can exploit transparent node mobility to perform "stealth" network attacks, that are not detectable even by stateful and state-of-the-art Network Intrusion Detection Systems (NIDS) [3].

The first contribution of this paper is the description of several strategies that an attacker can implement to evade NIDSs in networks that support the Mobile IPv6 protocol. These evasion techniques are not caused by flaws in any NIDS implementation [4]–[6]. They are the direct consequence of node mobility and of specific characteristics of Mobile IPv6, such as Route Optimization. Moreover, they are immediately applicable in all the networks that support this protocol.

The second contribution of this paper is the design of a new NIDS cooperation strategy based on the exchange of *state* information [7] among distributed, stateful NIDSs. The proposed solution prevents an attacker from exploiting node mobility to evade detection and does not require any modification to the Mobile IPv6 protocol nor to the hardware and software of mobile nodes.

We demonstrate the viability of the proposed solution by implementing a prototype based on open source software, whose effectiveness and performance are experimentally evaluated in a realistic network scenario.

The strategies that allow an attacker to evade NIDS detection in Mobile IPv6 networks are described in Section II. Our solution for defeating mobility-based NIDS evasion is presented in Section III. The implementation details of the prototype used for experimental evaluation are discussed in Section IV. The network testbed and the experimental results are presented in Section V. Section VI compares our work with previous papers in the fields of NIDS evasion and parallel and distributed NIDS architectures. Section VII concludes the paper and outlines future works.

II. NIDS evasion in Mobile IPv6 networks

Mobility-based evasion, described for the first time in [3], is a NIDS evasion technique that allows an attacker to avoid detection by exploiting network mobility. In this paper we consider three different mobility-based evasion scenarios in Mobile IPv6 networks: *mobile attacker and fixed victim*, *mobile victim and fixed attacker* and *mobile victim and mobile attacker*, described in Sections II-A, II-B, and II-C, respectively. In all these scenarios we assume that:

- both the Home Network and the Foreign Network are monitored by a state-of-the-art stateful NIDS;
- the attacker tries to exploit a remote vulnerability of the victim by sending a malicious payload to it;
- it is possible to divide the malicious payload in (at least) two portions;
- a NIDS is not able to detect the attack by analyzing only a portion of the malicious payload.

Experimental results described in Section V show that all these assumptions can be easily met in realistic network topologies and with real network-based attacks. We remark that all the evasion strategies described below are not caused

978-1-4577-0351-5/11/$26.00 ©2011 IEEE

by a programming flaw in a specific NIDS implementation; they are the direct result of technologies that allow transparent node mobility. Moreover, while the attack relies on the fragmentation of a malicious payload, each fragment of the payload can be sent to the victim in a complete network packet. Hence these attack strategies do not depend on IP fragmentation, a practice that is already discouraged in IPv6 networks [8].

A. Mobile attacker and fixed victim

In this scenario, shown in Figures 1 and 2, the attacker controls a Mobile Node, that is used to carry out a stealth attack to a fixed Correspondent Node (victim). In Figure 1 the two clouds on the left represent the Home and Foreign Networks, while the third cloud represents Internet. The Correspondent Node is presented as a fixed PC, while the Mobile Node is a laptop. The dashed arrow indicates the migration of the Mobile Node from the Home Network to the Foreign Network. The information sent by the Mobile Node are the two portions of the attack, which follow the paths indicated by the solid arrows.

The attack can be performed by following the three steps described below.

Step 1: first attack portion. At the beginning of this scenario, the Mobile Node (attacker) is connected to its Home Network. The attacker splits the malicious network payload in two parts, and sends a network packet containing the first part of the attack (1° Portion, in Figures 1 and 2) from the Mobile Node to the Correspondent Node. This network packet is received and analyzed by the Home NIDS, that does not raise any intrusion alert since the packet contains only a portion of the attack.

Step 2: attacker roaming. The Mobile Node roams from the Home Network to the Foreign Network. In compliance with Mobile IPv6 specifications [1] the Mobile Node generates a unique *Care-of Address* belonging to the Foreign Network address space, and transmits it to the Home Agent through a *Binding Update* message. The Home Agent replies with a *Binding Acknowledgment* message. The Mobile Node also challenges the Correspondent Node's ability to support IPv6 mobility by sending *Home Test Init* and *Care-of Test Init* messages. If the Correspondent Node supports Mobile IPv6, then it receives a *Binding Update* message from the Mobile Node, and all the following network packets between the Correspondent Node and the Mobile Node will not be handled by the Home Agent (this is known as *Route Optimization*). On the other hand, if the Correspondent Node does not support Mobile IPv6, it continues to send packets to the Home Address. They will be received by the Home Agent and tunneled to the Mobile Node in the Foreign Network.

Step 3: second attack portion. When the roaming process is complete, the attacker sends to the Correspondent Node a network packet containing the second portion of the malicious payload. If Route Optimization is active, this network packet (2° Portion in Figure 1) is routed directly to the Correspondent Node. Hence, it is received and analyzed by the Foreign NIDS, that does not trigger any intrusion alert since the packet contains only a portion of the attack. Hence, by exploiting node mobility the attacker is able to deliver the malicious payload while evading detection by the Home and Foreign NIDSs. The Foreign Network infrastructures can be exploited by a mobile attacker to damage third parties, without the network administrators being able to prevent, stop, or even detect the attack. This holds true even when a different NDIS is placed at the victim's network.

If Route Optimization is *not* enabled, the packet containing the second attack portion (2° Portion in Figure 2) is routed through the Home Network. Hence, this packet is received by both the Home and the Foreign NIDSs. However, only the Home NIDS had received the first portion of the attack, and is able to detect the intrusion. The Foreign NIDS, that monitors the network used by the attacker to execute the final step of the attack, is unable to raise any security alert.

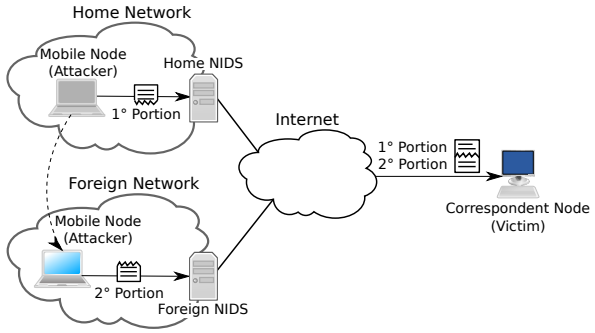

Figure 1. First evasion scenario: mobile attacker and fixed victim, with Route Optimization

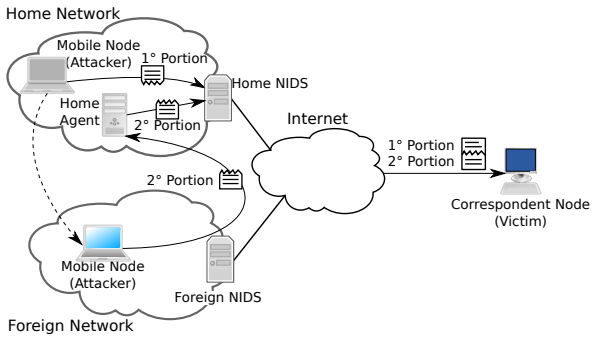

Figure 2. First evasion scenario: mobile attacker and fixed victim, without Route Optimization

B. Mobile victim and fixed attacker

In this scenario the attacker controls the Correspondent Node to which the Mobile Node (victim) is talking to. The

attack can be performed by executing the following three steps.

Step 1: first attack portion. At the beginning of the attack, the Mobile Node (victim) is connected to its Home Network. The attacker splits the malicious payloads in two portions and sends the first portion of the attack inside a network packet to the Mobile Node.

Step 2: detection of a roaming event of the victim. The Correspondent Node waits for the Mobile Node to roam to a different network. All the roaming events of a Mobile Node can be reliably detected by a Correspondent Node, since it receives *Home Test Init* and *Care-of Test Init* messages sent to it by the Mobile Node. Moreover, the attacker can choose whether to answer the messages, thus enabling Route Optimization, or not.

Step 3: last attack portion. In this step the attacker sends the last portion of the malicious payload to the Mobile Node.

If the attacker enables Route Optimization, the last portion of the attack flows directly from the Correspondent Node to the Mobile Node, without passing through the Home Network. Hence, the Home NIDS cannot detect the attack. On the other hand, the Foreign NIDS receives and analyzes the second portion of the attack, but is unable to raise a security alert since it has never received the first portion.

If the attacker does not enable Route Optimization, the second portion of the attack is routed to the Mobile Node through the Home Network. The Home NIDS can analyze the complete malicious payload and detect the attack, while the Foreign NIDS, that only receives the second portion of the attack, cannot.

In both cases, the Foreign Network, that is now hosting the compromised Mobile Node, is unable to detect the attack.

The attacker can also choose to implement a variant of this attack strategy by splitting the attack in at least three portions. The first portion is sent to the victim before the mobility event; the second portion is sent after the mobility event, but before Route Optimization; the last portion is sent after Route Optimization. This attack strategy is even harder to detect, since the three portions of the attack follow three distinct paths.

C. Mobile victim and mobile attacker

In the last scenario, both the attacker and the victim are supposed to be mobile nodes.

Step 1: first attack portion. At the beginning of the attack, the attacker and the victim are connected to the respective Home Networks. The attacker splits the malicious payload in two portions and sends the first portion to the victim.

Step 2: detection of a roaming event of the victim. The attacker waits for the victim to roam to a Foreign Network. For this roaming operation, the attacker acts as a Correspondent Node, and can decide whether to enable Route Optimization or not. We assume that the attacker enables Route Optimization.

Step 3: attacker roaming. The attacker roams to a Foreign Network. Since the victim is a Mobile Node, it supports Route Optimization. Hence, according to Mobile IPv6 specifications, it enables Route Optimization as a default setting.

Step 4: last attack portion. The attacker sends the last attack portion to the victim.

In this scenario, the first attack portion is received by the two NIDSs monitoring the victim's and the attacker's Home Networks, while the last attack portion is received by the two NIDSs monitoring the victim's and the attacker's Foreign Networks. Hence, none of these four NIDSs is able to detect the attack.

Several variants of this strategy are possible. The second and the third steps can be swapped without altering the outcome of the attack. Moreover, the attacker can split the malicious payload in more than two portions, delay the activation of Route Optimization, and send one or more intermediate attack portions to the victim through the attacker's Home Network and/or the victim's Home Network.

III. Solution through NIDS cooperation

To address the issue of mobility-based NIDS evasion we propose a cooperative solution, based on the exchange of state information [7] among distributed NIDSs. The main idea is to *export* state information related to a Mobile Node from the NIDS of its origin network, and to *import* this state information to the NIDS that monitors the destination network. This process, called *state migration*, allows the NIDS that monitors the destination network to receive all the state information that are needed to detect the intrusion, thus preventing an attacker to exploit mobility for evading detection.

The solution proposed in this paper focuses on Mobile IPv6, hence we need to coordinate state export, the transmission of state information and state import with the normal operations performed by mobile nodes in IPv6 networks. To this purpose, we introduce a new software module, called *External Agent*, that has to be deployed in all the networks that want to take advantage of NIDS cooperation. In our solution, the External Agent sniffs the control messages used by the Mobile IPv6 protocol and triggers the required state import and export operations.

We assume that the External Agent is able to analyze the traffic generated by all the mobile nodes within its network (since all this traffic is already monitored by a NIDS, this assumption can always be satisfied). We also assume that a trust relationship already exists between the networks among which the Mobile Node is roaming. In particular, we assume that the NIDS that performs the import operation trusts the state information previously exported by cooperating NIDS. Moreover, we assume that the transmission of state

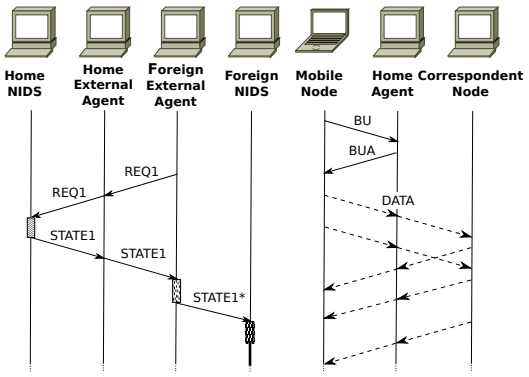

Figure 3. First Migration sequence diagram: Mobile Node roams from the Home Network to a Foreign Network

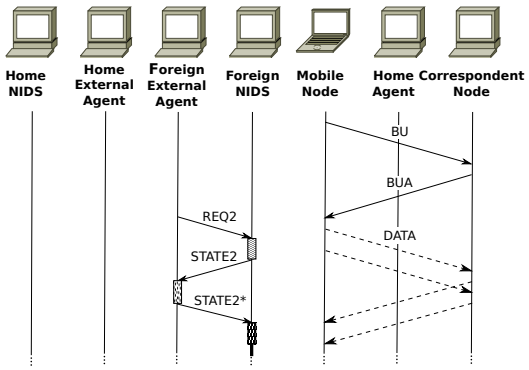

Figure 4. Route Optimization sequence diagram: Mobile Node and Correspondent Node enable Route Optimization

information is performed through a secure protocol (such as SSL/TLS) guaranteeing endpoint authentication, non-repudiation and message integrity. Finally, we assume that the External Agent deployed in a Foreign Network is able to determine the IP address of the External Agent deployed in the Home Network of a Mobile Node. This assumption can be easily satisfied by deploying the External Agent on the same machine that hosts the Home Agent (required by the Mobile IPv6 protocol specifications). It is worth observing that the proposed solution does not require any modification to the Mobile IPv6 protocol, hence it is compatible with all current IPv6 implementations.

The Mobile Node can perform three different actions:

1) **First Migration** the Mobile Node roams from the Home Network to a Foreign Network;
2) **Route Optimization** the Mobile Node and the Correspondent Node enable Route Optimization;
3) **Return to Home** the Mobile Node returns to its Home Network.

The following three sections describe how to react to each of these activities.

A. First Migration

In this phase, the Mobile Node leaves its Home Network and connects to the Foreign Network. In compliance with Mobile IPv6 protocol specifications, the Foreign Network advertises its network prefix to the Mobile Node (e.g. through an instance of the *radvd* [9] daemon). The Mobile Node then generates its *Care-of Address* and forwards it to its Home Agent by issuing a *Binding Update* message. All the messages required to perform state migration are shown in Figure 3.

The process of state migration starts after the Foreign External Agent (i.e., the External Agent deployed within the Foreign Network) sniffs the *Binding Update* (BU, in Figure 3) message that the Mobile Node sends to the Home Agent. From the BU message, the Foreign External Agent extracts the address of the Home External Agent (that is the same of the Home Agent), the *Care-of Address* and the Home Address of the Mobile Node. After having sniffed the *Binding Acknowledgment* (BUA) message that the Home Agent sends to the Mobile Node, the Foreign External Agent issues a request for all the state information related to the Home Address of the Mobile Node (the *REQ1* message in Figure 3). This request is received by the Home External Agent, and forwarded to the Home NIDS. The Home NIDS exports the state information related to the Mobile Node's Home Address (*STATE1* in Figure 3) and transmits it to the Home External Agent. The Home External Agent forwards this message to the Foreign External Agent that, after some pre-processing described in Section IV, sends the state information to be imported to the Foreign NIDS. After that, the Foreign NIDS contains all the state information related to the Mobile Node that is known by the Home NIDS, including the state related to attack portions previously analyzed by the Home NIDS. Hence, state import prevents mobility-based evasion.

B. Route Optimization

At the end of the *First Migration* phase, the Mobile Node is reachable through its Home Agent, that tunnels all the network packets to and from any Correspondent Node through an IPv6-in-IPv6 tunnel. If the Correspondent Node supports the Mobile IPv6 protocol, then Route Optimization will likely be enabled. However, Mobile and Correspondent node can start exchanging data before Route Optimization. The Foreign NIDS will then analyze packets flowing between the two endpoints, and will be unable to reassemble two attack portions transmitted before and after Route Optimization. We can prevent this issue by exporting all the state information extracted from the tunneled packets and merging it with the state information extracted from the packets exchanged between Mobile and Correspondent Nodes after Route Optimization. The messages required to perform state migration after Route Optimization are shown in Figure 4.

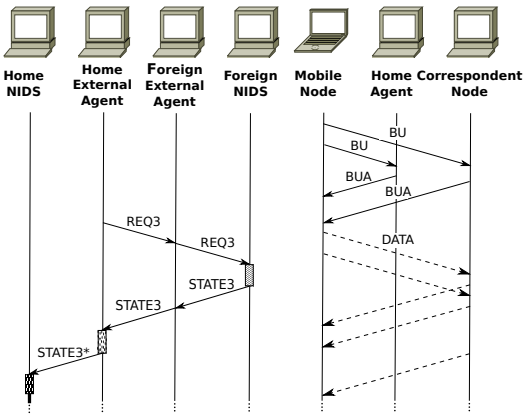

Figure 5. Return to Home sequence diagram: Mobile Node returns to its Home Network

The state migration process starts when the Foreign External Agent sniffs the BU and BUA messages exchanged between the Mobile Node and the Correspondent Node. From these messages the Foreign External Agent extracts the Home Address of the Mobile Node, the address of the Home Agent, and the address of the Correspondent Node. The Foreign External Agent issues an export request (REQ2 in Figure 4) to the Foreign NIDS, asking for all the state information related to the Home Agent. This state information (STATE2 in Figure 4) is received by the Foreign External Agent, where it is pre-processed in order to extract only the state information related to the network packets exchanged between the Mobile Node and the Correspondent Node that have just performed Route Optimization. The pre-processed state information (STATE2* in Figure 4) is then sent to the Foreign NIDS and imported.

After this state import operation, all the packets exchanged between the Correspondent and Mobile Nodes are seen by the Foreign NIDS as belonging to the same session. Hence the Foreign NIDS is able to reassemble and detect an attack even if portions of it were transmitted while the Mobile Node was in the Home Network, or after roaming but before Route Optimization.

C. Return to Home

When a Mobile Node returns to its Home Network all the state information related to the Mobile Node in its previous Foreign Network has to be exported from the Foreign NIDS and imported to the Home NIDS. The messages required to perform this operation are shown in Figure 5.

The state migration activities start when the Mobile Node reconnects to the Home Network and issues BU messages to the Home Agent and to the Correspondent Node. The Home External Agent sniffs the BU messages and extracts the *Care-of Address* used by the Mobile Node in the Foreign Network to which it was connected. After sniffing the BUA from the Home Agent, the Home External Agent issues a request to the Foreign External Agent (REQ3 in Figure 5), asking for all the state information related to the *Care-of Address* previously held by the Mobile Node. This request is forwarded to the Foreign NIDS, that exports the requested state information (STATE3 in Figure 5) and sends it to the Foreign External Agent, that forwards the reply to the Home External Agent. The Home External Agent pre-processes the state information and sends it to the Home NIDS, where it is imported and merged with the internal state.

IV. Prototype implementation

The cooperation scheme proposed in Section III is implemented through a prototype that includes two main components: a modified version of the open source NIDS Snort [10], augmented with state import and state export capabilities; the External Agent that interacts with Snort to coordinate export and import operations.

We exploit the modular architecture of Snort to implement the import and export functionalities. The state information that is relevant to our solution is held inside the pre-processor plugins, that implement protocol specific and stateful analyses.

In particular we focus on the `Stream5` pre-processor, a target-based TCP reassembly module capable of tracking sessions for both TCP and UDP. In our implementation, when we export all the state information related to a particular node, we extract from the `Stream5` pre-processor the ordered list of network packets that have the node as one of the communication endpoints. On the other hand, when we import state information, we make Snort analyze the list of packets previously extracted by a Snort instance.

In order to properly invoke import and export functions, we add to Snort a concurrent thread that executes a XML-RPC server [11] and waits for incoming RPC calls. The server makes available two remote procedures: `state.export` and `state.import`. A client can issue a `state.export` or a `state.import` RPC in order to respectively export or import the state information related to one or more hosts.

The External Agent is an autonomous software component, written from scratch in C language. It has four main tasks:

- it tracks the movement of the Mobile Nodes between different networks by sniffing Mobile IPv6 control messages;
- it interacts with other External Agents deployed in different networks;
- it pre-processes state information before invoking the state import RPC;
- it invokes the import and export RPC of the Snort instance belonging to its network.

The first task is accomplished by monitoring the Home Network through the *libpcap* library [12] and looking for *Binding Update* and *Binding Acknowledgement* messages.

Using this information the External Agent is able to track the movements of the Mobile Nodes. In particular, the External Agent deployed in the Home Network of a Mobile Node always knows to which Foreign Network the Mobile Node is currently connected.

To implement the remaining three tasks, each External Agent embeds a XML-RPC server that accepts the `state.export` RPC. When an External Agent needs to interact with a NIDS in a different network, it issues a `state.export` call to the External Agent that is deployed in the same network of the NIDS.

The called External Agent verifies the syntactic correctness of state export requests. Moreover, it can implement an access control list to filter requests coming from untrusted networks, limit the RPC rate, and perform other security checks. Then the called External Agent forwards the RPC call to the local NIDS, receives the result and sends it back to the remote External Agent.

Furthermore, depending on the nature of the state information that is exported, a pre-processing step may be required in order to modify the state information before importing it in the local NIDS. In the current implementation, the pre-processing activities of the External Agent depend on which activity, among the three described in Section III (*First Migration*, *Route Optimization* and *Return to Home*) is being performed. In any case, pre-processing is only required to ensure that Snort will be able to merge the network packets imported through the `state.import` RPC with the packets gathered from the monitored network.

In the phase called First Migration (see Section III-A) the External Agent receives the requested state information (STATE1 in Figure 3), extracts all the packets, adds an extra IPv6 header to simulate an IPv6-in-IPv6 tunnel between the Mobile Node and the Home Agent, and rebuilds an XML representation ready to be imported.

In the Route Optimization phase (Section III-B) the Foreign External Agent receives the requested state information (STATE2 in Figure 4). It extracts all the network packets and checks whether they belong to an IPv6-in-IPv6 tunnel. If this is the case, then the outer IPv6 header is removed, and all occurrences of the Home Address in the inner IPv6 header are substituted with the Mobile Node's *Care-of Address*. All the other network packets are dropped. The pre-processed packets are then used to rebuild an XML state representation that is imported by the Foreign NIDS.

In the Return to Home phase (Section III-C) the Home External Agent receives the requested state information (STATE3 in Figure 5) and extracts all the network packets. If the Mobile Node was not using Route Optimization in the Foreign Network, then the Home External Agent has to remove the outer IPv6 header added by the Home Agent to perform IPv6-in-IPv6 tunneling. If Route Optimization was enabled, the Home External Agent substitutes all the occurrences of the *Care-of Address* in the IPv6 header with the Home Address of the Mobile Node. The pre-processed packets are then used to rebuild an XML state representation that is imported by the Home NIDS.

V. Experimental results

We demonstrate the viability and the effectiveness of the solution proposed in Section III through experiments carried out in a real IPv6 network with support for node mobility.

The network architecture of our experimental testbed is represented in Figure 1. It contains two IPv6 networks with mobility support: the Home Network and the Foreign Network. Each network is monitored by a stateful NIDS, implemented through the modified version of Snort described in Section IV. For simplicity, the same machine executes an instance of the *radvd* daemon and of our External Agent described in Section IV (nothing prevents these two software to run on a different host). Hence, the machines named Home NIDS and Foreign NIDS in Figure 1 act now as NIDS, Home Agent and External Agent for the Home and the Foreign Network, respectively. These machines are GNU/Linux hosts (kernel version 2.6.34) equipped with a Pentium 4 CPU 1.8 GHz and 1 GB RAM, and are directly connected to a Cisco Aironet 1100 access point that gives connectivity to wireless mobile nodes. The Mobile Node is a Linux (kernel version 2.6.35) laptop equipped with a wireless network interface, while the Correspondent Node is a fixed Linux host (kernel version 2.6.35). Both the Mobile Node and the Correspondent Node run the *mip6d* daemon, hence they support Mobile IPv6 and Route Optimization.

We test the network environment by verifying that the Mobile Node can roam between Home and Foreign networks without interrupting open TCP connections. We also test the ability of the Home and Foreign NIDSs to detect known attacks by sending network packets that match a signature known by Snort. The signature used (snort signature id: 652) refers to a heap-based buffer overflow attack in the news reader of Microsoft Outlook Express 6 (CVE-2007-3897 [13]), and matches with the following pattern: `1094795585 |0D 0A|1094795585 |0D 0A|` (all the following experiments can be replicated with any other Snort signature). We verify that Snort is able to detect the attack even though the pattern is split between payloads of two different TCP packets, thanks to session reassembly implemented in the `Stream5` pre-processor.

In the following experiments we assume that the attacker controls the Correspondent Node, and tries to exploit a remote vulnerability in the Mobile Node (as explained in Section II-B).

In the first run we test the ability of our prototype to prevent the attacker to evade detection when the Mobile Node roams from the Home to the Foreign Network, as described in Section III-A. At the beginning of the experiment, the Mobile Node is associated to the Home Wireless Access Point and its Home Address (2001:db8::beef) belongs to the

Home Network's address space (2001:db8::/64). The Mobile Node connects to the Correspondent Node through the 119 network news server port, by using *netcat6* to open the TCP connection. Then the Correspondent Node starts the attack and sends the first half of the malicious payload, consisting of `1094795585 |0D 0A|`. The Home NIDS detects this packet. However, since this string does not match any signature, no alert is raised.

We then simulate a roaming event by using *iwconfig* to associate the wireless NIC of the Mobile Node to the Foreign Wireless Access Point. The Mobile Node receives network advertisements from the *radvd* daemon deployed in the Foreign Host, notifying that it is now connected to the Foreign Network (2001:db8:1::/64). The Mobile Node uses its MAC address to generate a *Care-of Address* (2001:db8:1:0:218:deff:fe25:599) that is unique and that belongs to the Foreign Network. It then issues *Binding Update* message to the Home Agent to notify it of its new network address.

The Foreign External Agent detects the *Binding Update* and the *Binding Acknowledgement* messages, extracts the Mobile Node's Home Address (2001:db8::beef) and issues a `state.export` RPC call for all the state information related to this IPv6 address. It receives the required state information exported from the Home NIDS and starts the pre-processing phase by extracting all the network packets. It then adds to each packet an outer IPv6 header whose IP addresses are the Mobile Node's Care-of Address and the Home Agent's IPv6 address, thus simulating the IPv6-in-IPv6 tunnel. The resulting network packets are then used to generate the pre-processed state information (in XML format) that is then transmitted to the Foreign NIDS as a parameter of the `state.import` RPC. At this point the Foreign NIDS contains all the state information related to the first portion of the attack, sent by the Correspondent Node when the Mobile Node was connected to its Home Network.

Then, the Correspondent Node sends the second portion of the attack over the same TCP connection previously opened by the Mobile Node. This packet is tunneled by the Home Agent to the Mobile Node, and received by the Foreign NIDS, that succeeds in reassembling the complete malicious payload end detecting the attack. We remark that, thanks to Snort's ability to reorder out-of-sequence network packets, the attack is successfully detected even if the relevant state information is imported only after the Foreign NIDS receives the second portion of the attack.

We repeated the same experiment without enabling the Home and Foreign External Agents, and verified that, as expected, the Foreign NIDS is not able to detect the attack. Hence, we demonstrated that an attacker can exploit Mobile IPv6 to perform stealth network attacks, and that our solution solves this problem.

We carried out several similar experiments that demonstrate the effectiveness of our solution in the scenarios described in Sections III-B and III-C. Due to space limitations, we cannot describe all the steps executed in these runs. However, we recorded all the network traffic generated and received by the Mobile and Correspondent Nodes and by the Home and Foreign NIDSs in `pcap` format. These traces are available for download at http://cris.unimore.it/DefeatingMIPv6Evasion.

We then perform several experiments to characterize the performance offered by our prototype implementation, expressed in terms of the time that is required to execute state migration activities.

We first measure the time required by our modified Snort implementation to export and import state information, without considering the network delays. To this purpose, we instrument the XML-RPC server to measure the time needed to execute `state.export` and `state.import` RPCs, and we repeat the previous experiment. To emulate different network conditions, we let the Home and Foreign NIDSs analyze synthetic traffic traces before performing state export and import operations. By changing the number of TCP connections contained in these traffic traces we can control the number of sessions maintained by the `Stream5` pre-processor, and measure how this variable influences state export and import times. Experimental results are shown in Figure 6.

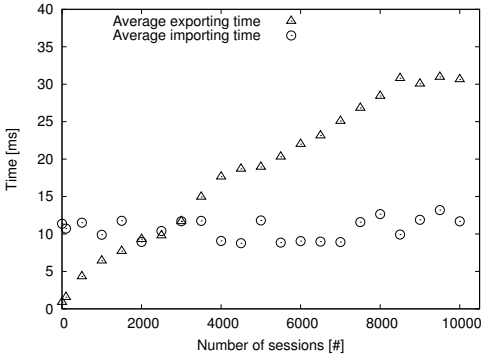

Figure 6. State importing and exporting times per session

The X-axis represents the number of concurrent connections maintained by Snort's `Stream5` pre-processor, while the Y-axis represents time, expressed in milliseconds. Triangles and circles represent the duration of state export and state import operations performed with different numbers of concurrent connections. Each point in the graph represents the average computed over five different measures.

By looking at the triangles, it is possible to see that the time required to export the state of the Mobile Node increases linearly with respect to the number of concurrent connections until about 8000 connections. After that, the state export time remains constant. This is consistent with the default configuration of Snort, that limits the number

of concurrent connections tracked by `Stream5` to 8192. However, we remark that in our experiments, even in the worst case, the time required to export the state information is 31 milliseconds. This linear relation does not hold for the time required to import the state of the Mobile Node in the Foreign NIDS, that is independent of the number of concurrent connections and always lower than 14 milliseconds.

We also measured the time required to perform a complete state migration, defined as the time elapsed from when the Foreign External Agent issues a `state.export` RPC to when it receives the result of the `state.import` operation from the Foreign NIDS. The average state migration time is 409 milliseconds. It is dominated by network delays and it is not correlated with the number of concurrent connections maintained by `Stream5`. This metric exhibits a standard deviation of 176 milliseconds, and in our experiments it never exceeded 765 milliseconds.

These experiments demonstrate that the time required to perform state migration is one order of magnitude lower than the time required to roam between the Home and Foreign Networks (on average 8.835 seconds in our experimental testbed), and compatible with real time analysis of live network traffic. These experimental results are summarized in Table I. We also remark that our prototype is able to handle out-of-order network packets, hence it tolerates network delays.

Table I
TIMES REQUIRED BY STATE MIGRATION ACTIVITIES

	Average [ms]	σ [ms]	Peak [ms]
State import	12	1	13
State export (worst case)	30	1	31
Complete state migration	409	176	765
Network roaming	8835	3495	13209

VI. RELATED WORK

NIDS evasion is a well known problem in the security literature. The seminal paper in this area [4] identified several evasion strategies that were effective against early stateless NIDS architetures. This work has been extended by other researchers [5], [6], [14]–[16] that explored new strategies that an attacker could pursue to deliver a malicious payload that was able to evade detection. There are many types of similar attacks, for example based on packet fragmentation, partial packet overlap, packets with wrong checksums, and even on differences between the TCP/IP implementation of a NIDS and that of the monitored hosts. All these evasion strategies rely on flaws in the algorithms used by a NIDS to reassemble network packets and their payloads, hence they can be (and indeed are) thwarted by a modern, well-configured and stateful NIDS.

The NIDS evasion strategy studied in this paper is a clear step ahead with respect to previous works: it exploits node mobility to prevent stateful NIDSs to gather enough data to detect an attack. Thanks to the mobility-based evasion technique, a NIDS is unable to analyze the complete malicious payload. Hence it is impossible for a NIDS to detect the attack independently of the algorithms used to reassemble packets, track connections and reconstruct data flows. Moreover, evasion strategies based on node mobility do not require packet fragmentation (e.g., through *fragroute*) nor sophisticated packet mangling (e.g., through *scapy*) activities, that are discouraged [8] and easily detected by modern NIDSs.

Mobility-based NIDS evasion was outlined for the first time by the same authors in [3] as a general strategy. This is the first paper that offers a detailed description and explanation of the steps required to exploit mobility-based evasion in the Mobile IPv6 protocol, including specific protocol features such as Route Optimization. In this paper we also describe, implement and evaluate the first solution to the mobility-based evasion problem. We claim that the only viable solution is based on some form of cooperation among the distributed NIDSs controlling different segments of the network.

Nowadays, there are several distributed NIDS prototypes and products. Software such as Prelude [17] can be used to realize hierarchical NIDS architectures that aim to collect, and possibly correlate, security alerts generated independently by different NIDSs. Research efforts such as Emerald [18], DOMINO [19] and Indra [20] extend this approach to multiple NIDSs that are geographically distributed across different domains. Other NIDS solutions focus on specific tasks, such as worm detection [21], load balancing and fault tolerance [22]. However, none of the proposed distributed NIDS architectures can solve the problem of mobility-based NIDS evasion.

Our solution is based on the exchange of state information among cooperative NIDSs. Similar cooperation schemes were already proposed in the context of parallel NIDS architectures [7], [23], [24] to allow the exchange of state information among clusters of NIDSs, each analyzing a small fraction of traffic gathered from the same network link. On the other hand, our proposal describes a new NIDS cooperation scheme where the state information related to a Mobile Node "follows" the Mobile Node in the new network, thus preventing an attacker from exploiting mobility to evade detection.

VII. CONCLUSION

In this paper we describe a new NIDS evasion strategy that allows an attacker to evade detection in IPv6 networks that support node mobility. Unlike previous evasion techniques, the technique that is analyzed in this paper does not exploit flaws in a NIDS implementation, but it is a direct consequence of transparent node mobility.

As a second contribution, we propose the first NIDS cooperation scheme that can be used to thwart mobility-based evasion in Mobile IPv6 networks. The viability of the proposed solution is demonstrated through a prototype implementation, whose performance are compatible with real-time analysis of network traffic. This cooperation scheme is immediately applicable to real networks because it does not require any modification of the Mobile IPv6 protocol and of mobile node software.

We plan to extend the solution proposed in this paper by implementing state export and state import features on different Intrusion Detection Systems, in order to enable cooperation between heterogeneous NIDS implementations.

ACKNOWLEDGMENTS

This research has been funded by the IST-225407 EU FP7 project CoMiFin (Communication Middleware for Monitoring Financial Critical Infrastructures).

REFERENCES

[1] D. Johnson, C. E. Perkins, and J. Arkko, "Mobility support in ipv6," RFC 3775 (Proposed Standard), Internet Engineering Task Force, June 2004.

[2] A. Scrase, "Overview of 3gpp release 10," ETSI MCC department, September 2010. [Online]. Available: http://www.3gpp.org/Release-10

[3] M. Colajanni, L. Dal Zotto, M. Marchetti, and M. Messori, "The problem of nids evasion in mobile networks," in *Proc. of the 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS 2011)*, Paris, France, February 2011.

[4] T. H. Ptacek and T. N. Newsham, "Insertion, evasion, and denial of service: Eluding network intrusion detection," Secure Networks, Inc., Suite 330, 1201 5th Street S.W, Calgary, Alberta, Canada, T2R-0Y6, Tech. Rep., 1998.

[5] V. Paxson, "Defending against network ids evasion," in *Recent Advances in Intrusion Detection*, 1999.

[6] S. Siddharth, "Evading nids, revisited," 2005. [Online]. Available: http://www.symantec.com/connect/articles/evading-nids-revisited

[7] M. Colajanni, D. Gozzi, and M. Marchetti, "Enhancing interoperability and stateful analysis of cooperative network intrusion detection systems," in *Proc. of the ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ACM/IEEE ANCS 2007)*, Orlando, FL, USA, Dec. 2007.

[8] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460 (Draft Standard), Internet Engineering Task Force, Dec. 1998, updated by RFCs 5095, 5722, 5871. [Online]. Available: http://www.ietf.org/rfc/rfc2460.txt

[9] P. Savola, "radvd." [Online]. Available: http://www.litech.org/radvd/

[10] "Snort home page." [Online]. Available: http://www.snort.org

[11] D. Winer, "Xmlrpc home page." [Online]. Available: http://www.xmlrpc.com/

[12] V. Jacobson, C. Leres, and S. McCann, "Pcap man page." [Online]. Available: http://www.tcpdump.org/pcap3_man.html

[13] "Cve-2007-3897." [Online]. Available: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-3897

[14] Y. Yoon, J. Y. Oh, and Y. M. Yoon, "Nids evasion method named SeolMa," *Phrack Magazine*, vol. 0x0b, no. 0x39, June 2001.

[15] S. Martin, "Anti-ids tools and tactics," SANS Institute Reading Room, 2001. [Online]. Available: www.sans.org/reading_room/papers/?id=339

[16] K. Timm, "Ids evasion techniques and tactics," SecurityFocus, 2002. [Online]. Available: http://www.securityfocus.com/print/infocus/1577

[17] "Prelude hybrid intrusion detection system." [Online]. Available: http://www.prelude-ids.org

[18] P. A. Porras and P. G. Neumann, "Emerald: Event monitoring enabling responses to anomalous live disturbances," in *In Proceedings of the 20th National Information Systems Security Conference*, 1997, pp. 353–365.

[19] V. Yegneswaran, P. Barford, and S. Jha, "Global intrusion detection in the domino overlay system," in *Proc. of the ISOC Symposium on Network and Distributed Systems Security*, Feb. 2004.

[20] R. Janakiraman, M. Waldvogel, and Q. Zhang, "Indra: A peer-to-peer approach to network intrusion detection and prevention," in *Proc. of the 12th IEEE International Workshops on Enabling Technologies*, Linz, Austria, Jun. 2003.

[21] M. E. Locasto, J. J. Parekh, A. D. Keromytis, and S. J. Stolfo, "Towards collaborative security and p2p intrusion detection," in *Proc. of the IEEE Information Assurance Workshop*, Maryland, USA, Jun. 2005.

[22] M. Marchetti, M. Messori, and M. Colajanni, "Peer-to-peer architecture for collaborative intrusion and malware detection at a large scale," in *Proc. of the 12th Information Security Conference (ISC 2009)*, Pisa, Italy, September 2009.

[23] M. Colajanni and M. Marchetti, "A parallel architecture for stateful intrusion detection in high traffic networks," in *Proc. of the IEEE/IST Workshop on "Monitoring, attack detection and mitigation" (MonAM 2006)*, Tuebingen, Germany, September 2006.

[24] R. Sommer and V. Paxson, "Exploiting independent state for network intrusion detection," in *Proc. of the 21st Annual Computer Security Applications Conference*, Tucson, AZ, USA, December 2005.