An Intrusion Debugging Technique Implemented in Mobile Based Network

N.K.Barpanda
Reader ,Dept.of Electronics

Premansu Sekhara Rath
Ph.D Scholar in SSSUTMS, Sehore Sambalpur University,Odisha, India.

Abstract: Now a days most of devices are mobility in nature. But the protocols supporting those devices should be taken into serious consideration in the field of security. In our work, we have introduced a novel class of network attack, where an attacker can't be identified by existing defensive mechanisms related to NIDS. We have proposed a new NIDS which can detect and prevent the mobility based new attack. The proposed system's performance is measured through number of protocols like mobile IPV4 and mobile IPV6 and WiFi protocols applied to the network.

Keywords: Mobility, NIDS, Mobile IPV4, Mobile IPV6, WiFi, Dodging

1. Introduction

To make the work simple, easy and effective, people uses mobile devices like cell phones , credit card, ATM card, laptops etc. RFID does the device management. The infrastructure cost is reduced highly by using the policy " Bring your own device". All users easily gain controlled access to the internal network resources through the mobile devices. CISCO predicted that the number of mobile devices will exceeded the number of world population by 2018.

The huge success and popularity of user mobility attracts to the attacker. At every instant, the devices is facing new attack due to exposure to the internet. Hence, we introduce a novel form of attacks called mobile dodging, that can be applied to mobile protocols like Mobile IPV4, Mobile IPV6 ad WiFi. This protocol support roaming events , that is it do not interrupt established connections. This is a mandatory feature for all applications requiring a stable connection, but it exposes mobile nodes and related networks called' stealth' network attacks. Existing defensive mechanism are not designed for facing transparent mobility, hence they are inherently incapable of detecting a mobile stealth attack. We initially present a model of mobile dodging attack that can be applied to any well known protocol. Then we propose an innovative solution that acts a novel way for NIDS cooperation.

2. Related Work

Mobile ad-hoc networks represent an important source of information about intrusion detection systems for wireless environments. For example, Thamilarasu et al. propose a Cross-layer Intrusion Detection System (IDS) in order to mitigate DoS attacks in ad-hoc networks with a focus on collisions, misdirection and packet drops. The cross-layer design is able to detect intrusion at different protocol layers and to exploit the information from one layer to another layer.Zhang at el proposed a distributed and cooperative IDS structure where each node participate with its information resources. We differ also from proposals using distributed intrusion detection systems (e.g., [6]) that gather data from differ-ent sources and send alerts to one aggregator analyzing and correlating all available information. These solutions are based on different IDS architectures: hierarchical (e.g., Emerald [14]), hierarchical and autonomous for cloud systems [17], peer-to-peer (e.g., [21,19,20] where the main goal is to avoid single points of failure).

3. Mobility Based NIDS dodging

We describe the mobile dodging attack by considering the most advanced stateful NIDS architectures, because stateless systems can be easily bypassed by several types of attacks and are now deprecated. In a stateful NIDS, the information of a network packet, which is relevant to intrusion, is used to create and update an internal state about all the active transport level connections. For each connection, a pre-processor maintains several metadata and two ordered lists of payloads (one for each direction) exchanged by the endpoints. The detection algorithm is then executed on the entire state information. As a consequence, although no individual packet contains enough information to detect an intrusion, a stateful NIDS can detect it by correlating information extracted from different packets. The problems originate when we consider a scenario allowing node mobility where an attacker can pursue mobility-based NIDS evasion that was introduced in [10]. This attack can be carried out in three scenarios that we describe in the following sub-sections. We underline that the following scenarios are independent of the mobile node roaming technology. Mobility-based NIDS evasion can be carried out if these realistic conditions are met:

1. the attack is a malicious payload exploiting a remote vulnerability;

2. it is possible to divide the malicious payload in at least two portions Portion 1, Portion 2 such that nor Portion 1 neither Portion 2 are detected by NIDS signatures;

3. the roaming process is transparent; active transport level connections are not interrupted by the handover process.

3.1 Mobile attacker, fixed victim

The attacker uses a mobile node which initially operates in a Home Network that allows node mobility. To pursue the threat, the attacker will migrate to a Foreign Network at some favorable time. Both the home and the foreign networks are monitored by stateful, signature-based NIDSs. Initially, the attacker chooses a remote vulnerability of the victim node and splits the corresponding payload in Portion 1 and Portion 2. Since IP packet fragmentation is discouraged in IPv6 and easily detected by modern NIDSs as anomalous network activity, the attacker packs the two portions inside two not fragmented TCP segments with consecutive sequence numbers. Then, the attacker establishes a TCP connection with the victim and sends Portion 1 from his home network. The home NIDS intercepts and analyzes Portion 1, updates its state information, but it does not have enough information to detect an intrusion. Meanwhile, the attacker roams to the foreign network.

After the migration, packets between the attacker and the victim can be routed through two different schemes: Direct Communication or Tunneling. Fig. 1a refers to direct communication, where the Portion 2 of the payload is routed directly through the foreign network. Portion 2 is intercepted by the foreign NIDS that has not received the previous packet and does not have the necessary state information to recognize the attack. This state information is possessed by the home NIDS, but it is useless since it does not receive Portion 2. As a result, none of the two NIDSs deployed in the home and in the foreign net-work can detect the attack. We note that this failure is not the result of a bug in NIDS implementations. Node mobility allows the malicious mobile attacker to perform a stealth attack, that is not due to some bug(s) in NIDS implementations. Depending on how node mobility is implemented, only a stateful NIDS installed in the victim node's network may be able to detect the intrusion attempt. In any case, both the home and the foreign network infrastructures can be exploited by a mobile attacker to damage third parties, without any evidence for the security administrator.

Fig. 1b refers to the tunneling scheme. The packets sent by the attacker are routed through the home network where a Home Agent processes them. This scheme guarantees the mobile attacker to migrate from one network to another while the fixed victim continues to reply to an address within the home network. Depending on the mobility implementation, the attacker might convey its packets through a Foreign Agent that is connected with the home agent. In both cases, Portions 1 and 2 of the payload are intercepted by the home network NIDS that can successfully detect the intrusion attempt.

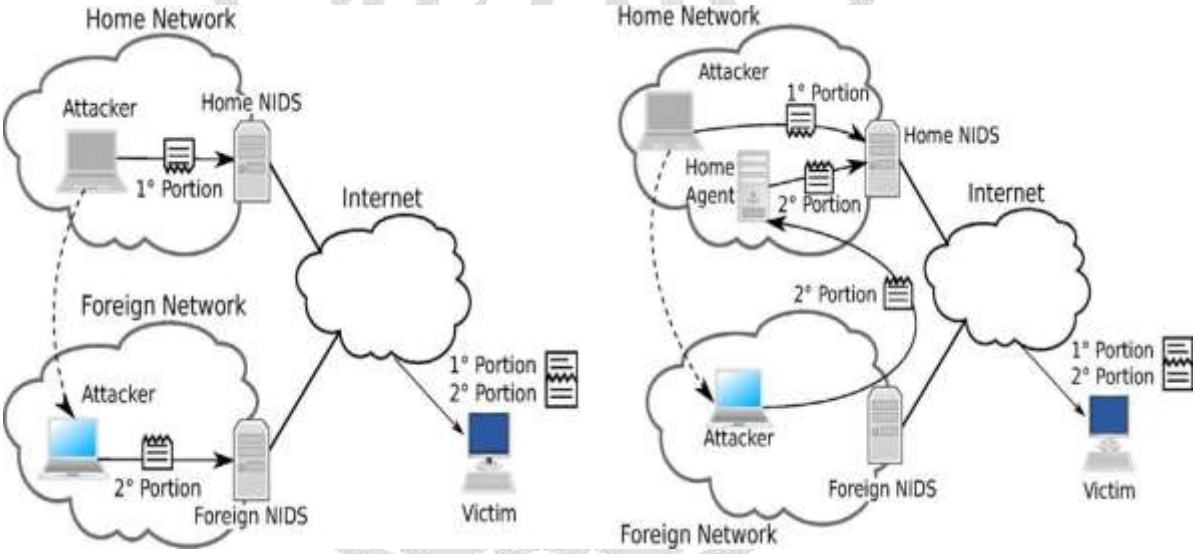
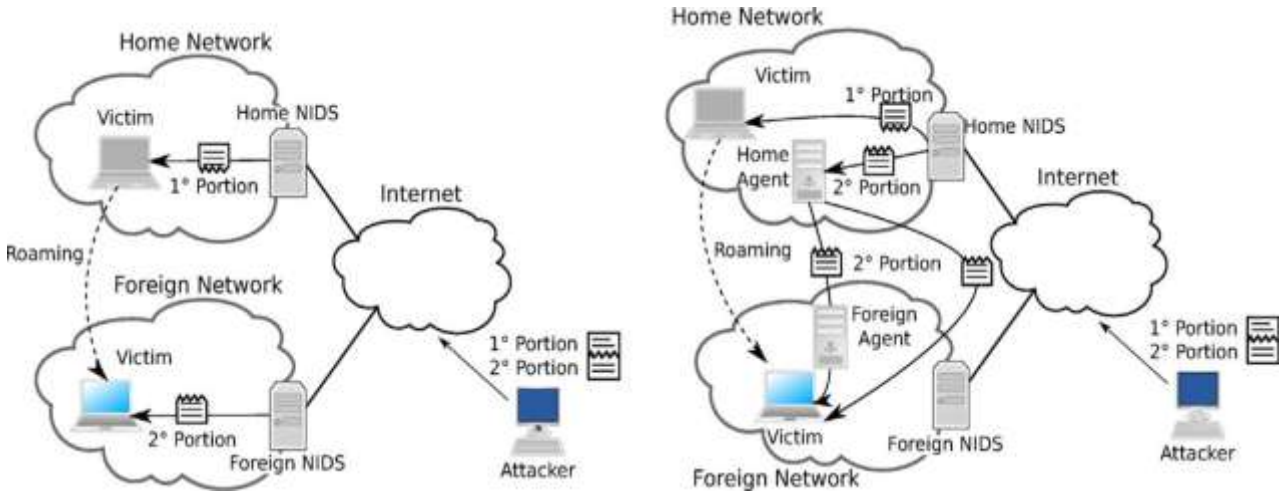


Fig.1. a.Direct communication b. Tunneling

3.2 Fixed attacker , mobile victim:

In this scenario, the roles are reversed with respect to those in Section 3.1. Here, the mobile node is the victim while the attacker node is fixed. We assume that the home and foreign networks are monitored by two stateful NIDSs, and that the attacker knows when the mobile victim roams across different networks. We show the feasibility for different mobile protocols in Sections. Fig. 2a illustrates the attack in the direct communication scheme. The attacker establishes a TCP connection with the victim and sends Portion 1 through its network to the home network. Portion 1 is intercepted and analyzed by the home NIDS, but it does not trigger an alert since it is not recognized as malicious. Then, the attacker waits for the mobile victim to roam to the foreign network. After the migration of the mobile victim, the attacker sends Portion 2 to the foreign network. Here, it is intercepted and analyzed by the foreign network NIDS, but it does not trigger an alert since it is not recognized as malicious. Ultimately, neither the home NIDS nor the foreign network NIDS receive enough information to be able to detect the stealth attack.



a. Direct communication fig-2 b. Tunneling

Fig. 2b shows the attack in the tunneling scheme. Portion 1 is sent directly to the mobile victim, hence it is received and analyzed by the home NIDS. Portion 2 is sent by the attacker after the mobile victim roams to the foreign network, and is routed through a home agent. Depending on the mobility implementation, the home agent can forward packets directly to the mobile victim or through a foreign agent in charge of the foreign network. In this setup we have a partially stealth attack. The home NIDS is able to inspect both portions of the payload, and to detect the attack. However, it cannot sanitize the mobile victim nor protect the nodes within the foreign network from the compromised machine. On the other hand, the foreign NIDS can only analyze Portion 2, thus being unable to detect the intrusion attempt. As a result, the mobile victim has been compromised while connected to the foreign network, the network administrator having no chance to detect the attack.

3.3 Mobile attacker , Mobile victim

The last scenario is a combination of the previous two cases. Here, both the attacker and the victim are mobile nodes. We assume that all the four networks involved in this scenario (victim home network, victim foreign network, attacker home network, attacker foreign network) are monitored by stateful NIDSs, and that the attacker knows when the victim roams to the foreign network. The attacker establishes a TCP connection with the victim and sends Portion 1 through its home net-work to the victim home network. Then, the attacker waits for the victim to roam to the foreign network. Then, the attacker roams to a (possibly different) foreign network and sends Portion 2. It is worth noting that the migration steps can be inverted without changing the attack outcome. Since the attacker roams to a foreign network before sending Portion 2, none of the two NIDSs deployed in the attacker's home and foreign networks receive the complete payload. Hence, they are not able to detect the attack. Moreover, since the victim is also roaming, the detection ability of the stateful NIDSs that monitor the victim's home and foreign networks is reduced. We distinguish between node mobility through tunneling and direct communication.

3.3.1Tunneling

Portion 2 is forwarded to the victim through a home agent, as shown in Fig. 2b. In this case the home NIDS of the victim receives Portion 1 and 2, and it is able to detect the attack. However, the attacker is able to inject the payload into the victim without evidence for the victim home and network NIDSs (partially stealth attack).

3.3.2 Direct communication

Portion 1 and 2 are routed directly to the victim, as shown in Fig. 2b. Here both the victim home and foreign NIDS are unable to detect the attack, because they do not receive the complete payload. Hence, none of the four stateful NIDSs can detect the attack, which is completely stealth.

4. Solution through Mobility based NIDS dodging

Mobility-based NIDS dodging prevents even modern stateful NIDSs to build a complete state, thus exposing them to the same evasion strategies that were effective only against obsolete stateless NIDS systems. We propose a cooperative solution based on distributed stateful NIDSs exchanging state information. The idea is to extend with three functions the mobility protocols that allow a mobile node to roam: extraction and serialization of state information related to a mobile node from the NIDS deployed in the origin network (state export), transmission of the serialized state to the destination network NIDS, deserialization

and merging of the transmitted state information within the state of the destination network NIDS (state import). The entire process is called state migration and allows NIDS state information to ''follow'' the mobile node in the new network, thus preventing de facto mobility-based dodging.

To implement state migration we introduce the External Agent, a modular software deployed in all the networks hosting cooperating NIDSs. For each supported mobile protocol, an external agent detects roaming events and triggers the appropriate state export and import operations among the NIDSs involved in the origin and destination networks. This solution requires only limited changes to the NIDS code base and can be

extended to different mobile protocols through new Plugging. We discuss three cases, corresponding to possible migrations of the mobile node. In the First Migration case, the mobile node roams from the home network to a foreign network. In the Return to Home case, the mobile node returns to its home network. In the Further Migration case, the mobile node roams from a foreign network to another foreign network. In the descriptions of these scenarios, we assume that:

1. state migration takes place via a secure channel (e.g., SSL/TLS) to guarantee authentication of the external agents, non-repudiation, message integrity and confidentiality;

2. a trust relationship between the networks among which the mobile node is roaming (otherwise, importing un-trusted data into a NIDS could compromise its integrity and thereby network security);

3. the external agent is capable of analyzing the whole network traffic (or at least the portion generated by the mobile nodes) within its network, with the goal of detecting when a mobile node joins the network;

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.

5. Performance Evaluation:

To show the viability and the effectiveness of the proposed solution to mobile evasion, we design and implement a prototype which provides state migration support among multiple NIDS sensors. This framework consists of three main components. The implementation of the state import and state export functions that extract and insert state information related to a specific IP address, respectively. This patch is integrated in the source code of Snort that is a popular open source NIDS.

The external agent that invokes the state.import and state.export functions. It is implemented as a software module that interacts with the local Snort and other external agents working on cooperating Snort modules set of plugins that manage the specific details of each mobile protocol. In this version, we have three plugins for WiFi, Mobile IPv4 and Mobile IPv6 protocols. There are no conceptual limits to integrate future protocols or different versions of existing mobile protocols.

These machines are connected to a Cisco Aironet 1100 access point providing wireless connectivity to mobile nodes. The mobile node is a laptop with a wireless network interface, while the correspondent node is a fixed host. All machines run GNU/Linux with kernels 2.6.35 or 2.6.39, recompiled with all the needed modules to support mobility. Node mobility through Mobile IPv4 is guaranteed by the home host and foreign hosts running the Dynamics Mobile IP daemon. In order to support Mobile IPv6 and route optimization, the home host, the mobile node and the correspondent node run the mip6d daemon, and the home host and foreign host run the radvd daemon. We replay all the described attack scenarios and verify the possibility of evading the NIDSs as reported in Section 3.

To evaluate the prototype performance we select one environment, Mobile IPv6, and analyze two main features: timings of the state migration procedure and impact of migrations on NIDS performance. We measure the time required by state migration and the execution time of state.export and state.import. We replay the mobility-based evasion several times under various network conditions. In particular, we generate synthetic network traces characterized by a different number of simultaneously active TCP connections. This is a factor with a direct impact on the resource consumption of the two NIDSs because each NIDS has to create and maintain a dedicated session for each monitored TCP connection. The measures related to these tests are reported in Fig. 8a, where the X-axis is the number of concurrent TCP connections in the network traces monitored by Snort, and the Y-axis reports the time in ms. The triangles and circles denote the duration of state export and state import operations under different numbers of concurrent connections, respectively. Each point in the graph repre-sents the average computed over five different experiments. Fig. 8a shows that the time required to import the state of the mobile node in the Foreign NIDS is independent of the number of concurrent connections and is lower than 14 ms. On the other hand, the time required to export the state of the mobile node increases linearly with the number of active TCP con-nections until Snort reaches the limit of concurrent connections tracked by the Stream5 pre-processor, which is 8192 by default. After that, the state export time remains constant. In our tests the time required to export the state information is less than 32 ms, even in the worst case.

We also measure the time required to complete the state migration process, defined as the time elapsed between the state.export request sent to the home external agent by the foreign external agent, and the receipt of the result of the state.import operation from the foreign NIDS. The related experimental results are summarized in Table 1. The average state migration time (409 ms) is dominated by network delays. This value is one order of magnitude lower than the time required by the mobile node to complete the roaming from the home to the foreign network (on average 8.835 s in our experimental tested). These results and the ability of our prototype to handle out-of-order network packets[1] show that it is compatible with real time analysis of live network traffic.

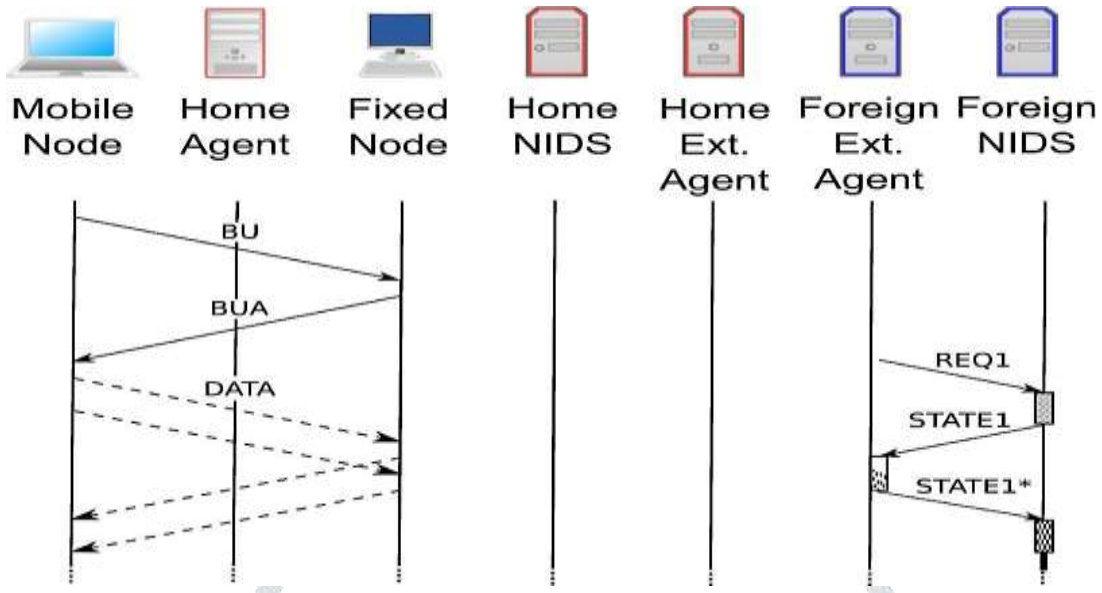
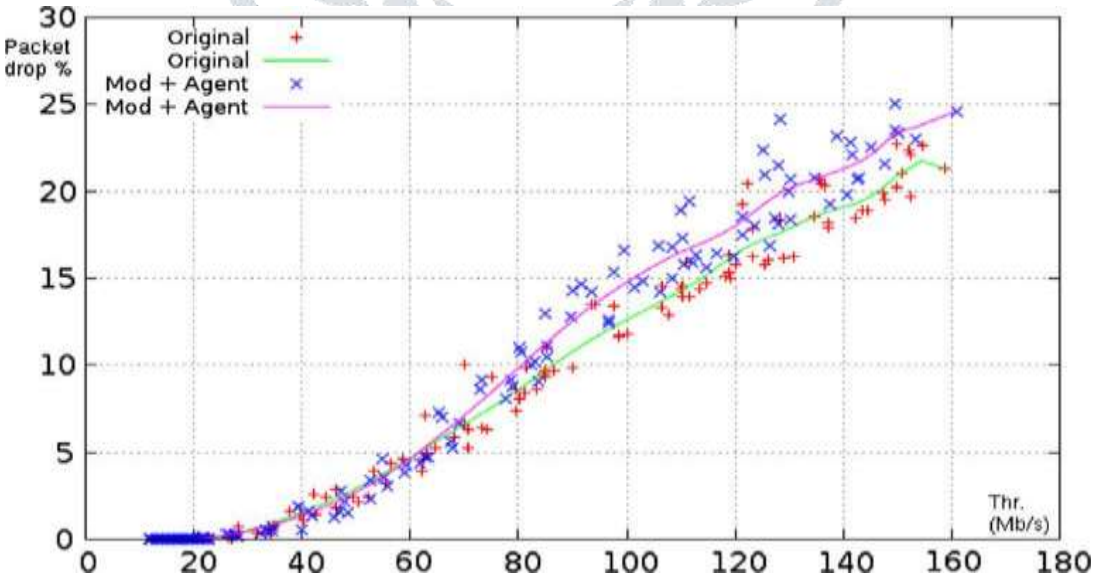


Fig-3 Route optimization sequence diagram

To estimate the impact of state migration on Snort performance, we run several experiments for evaluating the maximum NIDS bandwidth for a given packet loss rate. The choice of the traffic trace is not simple for a fair NIDS evaluation because there is no standard consensus, there is no benchmark and the widely adopted IDEVAL set [18] is now deprecated because it refers to a scenario that is not at all representative of modern traffic. In our experiments we use a trace of the Capture-the-Flag Hacking Contest held in 2010, which is reasonably recent, publicly available and contains several attacks.

In the first experiment we compare the performance of the original Snort against our framework, which includes a modified version of Snort and the external agent that we run on the same machines. We use the default Snort configuration, and we replay the registered network traces at different speeds ranging from few Mb/s up to 160 Mb/s using TCPreplay. We measure the average bit rate generated by TCPreplay and the number of dropped packets registered by Snort. The results are reported in Fig. 8b. On our hardware, in both the scenarios Snort is able to process 100% of packets up to 20–25 Mb/s, then it starts dropping packets. The behavior of the original Snort and the patched Snort is the same up to 70–80 Mb/s with a 10% margin. By increasing the bit rate of the network traces up to 160 Mb/s, the patched Snort performs slightly worse at a 25% of dropped packets against 22% of the original Snort. However, both values are too high to be acceptable in real contexts. This means that in the bandwidth range manageable by Snort, our framework offers performance similar to the unmodified version. It is worth remarking that our results provide a lower bound to the real performance of the prototype, because we run both Snort and the EA on the same host.



0

Fig-4 Snort: original vs patched

Table -1 Time required by state migration activities

State	Average(ms)	б(ms)	Peak (ms)
State import	15	115	17
State export	35	1	39
Complete state migration	421	191	771
Network roaming	8847	3497	13,309

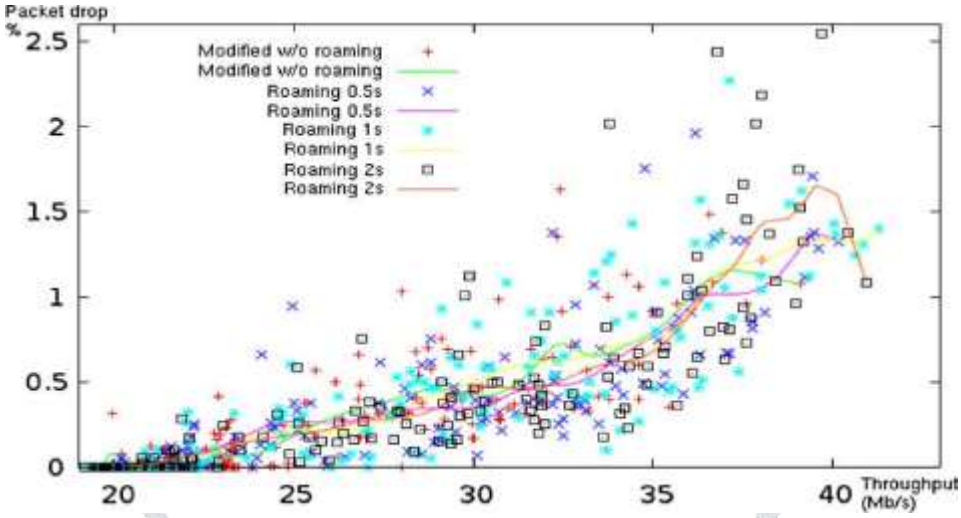
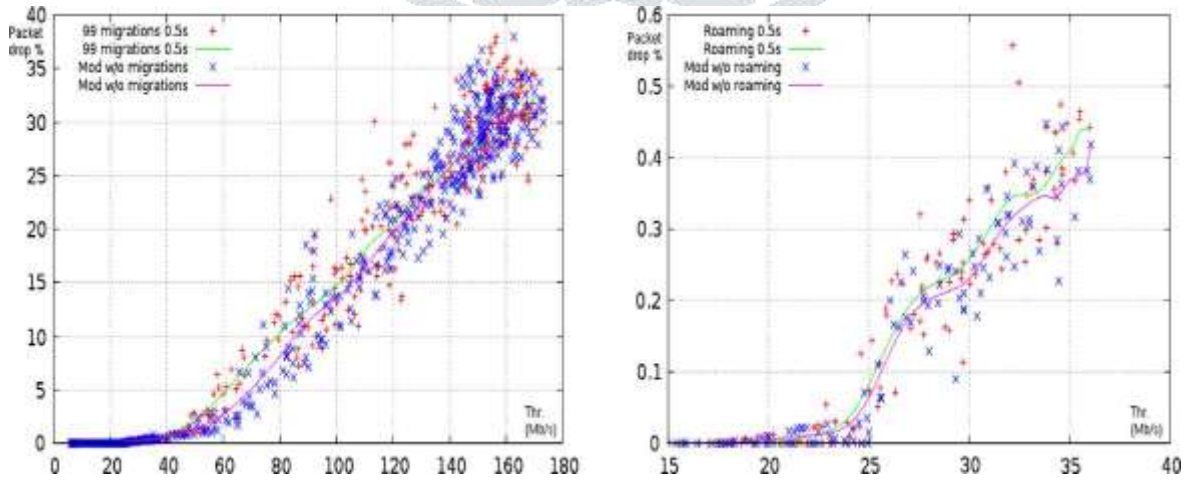


Fig-5 Patched Snort performance with and without roaming for both traces

The goal of the second set of experiments is to determine the impact of state migration on framework performance. In addition to the traffic traces used in the previous experiment, we prepare several traces simulating the migration of different mobile nodes. These traces contain some TCP segments exchanged between the mobile node and the correspondent node, the binding update packets used by Mobile IPv6, and the state.export requests sent by the foreign external agent to the home external agent. We focus on the state.export mechanism due to its heavier impact on Snort than state.import.

We replay the registered network traffic at different speeds ranging from about 20 Mb/s to 40 Mb/s, a small range around which Snort starts to drop packets. Then, we replay the synthetic network traces by simulating mobile nodes roaming at regular intervals: 0.5 s, 1 s and 2 s. We measure the number of packets dropped by Snort and the actual average bit rate generated by Tcpreplay. The results are plotted in fig. 5, where we can appreciate that there is no clear difference in performance for different roaming intervals. This reflects the low impact of the state migration on snort performance. In particular, the state.export operation keeps snort busy for 10-30 ms, that is a limited time period during which received packets fill the buffer. As soon the operation ends, snort processes events and it frees the buffer prior to the next roaming.



a. Patched Snort: w/o roaming vs roaming Every 0.5 seconds

b. Patched Snort: Zoom in the 15-40 Mbps

Fig 6. Impact of state migration

The last set of experiments aims to verify whether the impact of the state migration process could become more distinguishable at higher bit rates. We compare the performance of our framework in the case of two migrations per second and in the case of no migration at all, and consider a range from about 10 Mb/s up to 170 Mb/s. The results are shown in fig 6-a and in 6-b that is a zoom of the range 15–35 Mb/s. While the average results obtained in the two scenarios are really close to each other, the impact of the state migration process is evidenced by the dispersion of measures than is larger than that related to the absence of state migration. These results confirm the effectiveness and the validity of the proposed solutions and prototype.

5. **Conclusion**

We describe a new attack, called mobility-based dodging, that can be used to perform stealth network intrusions and that is undetectable by state-of-the-art NIDS architectures. This attack is not due to design or implementation flaws, but it is a strategy combining fragmentation of a malicious payload and node mobility. We also propose an original solution to mobility-based dodging and implement it in a prototype as an extension of the popular Snort software. Several experimental results confirm the efficacy and the efficiency of the proposed solution for several protocols offering network mobility.

References:

[1]P S Rath , N K Barpanda, R P Singh , a a prototype multiview approach for reduction of false alarm rate in NIDS , vol.5 published in IJCNCS, in march 2017 p-49-59.

[2]T. Alpcan, C. Bauckhage, A.D. Schmidt, A probabilistic diffusion scheme for anomaly detection on smartphones, in: P. Samarati, M. Tunstall, J. Posegga, K. Markantonakis, D. Sauveron (Eds.), Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices, Springer, Berlin, DE, 2010, pp. 31–46.

[3]M. Becher, F. Freiling, J. Hoffmann, T. Holz, S. Uellenbeck, C. Wolf, Mobile security catching up? Revealing the nuts and bolts of the security of mobile devices, in: D. Frincke (Ed.), Proc. Int. Symp. Security and Privacy, SP'11, IEEE, Los Alamitos, CA, 2011, pp. 96–111.

[4]I. Butun, S.D. Morgera, R. Sankar, A survey of intrusion detection systems in wireless sensor networks, Commun. Surv. Tutorials 16 (1) (2014) 266–282.

[5]L.D. Carli, R. Sommer, S. Jha, Beyond pattern matching: a concurrency model for stateful deep packet inspection, in: Proc. 21st Conf. Computer and Communications Security, SIGSAC'14, ACM, New York City, NY, 2014, pp. 1378–1390.

[6]M. Colajanni, D. Gozzi, M. Marchetti, Enhancing interoperability and stateful analysis of cooperative network intrusion detection systems, in: R. Yavatkar, D. Grunwald, K. Ramakrishnan (Eds.), Proc. 3rd Int. Symp. Architectures for Networking and Communication Systems, ANCS'07, ACM, New York City, NY, 2007, pp. 165–174.

[7]M. Colajanni, M. Marchetti, A parallel architecture for stateful intrusion detection in high traffic networks, in: G. Carle (Ed.), Proc. 1st Workshop on Monitoring, Attack Detection and Mitigation, MonAM'06, IEEE, Los Alamitos, CA, 2006, pp. 9–16.

[8]M. Colajanni, L.D. Zotto, M. Marchetti, M. Messori, Defeating NIDS evasion in mobile IPv6 networks, in: L. Bononi, A. Banchs (Eds.), Proc. 1st Int. Symp. World of Wireless Mobile and Multimedia Networks, WoWMoM'11, IEEE, Los Alamitos, CA, 2011, pp. 1–9.

[10] M. Colajanni, L.D. Zotto, M. Marchetti, M. Messori, The problem of NIDS evasion in mobile networks, in: T.E. Ghazawi, L. Fratta (Eds.), Proc. 4th Int. Conf. New Technologies, Mobility and Security, NTMS'11, IEEE, Los Alamitos, CA, 2011, pp. 1–6.

[11] M. Curti, A. Merlo, M. Migliardi, S. Schiappacasse, Towards energy-aware intrusion detection systems on mobile devices, in: Proc. 1st Int. Conf. High Performance Computing and Simulation, HPCS'13, IEEE, Los Alamitos, CA, 2013, pp. 289–296.

[12] P. Garcia-Teodoro, J.E. Diaz-Verdejo, G. Macia-Fernandez, E. Vazquez, Anomaly-based network intrusion detection: techniques, systems and challenges, Comput. Secur. 28 (1) (2009) 18–28.

[13] N. Gobbo, F. Palmieri, A. Castiglione, M. Migliardi, A. Merlo, A denial of service attack to UMTS networks using SIM-less devices, IEEE Trans. Dependable Sec. Comput. 11 (3) (2014) 280–291.

[14] N. Golmie, Seamless mobility: are we there yet?, IEEE Wirel Commun. 16 (4) (2009) 12–13.

[15] Y.L. Ho, S.-H. Heng, Mobile and ubiquitous malware, in: Proc. 7th Int. Conf. Advances in Mobile Computing and Multimedia, MoMM'09, ACM, New York City, NY, 2009, pp. 559–563.

[16] M. Khan, A. Ahmed, A.R. Cheema, Vulnerabilities of UMTS access domain security architecture, in: Proc. 9th Int. Conf. Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD'08, IEEE, Los Alamitos, CA, 2008, pp. 350–355.

[17] A.H. Kholidy, E. Abdelkarim, S. Abdelwahed, F. Baiardi, Ha-cids: a hierarchical and autonomous IDS for cloud systems, in: Proc. 5th Int. Conf. Computational Intelligence, Communication Systems and Networks, CICSyN'13, IEEE, Los Alamitos, CA, 2013, pp. 179–184.

[18] R. Lippmann, D. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. Webster, D. Wyschogrod, R. Cunningham, M. Zissman, Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation, in: Proc. Conf. Exp. DARPA Information Survivability, DISCEX'00, IEEE, Los Alamitos, CA, 2000, pp. 12–26.

[19] M. Locasto, J. Parekh, A. Keromytis, S. Stolfo, Towards collaborative security and P2P intrusion detection, in: Proc. 6th Ann. Workshop Information Assurance, IAW'05, IEEE, Los Alamitos, CA, 2005, pp. 333–339.

[20] C.D. Loiola, E. Wagner, D. Lopes, Z. Abdelouahab, B. Froz, Network intrusion detection system based on SOA (NIDS-SOA): enhancing interoperability

between IDS, in: K. Elleithy, T. Sobh (Eds.), Innovations and Advances in Computer, Information, Systems Sciences, and Engineering, Springer, Berlin, DE, 2013, pp. 935–948.

[21] M. Marchetti, M. Messori, M. Colajanni, Peer-to-Peer architecture for collaborative intrusion and malware detection at a large scale, Lect. Notes Comput. Sci. 5735 (1) (2009) 475–490.