International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 4, Issue 4, July - August 2015 **ISSN 2278-6856**

Network Security: Software IDPS versus Embedded IDPS

Y.SENHAJI [1], H.MEDROMI[2]

[1] Architecture System Team Hassan II University of Casablanca ENSEM
Casablanca, Morocco

[2] Architecture System Team Hassan II University of Casablanca ENSEM
Casablanca, Morocco

Abstract

This paper deals with the issue of computer security, which aims to find out which system will provide us the best response time to an intrusion attack. To do this we developed three systems: The first is an application developed with C ++ making the role of an IDPS exploiting the PCAP library. The second is a Python script embedded in a Yun Arduino board and doing the role of an IDS by exploiting RAW socket. The third system is an android application that generates targeted intrusion attacks.

Keywords: Network Security, IDPS, Real Time, Embedded System, Distributed System, Arduino.

1. INTRODUCTION

This research topic focuses on securing computer networks by using IDPS. The goal is to find out which system will provide us the best response time to an intrusion attack. To do this, we develop three systems. Visual C++ software, Python script and an Android simulator. After several attacks, we record the statement of intrusion detection times. The following describes our results.

2. IDPS DEVELOPMENT

To achieve our simulation, we have developed 3 Systems:
The first is an application developed with C ++ making the role of an IDPS exploiting the PCAP library (Called IDPS).
The second is a Python script embedded in a Yun Arduino board and doing the role of an IDS by exploiting RAW socket (Called ARD).
The latter system is an application that generates targeted intrusion attacks (Called Attack Simulator).
Thus, we will initially attack the IDPS then the ARD. After that, we will focus on a system protected by the binomial HIDPSS and ARD and then another system protected by the binomial NIDPSS and ARD.

2.1. Software System (IDPS)

We have developed C++ software which is based on the WINPCAP tool.
WinPcap is an OPEN SOURCE library that allows the capture and the analyses of packet in Win 32 platforms. The choice of the operating system of test, in this case, is not important because the capture is made on rough packages directly from the network interface. The WinPcap library does not use, to reach the network, the primitives of the operating system such as for example the sockets. The figure below illustrates the position of the Capture Agent which sniffs the packages in the core between the network interface and TCP/IP interface.

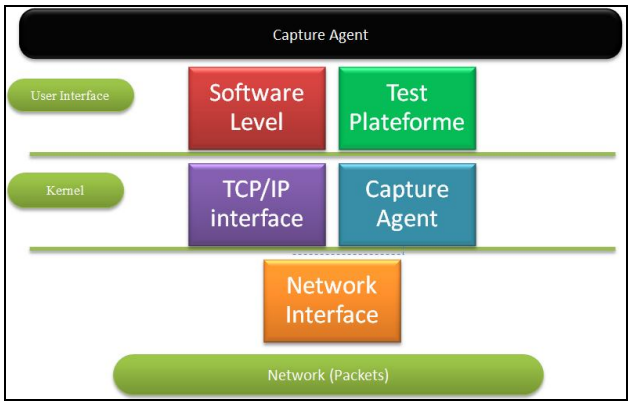

Figure 1 Scenario of the capture agent

The IDPS software is a graphical application which arises as shown in the figure below.

Figure 2 IDPS Interface

By clicking on the button “initialiser” the application detects all network interfaces of the workstation (wired or wireless) and loads them in a drop-down list. By choosing an interface in this list, its information such as name, type, IP address etc., are displayed in a text box in the upper part of the application window .Clicking the Capture button starts the process of listening to the network and preparing the data for the other agents of treatment.

2.2. Embedded System (ARD)

For the embedded part, we use an Arduino Yun.
Arduino Yun is an electronic board that uses the Atmel processor ATmega32U4. Besides of that, it has an additional processor: Atheros AR9331, that turn the Linux distribution OpenWrt Linino.
We use a python script based on RAW Socket for capturing and analyzing the network traffic.

2.3. Intrusion attack simulator

For simulating an intrusion attack, we develop an android application on "Android Studio" that generates intrusion to a specific target identified by its IP address.

Figure 3 Android Attack Simulator

3. SIMULATION RESULTS

3.1. Direct attack of the software system

We carry out a series of attacks directly on the IDPS system to assess its response time to an attack. Thus, we get the results below.

Table 1 Summary of different detection time IDPS

Attack Number	Attack's Time	Attack's detection Time	detection time in ms
1	12:16:24,888	12:16:26,077	0:00:01,189
2	12:16:46,104	12:16:47,124	0:00:01,020
3	12:17:01,595	12:17:02,186	0:00:00,591
4	12:17:17,881	12:17:19,204	0:00:01,323
5	12:17:35,774	12:17:36,275	0:00:00,501
6	12:17:59,674	12:18:00,348	0:00:00,674
7	12:18:13,027	12:18:14,380	0:00:01,353
8	12:18:27,020	12:18:28,418	0:00:01,398
9	12:18:43,650	12:18:44,461	0:00:00,811
10	12:19:02,323	12:19:03,494	0:00:01,171
11	12:19:15,661	12:19:16,541	0:00:00,880
12	12:19:25,365	12:19:26,599	0:00:01,234
13	12:19:35,661	12:19:36,651	0:00:00,990
14	12:20:01,666	12:20:02,688	0:00:01,022
15	12:20:19,918	12:20:20,737	0:00:00,819
		Average	0:00:00,998
		Min	0:00:00,501
		Max	0:00:01,398

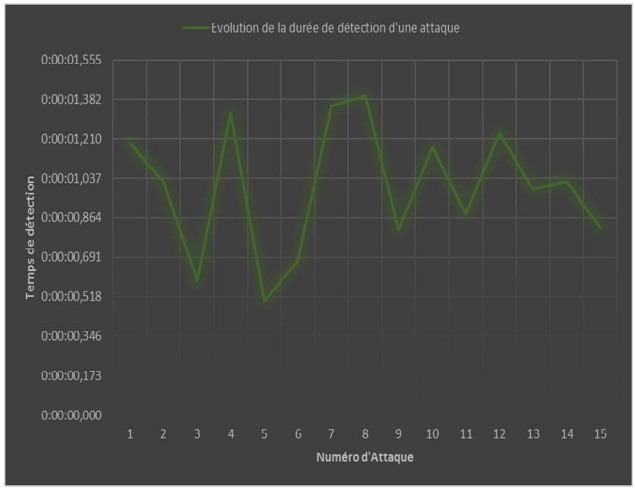

Figure 4 Evolution of the detection time of an attack – IDPS

Of course, this detection time may vary depending on:

- The physical characteristics of our simulation system workstations, network cards, Switch ...
- Network saturation at the time of the attack
- The number of attacks
- The duration between attacks
- The number and nature of security rules
- etc.

We note that in our case the detection time of a threat varies within a range of 501 ms to 1398 ms.

3.2. Direct attack of the embedded system

We carry out a series of attacks directly on the ARD system to assess its response time to an attack. Thus we get the results below.

Table 2 Summary of different detection time ARD

Attack Number	Attack's Time	Attack's detection Time	detection time in ms
1	14:16:04,043	14:16:04,649	0:00:00,606
2	14:16:12,919	14:16:13,600	0:00:00,681
3	14:16:23,403	14:16:24,027	0:00:00,624
4	14:16:36,756	14:16:37,364	0:00:00,608
5	14:16:52,091	14:16:52,722	0:00:00,631
6	14:17:02,716	14:17:03,355	0:00:00,639
7	14:17:13,792	14:17:14,431	0:00:00,639
8	14:17:25,804	14:17:26,428	0:00:00,624
9	14:17:35,257	14:17:35,882	0:00:00,625
10	14:17:48,736	14:17:49,360	0:00:00,624
11	14:18:02,838	14:18:03,433	0:00:00,595
12	14:18:11,793	14:18:12,418	0:00:00,625
13	14:18:21,948	14:18:22,661	0:00:00,713
14	14:18:37,000	14:18:38,041	0:00:01,041
15	14:19:03,101	14:19:03,711	0:00:00,610
		Average	0:00:00,659
		Min	0:00:00,595
		Max	0:00:01,041

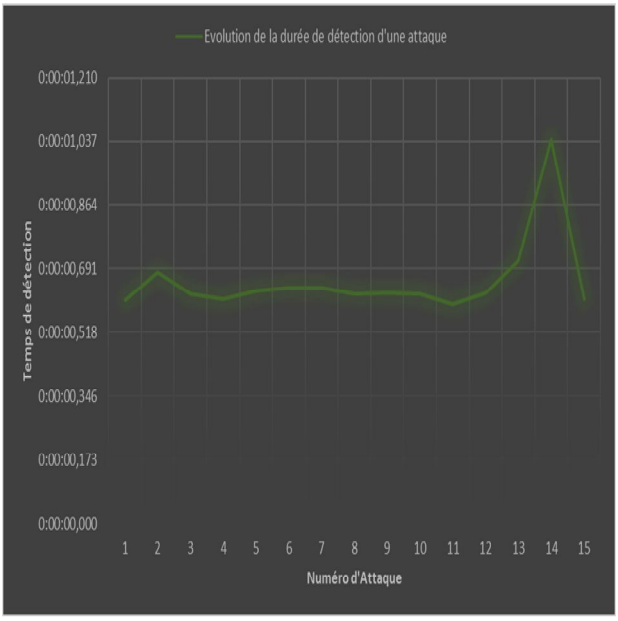

Figure 5 Evolution of the detection time of an attack – ARD

Of course, this detection time may vary depending on the same reasons as those mentioned previously.

We notice that the Arduino system has a better response time and the detection time is almost stable and he does not know the same variations as for the IDPS case.

3.3. Attack of the HIDPS/ARD System

3.3.1. Diagram of the simulation

As a first step, we pair an HIDPS and an ARD as below:

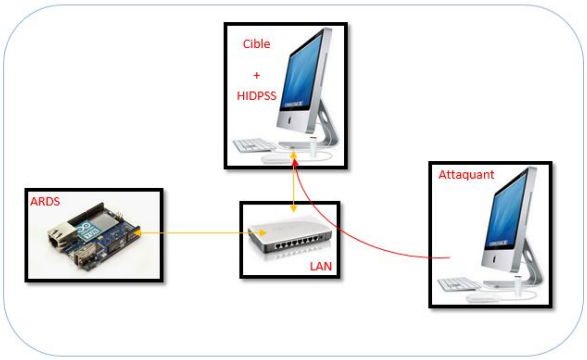

Figure 6 Case HIDPS/ARD

In this case, the target workstation of the attack has the software IDPS installed on it and the ARD is connected to the Local Area Network separately.

3.3.2. Evaluation of the detection time

We carry out a series of attacks on our target. Thus we get the results below.

Table 3 Summary of different detection time - HIDPS / ARD

Attack Number	Attack's Time	ARD Attack's detection Time	ARD detection time in ms	IDPS Attack's detection Time	IDPS detection time in ms
1	18:11:06,455	18:11:11,317	0:00:04,862	18:11:08,004	0:00:01,549
2	18:11:27,000	18:11:35,347	0:00:08,347	18:11:28,300	0:00:01,300
3	18:11:40,699	18:11:46,134	0:00:05,435	18:11:43,011	0:00:02,312
4	18:12:03,000	18:12:13,877	0:00:10,877	18:12:05,350	0:00:02,350
5	18:12:15,613	Not Detected		18:12:17,518	0:00:01,905
6	18:12:32,073	18:12:42,245	0:00:10,172	18:12:33,758	0:00:01,685
7	18:12:42,447	Not Detected		18:12:43,882	0:00:01,435
8	18:12:51,698	Not Detected		18:12:54,022	0:00:02,324
9	18:13:02,571	Not Detected		18:13:04,162	0:00:01,591
10	18:13:11,650	18:13:14,057	0:00:02,407	18:13:13,288	0:00:01,638
11	18:13:25,848	Not Detected		18:13:27,484	0:00:01,636
12	18:13:36,830	18:13:44,440	0:00:07,610	18:13:38,638	0:00:01,808
13	18:13:55,550	Not Detected		18:13:57,967	0:00:02,417
14	18:14:08,577	18:14:23,450	0:00:14,873	18:14:10,946	0:00:02,369
15	18:14:25,003	Not Detected		18:14:26,593	0:00:01,590
		Average	0:00:08,073	Average	0:00:01,861
		Min	0:00:02,407	Min	0:00:01,300
		Max	0:00:14,873	Max	0:00:02,417
		Detection rate	53,33%	Detection rate	100,00%

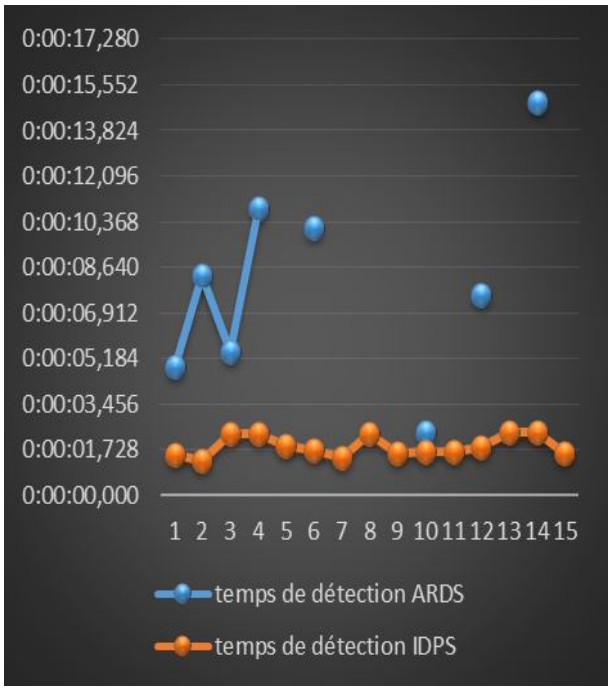

Figure 7 Evolution of the detection time of an attack – HIDPS/ARD

Of course, this detection time may vary depending on the same reasons as those mentioned previously.

But, nevertheless, we note that:

- The threat detection rate HIDPS is 100% at the time the ARD is only 53.3%
- The detection time of the HIDPS is significantly better than that of ARD

Thus, we discover that an embedded system is not in all cases the fastest system. But it depends of security purposes.

3.4. Attack of the NIDPS/ARD System

3.4.1. Diagram of the simulation

As a second step, we pair an NIDPS with an ARD as below:

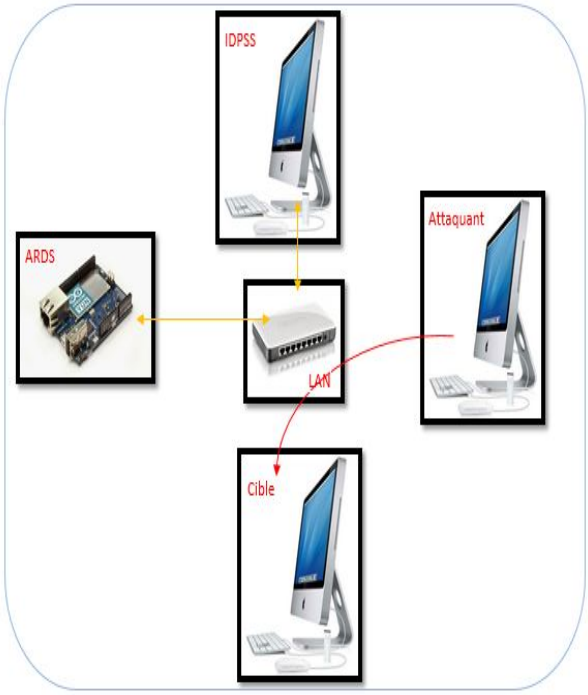

Figure 8 Case NIDPS/ARD

In this case, the target workstation of the attack hasn't the software IDPS installed on it. Both of the IDPS and the ARD are connected separately to the Local Area Network.

3.4.2. Evaluation of the detection time

We carry out a series of attacks on our detection system to assess its response time to an attack. Thus we get the results below.

Table 4 Summary of different detection time – NIDPS / ARD

Attack Number	Attack's Time	ARD Attack's detection Time	ARD detection time in ms	IDPS Attack's detection Time	IDPS detection time in ms
1	18:19:42,000	18:19:52,126	0:00:10,126	18:19:52,983	0:00:10,983
2	18:20:04,884	18:20:25,297	0:00:20,413	18:20:26,446	0:00:21,562
3	18:20:36,928	Not Detected		Not Detected	
4	18:21:04,352	18:21:08,428	0:00:04,076	18:21:10,346	0:00:05,994
5	18:21:23,728	Not Detected		Not Detected	
6	18:21:41,809	18:21:46,160	0:00:04,351	18:21:47,053	0:00:05,244
7	18:22:04,226	18:22:04,624	0:00:00,398	18:22:04,390	0:00:00,164
8	18:22:22,634	Not Detected		Not Detected	
9	18:22:41,651	18:22:46,293	0:00:04,642	18:22:48,025	0:00:06,374
10	18:23:02,000	18:23:03,215	0:00:01,215	18:23:02,221	0:00:00,221
11	18:23:21,000	18:23:33,712	0:00:12,712	18:23:34,670	0:00:13,670
12	18:23:35,674	18:23:49,072	0:00:13,398	18:23:50,895	0:00:15,221
13	18:23:56,002	18:24:03,415	0:00:07,413	18:24:05,091	0:00:09,089
14	18:24:11,773	18:24:15,752	0:00:03,979	18:24:17,276	0:00:05,503
15	18:24:25,143	18:24:30,162	0:00:05,019	18:24:31,456	0:00:06,313
		Average	0:00:07,312	Average	0:00:08,361
		Min	0:00:00,398	Min	0:00:00,164
		Max	0:00:20,413	Max	0:00:21,562
		Detection rate	80,00%	Detection rate	80,00%

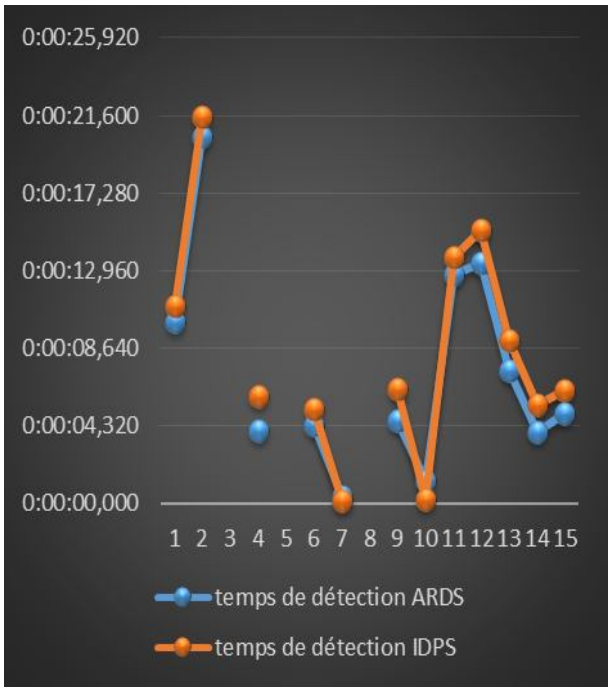

Figure 9 Evolution of the detection time of an attack – NIDPS/ARD

Of course, this detection time may vary according to the same conditions mentioned in the previous section.

But, nevertheless, we note that:

- The detection rates of ARD and NIDPS are not 100%
- The ARD detection time is on average faster than the NIDPSS

Thus, we can notice that unlike the previous case, the embedded system has better performance.

4. CONCLUSION AND FURTHER WORK

In this paper, we perform intrusion attack simulations on IDPS/ARD and we identified the different response time. We noted that the ARD system has a better response time and the detection time is almost stable and he does not know the same variations as for the IDPS case when the system is directly targeted. Moreover, the embedded system has, in the case of an analysis of the network, the fastest response time, when the software system prevails in the case of the direct protection of a host. Nevertheless, the software system offers opportunities for more advanced prevention. These results support the importance of probes combination and distribution in the design of the security architecture. A distribution that covers various scenarios and ensures in all cases the best response time.

As further work we can detail more the securing of IDPS and the securing of their LOG files by developing a network management controlled by its own firewall. Besides of that, we can also study the possibility to create with Arduino Boards a Proxy system to improve the prevention of the embedded system.

References

[1] Karen Scarfone, Peter Mell. “Guide to Intrusion Detection and Prevention Systems IDPS”. NIST. US Departement of Commerce. 2007
[2] Boriana Ditcheva, Lisa Fowler. “Signature-based Intrusion Detection”. University of North Carolina at Chapel Hill. 2005
[3] Rebecca Bace, Peter Mell. “NIST Special Publication on Intrusion Detection Systems”. Infidel, Inc., Scotts Valley, CA - National Institute of Standards and Technology. 2003
[4] Rachid Guerraoui, Lu´ıs Rodrigues, “Introduction to reliable distributed programming’’, Springer-Verlag, August 24, 2005.
[5] David Burgermeister, Jonathan Krier, “les systèmes de détection d’intrusions’’, Developpez.com, July 22, 2006.
[6] Web Site: Arduino - http://www.arduino.cc/.
[7] WINPCAP documentation. Copyright (c) 2002-2005 Politecnico di Torino Dsfg

AUTHORS

[1]**Youssef SENHAJI** received his Degree in engineering from the ENSAM, Meknès, Morocco. In 2009, he joined the System Architecture Team of the ENSEM School, Casablanca, Morocco. His current main research interests IDPS in a Distributed Multi-agents Systems.

[2]**Hicham MEDROMI** received his PhD in engineering science from the Sophia Antipolis University in 1996, Nice, France. He is responsible of the system architecture team of the ENSEM Hassan II University, Casablanca, Morocco. His current main research interests concern Control Architecture of Mobile Systems Based on Multi Agents Systems. Since 2003 he is a full professor for automatic productic and computer sciences at the ENSEM School, Hassan II University, Casablanca