Using a Cluster for Securing Embedded Systems

Mohamed Salim LMIMOUNI, Khalid BOUKHDIR, Hicham MEDROMI, Siham BENHADOU
Equipe Architectures des Systèmes (EAS), Laboratoire d'Informatique, Systèmes et Energies Renouvelables (LISER)
Hassan II University of Casablanca, Ecole Nationale Supérieure d'Electricité et de Mécanique (ENSEM)
Casablanca, Morocco

Abstract—**In today's increasingly interconnected world, the deployment of an Intrusion Detection System (IDS) is becoming very important for securing embedded systems from viruses, worms, attacks, etc. But IDSs face many challenges like computational resources and ubiquitous threats. Many of these challenges can be resolved by running the IDS in a cluster to allow tasks to be parallelly executed. In this paper, we propose to secure embedded systems by using a cluster of embedded cards that can run multiple instances of an IDS in a parallel way. This proposition is now possible with the availability of new low-power single-board computers (Raspberry Pi, BeagleBoard, Cubieboard, Galileo, etc.). To test the feasibility of our proposed architecture, we run two instances of the Bro IDS on two Raspberry Pi. The results show that we can effectively run multiple instances of an IDS in a parallel way on a cluster of new low-power single-board computers to secure embedded systems.**

Keywords—cluster; intrusion detection system; embedded system; security; parallel system

I. Introduction

Intrusion Detection Systems (IDS) were used for many years to protect networks and hosts. And since their design, they have not ceased to play a major role in the defense against intrusions and attacks. They allow analyzing and monitoring the activities on a network or a given machine to detect fraudulent use of resources, log, alert administrators and in some cases react and stop the threat to enforce the security policy.

Despite the progress made in intrusion detection, IDS remain limited in the protection of embedded systems against sophisticated attacks. This limit has prompted us to propose a new architecture to enable IDS to remain among the pillars of security solutions, especially in embedded systems security. We opted for the use of a cluster that offers high performance and better scalability.

A cluster is defined as a group of independent computers linked with a computational network and operating as a single computer [1]. In other words, a cluster is a collection of independent and cheap machines, used together as a supercomputer to provide a solution [1].

Using clusters has many advantages:

- The computers that form a cluster are cheap.
- You can add other nodes to the cluster as needed.
- On clusters, you can use open source software to reduce software costs.
- Clusters allow multiple computers to work together to solve several problems.

In this paper, a background is presented in the second section. The third section gives a short view on related works. The fourth section shows the proposed architecture. In the fifth section, we describe the implementation. The sixth section shows the obtained results.

II. Background

A. Embedded systems

1) Definitions

An embedded system is a microprocessor-based system that is built to control a function or range of functions and is not designed to be programmed by the end user in the same way that a PC is [2].

It's also defined as a computing system which is designed for specific control functions and is embedded as part of the complete device which may include hardware and mechanical parts [3].

2) Reference model

The reference model of embedded systems as illustrated in Figure 1 [4] shows the main components of an embedded system. The Hardware Layer contains the physical components of the embedded system. The System Software Layer and the Application Software Layer contain the software being processed by the embedded system.

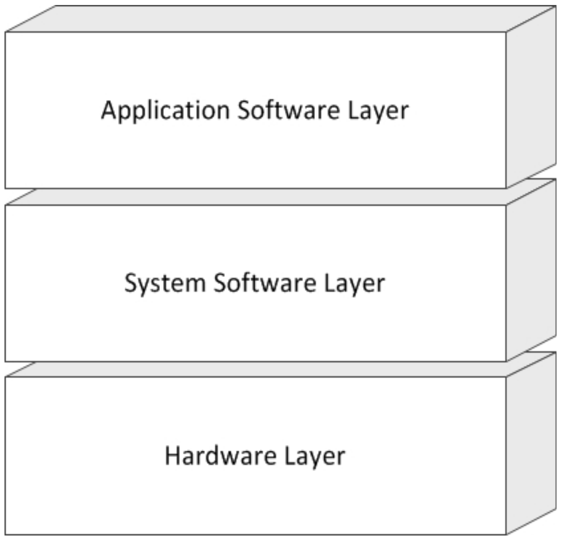


Fig. 1. Reference model of an embedded system

B. Intrusion Detection Systems

1) Definitions

An intrusion detection system is the intrusion alarm of the computer security field [5].

An intrusion detection system consists of an audit data collection agent that collects information about the observed system. This data is either stored or processed directly by the detector [5].

2) Classification

There are two main approaches used by IDS to detect intrusions:

- **Signature-based approach**: An IDS based on this approach monitors the packets in the network and compares them with a database of attributes or signatures of known vulnerabilities. This is similar to the way in which the antiviruses detect malwares. The problem with this approach is the time between the date of vulnerability discovery and the application of the signature associated with it. During this period, the IDS cannot discover attacks exploiting these vulnerabilities.
- **Anomaly-based approach**: An IDS based on the behavioral approach monitors traffic and compares it to an established standard (Baseline). This standard identifies what is "normal" - bandwidth use, used protocols, ports and machines - and alert the administrator or cancel the connection in the case of an anomaly or a different use.

C. Clusters

1) Definitions

A cluster is a single system comprised of interconnected computers that communicate with one another either via a message passing; or by direct, internode memory access using a single address space [6]. We can also define a cluster as a commonly found computing environment consisting of many PCs or workstations connected together by a local-area network [7].

2) Typical architecture

The typical architecture of a cluster is shown in Figure 2 [8]. A node of the cluster can be a single or multiprocessor system, such as a PC, workstation, or Symmetric MultiProcessor (SMP). The nodes must be connected via a Local Area Network (LAN) based on Ethernet, Myrinet or InfiniBand. The cluster middleware offers an illusion of a united system of the independent nodes. Parallel programming environments offer portable, efficient and easy-to-use tools for developing parallel applications.

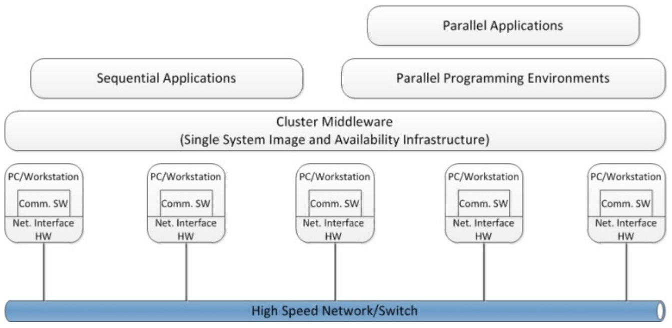

Parallel Applications
Sequential Applications
Parallel Programming Environments
Cluster Middleware
(Single System Image and Availability Infrastructure)
PC/Workstation
Comm. SW
Net. Interface HW
High Speed Network/Switch

Fig. 2. Typical architecture of a cluster

3) Types

There are three varieties of clusters, each one offers different benefits for the user. These varieties are:

- **Load balancing clusters**: used to provide a single interface for a set of resources that can grow arbitrarily. We can imagine a web server that redirects client requests to another node when it has reached its limit of load. This is called "load balancing". Only the node that handles the distribution is visible from the outside.
- **High performance clusters**: they consist of a set of computers linked together to provide maximum power in solving a problem. The heart of these clusters is formed of compute nodes that will receive the code to execute. On smaller clusters we can count ten nodes, while the largest have more than 80 000. The network architecture used to communicate between nodes becomes very heavy, expensive and it limits its performance. You should know also that the ratio of the number of nodes and the performance of this type of clusters is not linear. It is necessary that the program executed is highly parallelizable and that it requires little communication between the computing units.
- **High availability clusters**: the High Availability clusters are built to provide a secure and fault tolerant environment. The redundancy is the most used method. It consists on multiplying the material that could be subject to failure. Server applications are installed the same way on the cluster nodes.

III. Related Work

A. NIDS Cluster

"NIDS Cluster" [9] is a scalable solution based on a set of computers that analyze a traffic flow without sacrificing the detection accuracy. The nodes of this cluster run instances of Bro [10] [11] and share a low-level analysis state to compose a comprehensive picture of the network activity. One of the main objectives of this solution is load balancing between the cluster nodes, one of the advantages of using a cluster of computers.

"NIDS Cluster" is a solution that takes advantage of the load balancing provided by clusters but it does not exploit all the performance offered by cluster nodes. It's also a solution for computer clusters and not for embedded systems.

B. Gnort

Graphics Processing Unit (GPU) have become very powerful and researches have begun to draw power from their ability to do intense calculations for highly parallel operations including intrusion detection [12], and cryptography [13]. Gnort [14] is a high performance intrusion detection system pulling power of graphic processors to accelerate search for patterns in network packets. This work uses the Single Instruction on Multiple Data (SIMD) instructions to turn the Aho-Corassik algorithm which allows the string search in a text, to achieve a maximum bandwidth of 2.3 Gbit/s. This IDS has been implemented on a NVIDIA GeForce 8 Series, which offers many advantages through Compute Unified Device Architecture (CUDA) which is currently the most widely used GPU programming toolbox. It includes a compiler for GPU cores development in an extended C language dialect [15].

Gnort is a solution that takes advantage of the use of GPU but cannot achieve the performance of the use of a HPC cluster as Gnort uses a single NVIDIA GeForce 8 Series. It's also a solution for GPU clusters and not for embedded systems.

C. Distributed platform for intrusion detection based on multi-agents system

As part of research conducted within the Equipe Architectures des Systèmes (EAS) team, a real-time distributed architecture for intrusion detection based on the multi-agent aspect was proposed in 2010 [16]. This architecture consists of two levels of analysis benefiting from agent's reactive and cognitive capabilities. Several agents are distributed at different network points with different roles to detect attacks and intrusions.

This solution takes advantage of using the multi-agent aspect capabilities, but it's a solution for computer networks and not for embedded systems.

IV. PROPOSED ARCHITECTURE

The proposed hardware architecture as illustrated in Figure 3 shows the different components of the cluster:

- **Device**: the device to protect from intrusions and attacks, such us drones, satellites, mobile robots, etc.
- **Manager Card**: the card which manages the other cards of the cluster using the network. The manager card can:
  - ✓ Start the other card's work,
  - ✓ Stop the other card's work,
  - ✓ Monitor the other cards,
  - ✓ Configure the other cards,
  - ✓ Update the other cards,
  - ✓ Share information with the other cards,
  - ✓ Receive logs from the other cards,
  - ✓ Store logs,
  - ✓ Organize logs,
  - ✓ Generate statistics.
- **Worker Cards**: the cards running the IDS instances. The worker cards can:
  - ✓ Analyze the traffic on the secured interface (the interface being monitored),
  - ✓ Examining packets,
  - ✓ Send states to the manager card,
  - ✓ Send logs to the manager card.
- **Network Interfaces**: the link between the cards and the network. They are the hardware that acts as a communication processor and which is responsible for transmitting and receiving packets of data between the cards of the cluster via a network switch.
- **Network**: the Local Area Network (LAN) that connects the cluster nodes (cards).

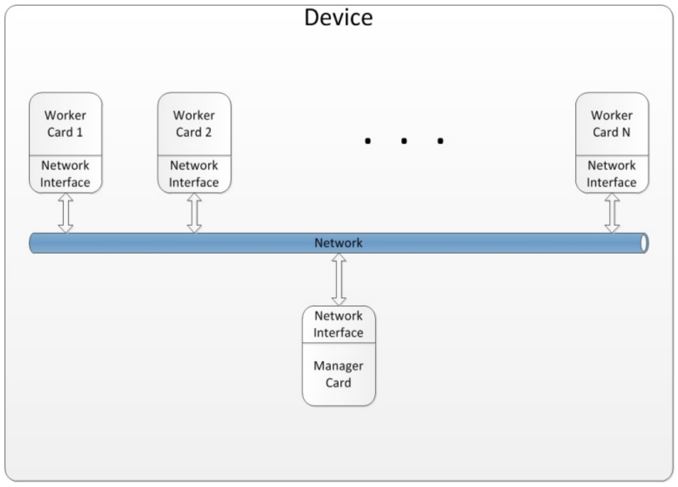


Fig. 3. Proposed hardware architecture

Figure 4 shows the layers of the proposed software architecture:

- **IDS**: the Intrusion Detection System parallelly run on the cards of the cluster and acting as a singular and cohesive entity.
- **Compiler**: the necessaire compiler for running the code source of the chosen IDS.
- **OS**: the Operating System allowing us to exploit the cards.
- **Drivers**: the software allowing us to use the different components of the cards.

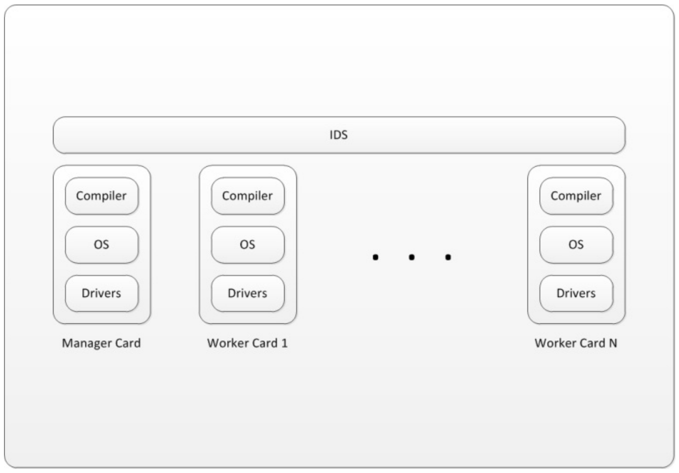


Fig. 4. Proposed software architecture

Figure 5 illustrates the data flow between the different components of the proposed architecture. The manager card affects tasks to the worker cards (traffic analysis, examining packets, etc.). The worker cards send their states and logs to the manager card to be stored, organized and used to protect the device from intrusions and attacks.

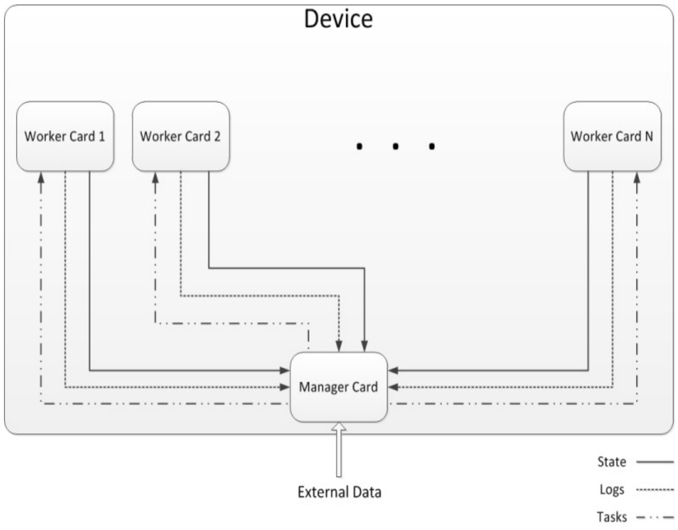


Fig. 5. Data flow in the proposed architecture

V. Implementation

We need at least two nodes if we want to talk about a real cluster. In our case, we built our cluster based on two Raspberry Pi 1 Model B [17]. This model of the Raspberry Pi is characterized by:

- **Processor**: ARM1176JZF-S 700 MHz
- **RAM**: 512MB
- **Power rating**: 3.5W
- **Power source**: 5V
- **Size**: 85.6 mm x 56.5 mm
- **Weight**: 45g

Connectors and main components of the Raspberry Pi 1 Model B are represented in Figure 6 [18].

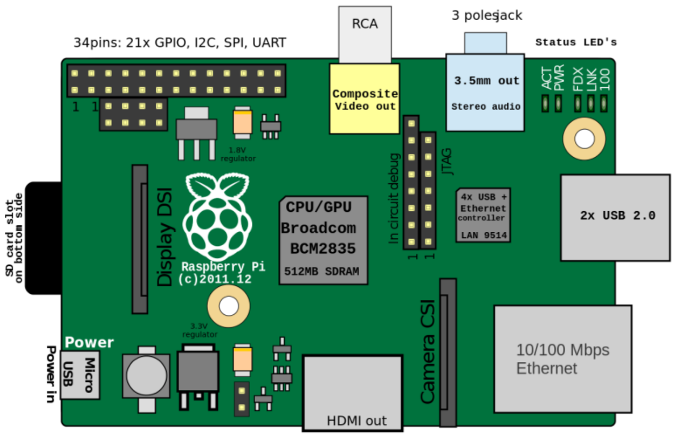


Fig. 6. Raspberry Pi 1 Model B

Figure 7 represents a description of our implementation. We installed the operating system Raspbian [19] on the two Raspberry Pi and we configured them to be on the same network segment. We used Bro [10] [11] as IDS. We installed Bro and its dependencies on the manager card but only Bro's dependencies on the worker card.

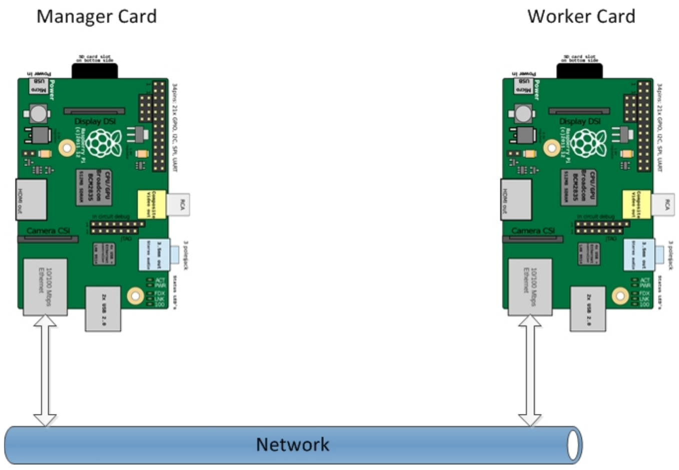
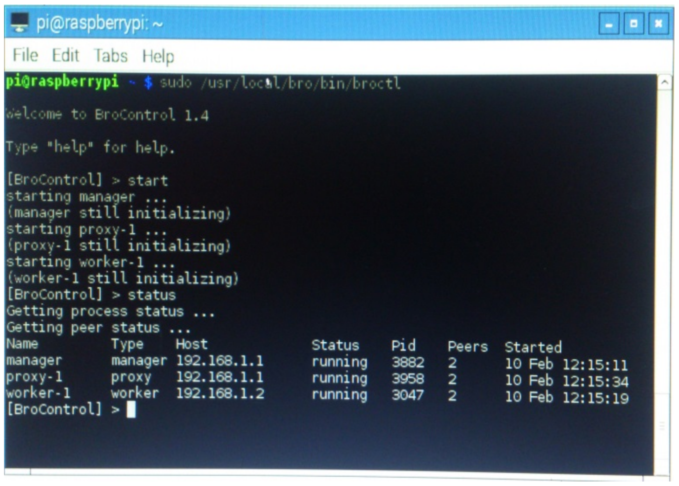


Fig. 7. Implementation

We encountered some challenges while mounting our implementation, the main ones were the non-availability of a Bro pre-built binary package for the operating system Raspbian and the configuration of a password-less communication between the two Raspberry Pi.

VI. Results

By using a display monitor, a keyboard and a mouse with the manager card, we started BroControl which is an interactive interface for managing Bro. We used the "broctl" script to start BroControl. Next, we started our Bro cluster by executing the "start" command on BroControl command-line ([BroControl]). "status" is the command to execute to check that all nodes are running. Figure 8 shows the obtained results after the execution of these commands.



```
pi@raspberrypi ~ $ sudo /usr/local/bro/bin/broctl

Welcome to BroControl 1.4

Type "help" for help.

[BroControl] > start
starting manager ...
(manager still initializing)
starting proxy-1 ...
(proxy-1 still initializing)
starting worker-1 ...
(worker-1 still initializing)
[BroControl] > status
Getting process status ...
Getting peer status ...
Name         Type    Host          Status    Pid    Peers  Started
manager      manager 192.168.1.1   running   3882   2      10 Feb 12:15:11
proxy-1      proxy   192.168.1.1   running   3958   2      10 Feb 12:15:34
worker-1     worker  192.168.1.2   running   3047   2      10 Feb 12:15:19
[BroControl] >
```

Fig. 8. Results

VII. Conclusions and Future Work

The growing need for powerful, faster and cheaper computers in the world increases the use of clusters. Today, clusters are used in different areas (commercial, scientific, etc.). The field of embedded systems security, an area in large changes, also needs to exploit the advantages of the use of clusters.

That is why we have proposed in this paper to use a cluster to secure and protect embedded systems from intrusions and attacks by running multiple instances of an Intrusion Detection System (IDS) on the different nodes of a cluster.

As future work, we would like to carry out detailed performance evaluations and test the realized implementation with real-life cases.

References

[1] S. Aydin and O. F. Bay, “Building a high performance computing clusters to use in computing course applications”, Procedia - Social and Behavioral Sciences, vol. 1, pp. 2396-2401, Feb. 2009.

[2] S. Heath, Embedded Systems Design, 2nd ed., Newnes, 2003.

[3] S. Mittal, “A Survey of Techniques For Improving Energy Efficiency in Embedded Computing Systems”, IJCAET, vol. 6, no. 4, pp. 440-459, 2014.

[4] T. Noergaard, Embedded Systems Architecture, 2nd ed., Newnes, 2013.

[5] S. Axelsson, “Intrusion Detection Systems: A Survey and Taxonomy”, Chalmers University of Technology, Göteborg, Sweden, Technical Report 99-15, 2000.

[6] G. Bell and J. Gray, “What’s Next in High-Performance Computing”, Communications of the ACM, vol. 45, issue 2, pp. 91–95, Feb. 2002.

[7] J. Dongarra, “Trends in high performance computing: a historical overview and examination of future developments”, Circuits and Devices Magazine, IEEE, vol. 22, issue 1, pp. 22-27, Feb. 2006.

[8] R. Buyya, “High Performance Cluster Computing: Programming and Applications”, Prentice Hall PTR, NJ, USA, 1999.

[9] M. Vallentin, R. Sommer, J. Lee, C. Leres, V. Paxson and B. Tierney, “The NIDS Cluster: Scalable, Stateful Network Intrusion Detection on Commodity Hardware”, in Proc. RAID 2007, 2007, p. 107–126.

[10] V. Paxson, “Bro: A System for Detecting Network Intruders in Real-Time”, Computer Networks, Elsevier, vol. 31, issue 23-24, pp. 2435-2463, Dec. 1999.

[11] The Bro Network Security Monitor. [Online]. Available: http://www.bro.org/

[12] N. Jacob and C. Brodley, “Offloading IDS Computation to the GPU”, in Proc. ACSAC’06, 2006, p. 371-380.

[13] D. L. Cook, J. Ioannidis, A. D. Keromytis and J. Luck, “CryptoGraphics: Secret Key Cryptography Using Graphics Cards”, in Proc. RSA Conference 2005, 2005, p. 334-350.

[14] G. Vasiliadis, S. Antonatos, M. Polychronakis, E. P. Markatos and S. Ioannidis, “Gnort: High Performance Network Intrusion Detection Using Graphics Processors”, in Proc. RAID 2008, 2008, p. 116-134.

[15] V.V. Kindratenko, J. J. Enos, G. Shi, M. T. Showerman, G. W. Arnold, J. E. Stone, J. C. Phillips and Wen-mei Hwu, “GPU clusters for high-performance computing”, in Proc. CLUSTER’09, 2009, p. 1-8.

[16] D. Raoui, S. Benhadou and H. Medromi, “New distributed platform for intrusion detection based on multi-agents system”, Journal of Engineering and Technology Research, vol. 2, issue 10, pp. 200-206, Oct. 2010.

[17] Raspberry Pi 1 Model B. [Online]. Available: http://www.raspberrypi.org/products/model-b/

[18] File:Drawing of Raspberry Pi model B rev2.svg. [Online]. Available: http://en.wikipedia.org/wiki/File:Drawing_of_Raspberry_Pi_model_B_rev2.svg

[19] FrontPage - Raspbian. [Online]. Available: http://www.raspbian.org/