IMoViS: A SYSTEM FOR MOBILE VISUALIZATION OF INTRUSION DETECTION DATA

Andrea SANNA and Claudio FORNARO

Introduction

Intrusion detection applications often produce large amount of data. The visualization of this information is a key task in order to allow the user to effectively detect attacks and intrusions. Information visualization is an important sub-discipline within the field of scientific visualization and focuses on visual mechanisms designed to communicate clearly to the user the structure of information and to *facilitate* the access to large data repositories. Information visualization helps people in dealing with all this information by taking advantage of the visual perception capabilities of the human being. By presenting the information in a more graphically oriented fashion, it is possible for the human brain to take advantage of its perceptual system in the initial information acquiring, rather than immediately relying entirely on the cognitive system. Some of the most important papers in the field are collected in a book.[1] Information visualization algorithms require merging of data visualization methods, computer graphics solutions, and graphical interface design.

A new challenge in information visualization is the use of Personal Digital Assistant (PDA) devices. PDAs were originally designed as personal organizers and their use was limited for a number of reasons, among them small screen size, low computational resources, limited wireless communication bandwidth, reduced interaction capability. But today's PDAs are efficient pocket computers and thus can be effectively used for remote monitoring purposes.

This paper proposes an integrated architecture, which is used by a security manager to remotely monitor large buildings for computer intrusion attempts; the front-end is just a PDA. The system is composed of two different parts:

- Intrusion detection and information collection system;
- Visualization interface.

Information collection is achieved by monitoring the network traffic of the LAN under control. The program used intercepts all the network traffic and scans it for traces of a possible attack.

For the Intrusion Detection part, an architecture based on existing software has been defined. Its purpose is to set some preprocessing steps for the information data, mainly cleaning and reduction, in order to meet the limited computational and communication resources of the mobile devices.

From the visualization point of view, this paper presents a graphical interface designed for PDAs. Data related to the building are organized hierarchically; this allows the user to see a global view, as well as detailed information concerning every machine located in each office/room of the building. Moreover, a tool for designing building maps is proposed. An arbitrary number of machines can be placed in every office and a set of information items, such as IP address, operating system, user name, and office telephone number, is associated with every machine.

The paper is organized as follows. The first section reviews main concepts of intrusion detection and some examples of visualization on PDAs, while the second explains in detail the proposed architecture and shows how intrusions are displayed on a PDA. Finally, the last section presents some remarks concerning performance issues of the proposed framework.

Background

Intrusion Detection Systems

To hinder the attacks performed against computer systems, information and security professionals turn to *Intrusion Detection Systems* (IDSs) to set up an active defense-in-depth strategy. A *firewall* is an essential and important part of network security but it does not have the ability to detect hostile intent. IDSs are classified as Host IDSs and Network IDSs. While the former are installed directly on the machine to be monitored and are intended just for that machine, the latter perform surveillance for the whole network. The most common scheme for a NIDS is composed of at least one sensor that intercepts the network traffic and an analysis engine. Different machines may handle alerting and attack analysis. An analysis engine considers packet protocol flags, source and destination addresses, sequential numbers, and application payload, such as email messages or web requests. Common attack signatures consist of strings to search for in the payload or network packet parameters. In addition, analysis may be applied to the whole TCP connection rather than individual packets or even include correlation of the connection to those occurred earlier or elsewhere on the network. The advantage of a NIDS is the ability to protect an entire network with a single machine, in a transparent way with respect to network hosts, with no impact on the network architecture and performance, detecting not only actual attacks, but also potential ones. Traffic wiretapping (sniffing) is possible because of the way data are transmitted over a LAN. Unencrypted data are split in packets (frames) and each one is directed to a particular NIC, which is identified by an address (a 48-bit number) called the MAC Address. No two NICs are manufactured with the same MAC Address (however, some NICs allow to change it). When a packet travels over the LAN, all the NICs that see it read the embedded destination MAC Address, but only the NIC whose MAC matches the one in the packet reads the whole packet and forwards it to the machine network protocol stack (e.g. TCP/IP) to be processed. On the contrary, NIDS' sensors use a NIC set up in *promiscuous mode*, so that they read all the network traffic, all the packets, independently of the destination MAC Address. The NIDS can, thus, have a big picture of the entire network and is able to recognize attacks conducted to every machine connected to the LAN.

Visualization on PDA Devices

PDAs, and mobile devices in general, have proven to be very effective tools for a large range of disciplines. The interested reader may consult a good survey on remote visualization.[2] This section briefly reviews the main areas where mobile devices (and among them PDAs) are used.

2D, and 3D graphics in particular, can be computed directly on PDAs by using operating system's APIs (for example see the article of Fairuz Shiratuddin and co-workers)[3] or Java Virtual Machines that allow designing graphical applications in Java, or by means of some ad-hoc software. Elite (http://home.rochester.rr.com/ohommes/elite/) is a fast 3D rendering engine for small devices running Java. This engine provides a framework for creation and display of 3D wireframe models. PocketGL (http://pierrel5.free.fr/pocketglb/) for Pocket PC (written in C and C++) allows drawing 3D objects and managing 3D transformations.

On the other hand, realistic visualization of large and complex models is not yet possible due to the computational limitations of PDAs. To overcome this problem, solutions for hardware-accelerated remote rendering have been recently presented.[4,5] These works aim to deploy hardware resources of centralized systems in order to allow the user to visualize and investigate large data set models on PDAs. This kind of application is strictly related to the problem of transmitting video data streams to remote devices.

One of the first applications of PDAs was for tourist and transportation purposes. Mobile devices can guide people through both real and virtual sites (museums, archeological sites, etc). Oppermann and Specht[6] and Vlahakis and colleagues[7] present two examples of virtual guides for museums and archeological sites, respectively. On the other hand, PDAs have been used to visualize maps of Virtual Harlem:[8] maps allow retrieving information about current location, as well as moving to any place in Virtual Harlem. Preim and co-workers present a mobile information system for public transportation:[9] this system helps travellers to find the best public transport to reach a destination and to estimate the time needed to arrive.

Another field of application where the use of mobile technologies seems very promising is telemedicine. Telemedicine scenarios include in-hospital care management, remote teleconsulting, collaborative diagnosis, and emergency situations handling. Different types of information need to be accessed by means of heterogeneous client devices in different communication environments in order to enable high quality continuous sanitary assistance delivery wherever and whenever it is needed. Personal mobile telemedicine systems using wireless communication links have been employed in several applications and have been extensively studied.[10,11,12] Java, XML, and XSL technologies are used by some reserhers to ensure software portability and effective data presentation on heterogeneous access devices.[13]

Education, entertainment, and training are fields where new technologies have always found large application. For instance, students of an elementary school can observe phenomena on a large display size and PDAs are used to aid data collection.[14] Many commercial programs are available for entertainment on mobile devices; a nearly exhaustive list of reviews and articles concerning games for PDAs can be found at http://www.pdarcade.com/.

The Proposed Architecture

This section provides an overview of the proposed architecture. Details concerning components of the whole system are presented in respective subsections.

A complete scheme is shown in Figure 1. Three main components can be identified:

- Snort
- Guardian
- Portable Intrusion Visualization Interface (P.I.V.I)

Snort (http://www.snort.org/) IDS is able to monitor network traffic. It uses a set of rules to identify attacks and intrusions (the terms attack and intrusion in this context refer to a violation of *Snort* rules). *Snort* stores each attack into a database (Snort DB) shared with the second component of the system: *Guardian*.

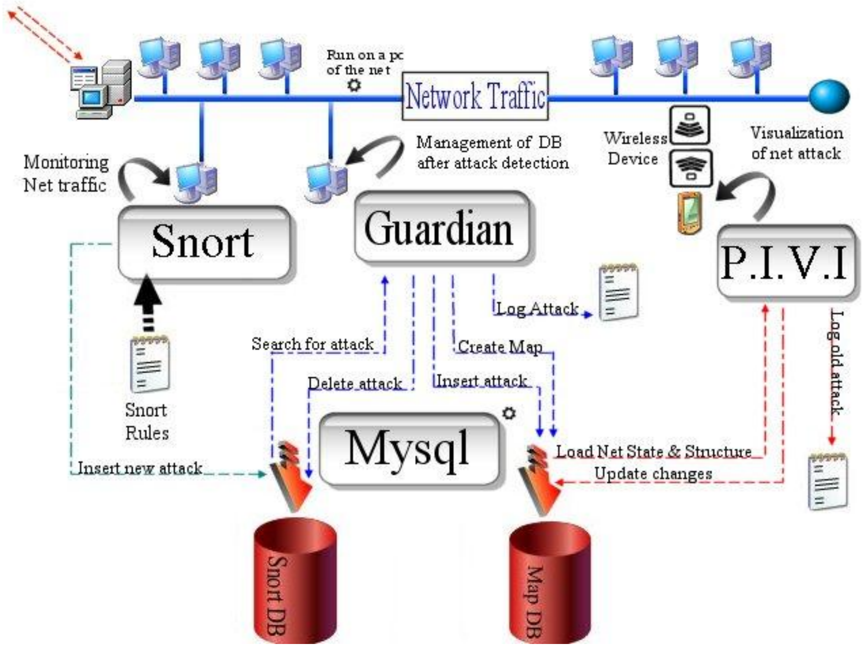


Figure 1: IMoViS Architecture.

Guardian is a program devoted to interface *Snort* to the visualization application and is not necessarily located on the same machine as *Snort*. *Guardian* manages the Snort DB in order to search for and delete attacks and produces a specific database (Map DB) used by the visualization application. *Guardian* can also produce log files useful for saving attack information and details.

P.I.V.I. is the visualization application. It loads a description of the building to be monitored, as well as the network status information from Map DB. The description of the building can be organized at different levels of detail and contains a graphical representation of the site under analysis. It is possible to specify which data have to be monitored, such as IP address, user name, and office telephone number. *P.I.V.I.* allows the user to delete already "processed" attacks from Map DB and to save log files of them.

MySQL (http://www.mysql.com/) has been chosen for database management system. MySQL has good performance in terms of speed and reliability and is open source. JDBC drivers allow the interaction between MySQL and the three system components: *Snort*, *Guardian*, and *P.I.V.I.*

Guardian and *P.I.V.I.* have been developed in Java and need a Java Virtual Machine (JVM) to be executed. Among the different Personal Java Application Environment implementations available for PDAs, the *Insignia Jeode Runtime* (http://www.insignia.com/) has been preferred to other implementations for its high adherence to Sun Microsystems Personal Java 1.2 specifications and for its advanced performance in executing Java byte-code. Jeode Runtime can be used both as a Pocket Internet Explorer plug-in to run Java applets from a Web page and as a stand-alone JVM to run Java applications.

Description of Snort

Snort is a Network Intrusion Detection System well known among the computer security professionals. Free and open source, it is rapidly becoming the tool of choice for Network Intrusion Detection. Unix and Windows versions are available and a huge and active enthousiastic community of users contributes to the development of filters and rules (signatures) suited to discover intrusion attempts (and, of course, makes them free to other users). It is considered one of the most advanced intrusion detection systems, free, but with the quality of a commercial product.[15] *Snort* is based on the sniffing libraries libpcap/winpcap.[16,17] The detection engine uses detection rules written using a simple but powerful language that describes per packet tests and actions: logging, content pattern matching, and attacks and probes detection. *Snort* has real-time alerting capability: alerts are sent to syslog, via SMB messages, or written to a separate “alert” file. A database may also be used to store them. *Snort* is configured using command line switches and Berkeley Packet Filter commands.[18] Third party add-ons may be used to simplify the administration tasks.

Description of Guardian

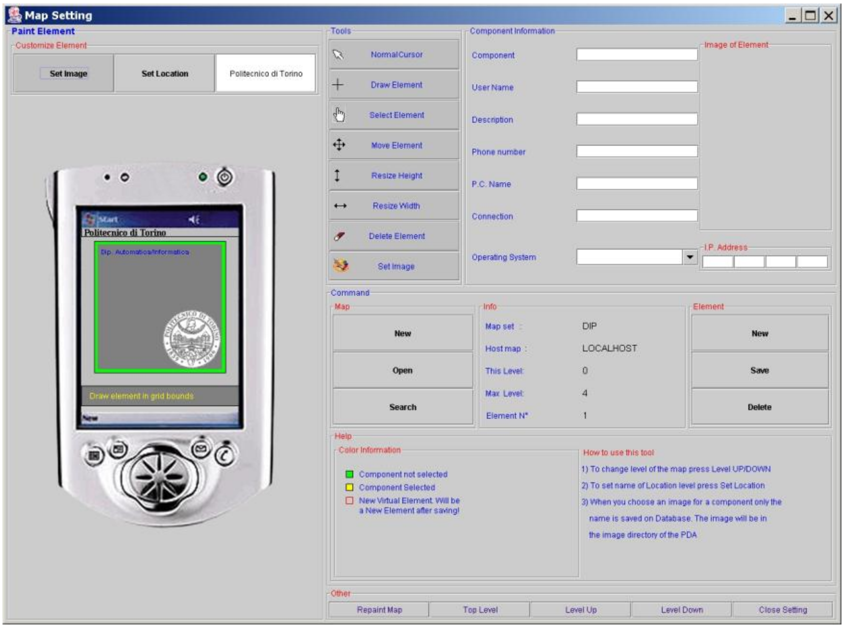
Guardian has two main purposes:

1. Interfacing *Snort* to *P.I.V.I.*;
2. Providing an effective tool for map design.

Every time a rule violation is detected, *Snort* pushes a record into the database (Snort DB in Figure 1). *Guardian* manages data produced by *Snort* and generates a second database (Map DB in Figure 1) containing a sort of “meta-data” used for the visualization process. Only a subset of the information produced by *Snort* is required by *P.I.V.I.*

Guardian periodically polls Snort DB in order to detect new data insertions. Two cases may occur:

- A new attack or violation is detected;
- A new event is detected.



Map set : DIP
Host map : LOCALHOST
This Level: 0
Max Level: 4
Element N° 1

🟩 Component not selected
🟨 Component Selected
🟥 New Virtual Element. Will be a New Element after saving!

How to use this tool
1) To change level of the map press Level UP/DOWN
2) To set name of Location level press Set Location
3) When you choose an image for a component only the name is saved on Database. The image will be in the image directory of the PDA

Figure 2: The Interface for Map Design.

The first case causes a data insertion in Map DB in order to generate a new visual event; the second case concerns an already active attack and just the information regarding it is updated in Map DB.

Snort inserts a new record each time a security rule is violated. After an attack has been processed, *Guardian* deletes the corresponding record in Snort DB keeping the database size near constant. The deletion of a set of records would lead to a loss of information; in order to avoid this, the user can configure *Guardian* to save the deleted information into a log file (see Figure 1).

The tool for map design, shown in Figure 2, is entirely written in Java. Three main zones can be identified: painting area (left part), information insertion area (upper-right part), and command area (lower-right part). The painting area allows defining rectangular areas over a grid. Rectangle sizes and spatial coordinates can be intuitively drawn by the mouse. Each rectangle is an *element* that can be selected, moved, resized, and deleted. An image can be placed inside a rectangle and the related information is introduced in the information area.

When a new map is designed, a new database is created using the *New* button of the command area. Existing databases can also be modified and deleted.

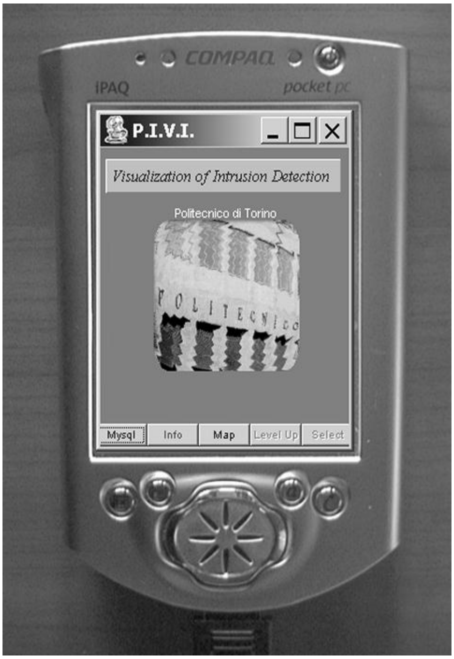


Figure 3: The Introductory Screenshot of the PDA.

The description of the building is organized in different levels. *Level up* and *Level down* buttons allow moving up and down through the levels' hierarchy.

Description of P.I.V.I

This section will provide details of *P.I.V.I.* and, in particular, of the graphical interface. The visualization application has been developed and tested on a Personal Digital Assistant Device Compaq iPaq H3630 equipped with the Microsoft PocketPC operating system. Basic features of this PDA are: 206 MHz Intel StrongArm CPU, 4096 colors TFT LCD display, 240 x 320 pixels (2.26 x 3.02 inches) resolution touch screen, 32 MB RAM and 16 MB Flash ROM. Although the graphical interface has been tailored for PDA devices, *P.I.V.I.* can be used on every device equipped with a Java Virtual Machine (multi-channel interface).

The introductory screenshot of *P.I.V.I.* is shown in Figure 3. The main problem involved in designing graphical interfaces for mobile devices is the display size.[19] In particular, large size images have to be accurately managed.

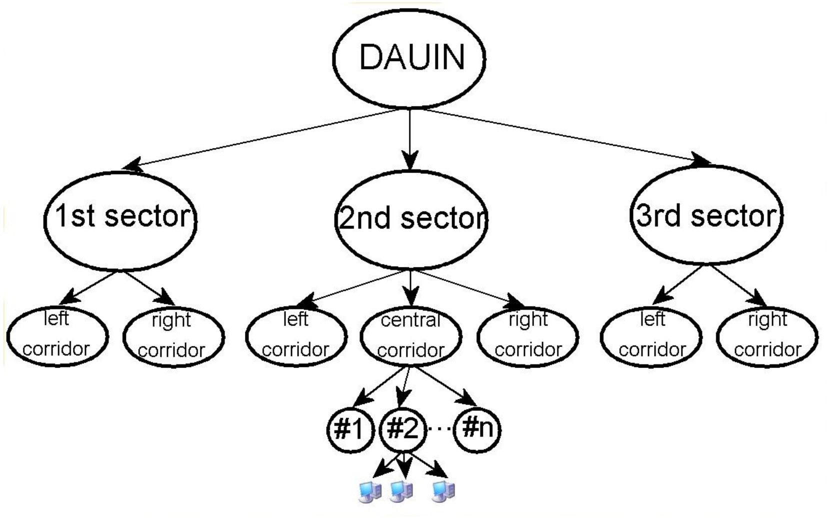


Figure 4: DAUIN's Hierarchical Representation.

Two solutions are possible:

- Displaying an image larger than the display, allowing the user to move and zoom it;
- Representing the information by means of a set of hierarchical images.

The former approach is easier to implement and is recommended when the user must have a global view at maximum level of detail; for instance, a radiograph has to be analyzed in its entirety. In these cases the user has to be able to move the image in order to "browse" it. This operation can require a lot of computational power and turns out to be particularly slow on low-end PDAs.

The latter strategy is appropriate for applications where hierarchical organizations of information may be obtained. The proposed intrusion detection system has been devised to monitor the network traffic within the aurthors' Computer Science Department (DAUIN). The hierarchical representation of the building is shown in Figure 4.

DAUIN is divided in three sectors; offices are organized as two rows (left and right) in the first and in the third sector, while, in the second sector, two corridors divide three rows of offices. Each office can contain arbitrary number (generally from one to four) machines.

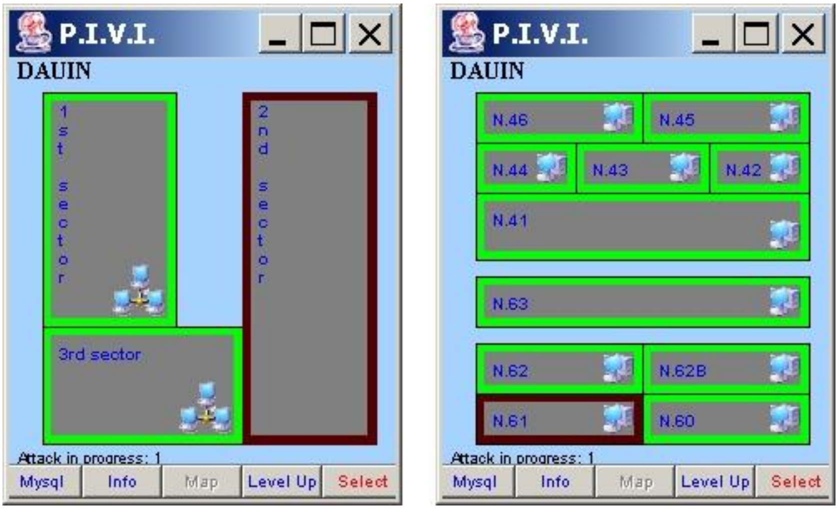


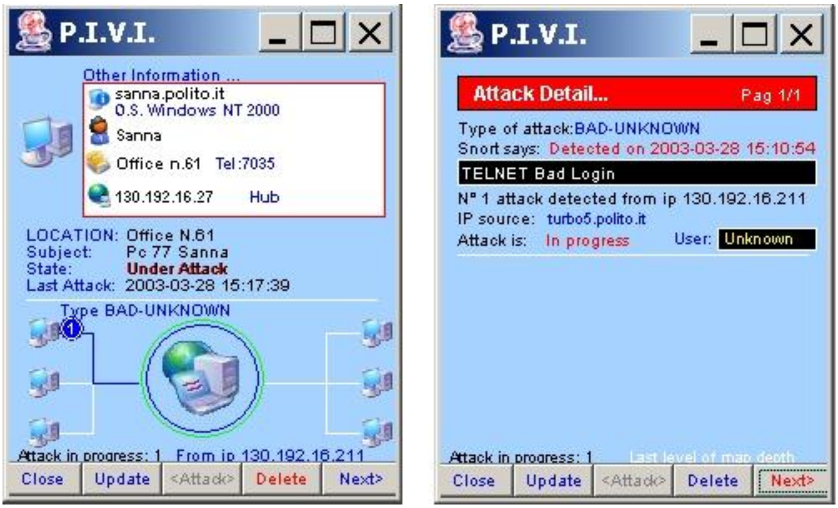
Figure 5: Two Screenshots of an Attack.

The starting panel (see Figure 3) allows the user to configure a set of parameters that are used to connect to the database (Map DB in Figure 1). *P.I.V.I.* loads the map of a building from Map DB and makes queries for new attacks.

The colors used to draw each building contour focus the attention of the user where a new attack has been detected.

An example is shown in Figure 5. The left image shows an attack to a machine placed in the department area labelled “*2nd sector*” (the border of this sector is darker in order to denote an attack). More details may be found by browsing the lower level related to the red part. The lower level is shown in the right image of Figure 5; here the attack involves a machine placed in office N. 61. Selecting this office, it is possible to retrieve all the details of the attack.

Figure 6 (left image) shows all the information about the attack: IP address and operating system of the PC, user name and office telephone number, and date and hour of the attack. The user can get further information by pressing the *Next* button placed at the right lower corner (see Figure 6, right image). The kind of attack (in this case a telnet bad login) is shown together with the IP address and the name of the remote machine performing the intrusion and, when available, the name of the user on the remote machine. A set of pages is automatically generated when the number of attacks affecting a machine grows.


P.I.V.I.

Other Information ...
sanna.polito.it
O.S. Windows NT 2000
Sanna
Office n.61 Tel:7035
130.192.16.27 Hub

LOCATION: Office N.61
Subject: Pc 77 Sanna
State: **Under Attack**
Last Attack: 2003-03-28 15:17:39

Type BAD-UNKNOWN

Attack in progress: 1 From ip 130.192.16.211

P.I.V.I.

Attack Detail... Pag 1/1
Type of attack:BAD-UNKNOWN
Snort says: Detected on 2003-03-28 15:10:54
TELNET Bad Login
N° 1 attack detected from ip 130.192.16.211
IP source: turbo5.polito.it
Attack is: In progress User: Unknown

Attack in progress: 1 Last level of map depth

Figure 6: Datails of Detected Attack.

An alarm is activated every time a monitored machine is attacked; this alarm is outlined by changing the color of the map from the lowest level of the hierarchy (“office level”) to the highest level (“building level”). The user can deactivate the alarm by selecting a level via the *Select* button (refer to Figure 5) and updating the status of the alarm. In this way, the alarm will be deactivated for the selected level and for all lower levels; for instance, a set of alarms concerning a unique part of the building can be deactivated at the same time. Moreover, the user can delete all the information related to each attack from Map DB; in this case a log file containing the deleted information can be generated.

Performance Issues

In order to test the proposed system the authors have developed a program able to generate “customizable” attacks over a LAN. The number of packets (every packet represents a violation to the *Snort*’s rules) and the delay between packets are changed in order to measure the number of lost packets (a packet is denoted as lost when the corresponding rule violation has not been detected and processed). Two tests have been performed:

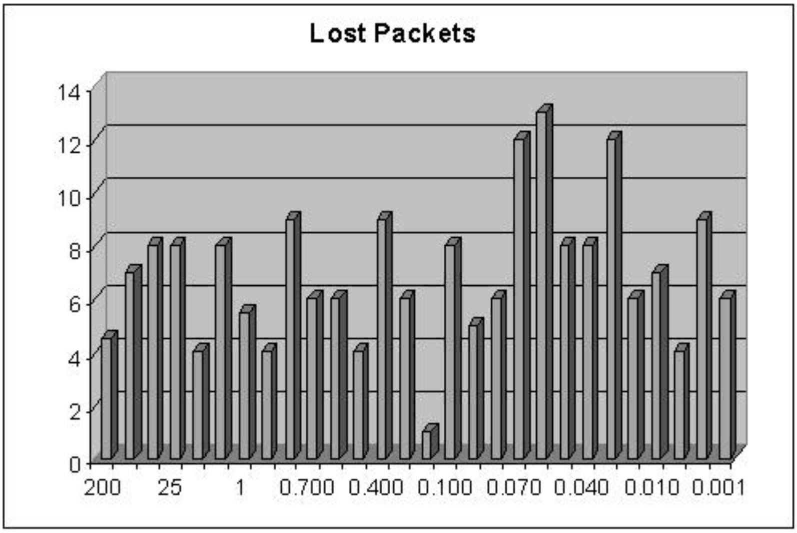


Figure 7: Test 1- lost packets.

1. The system is stressed by transmitting a fixed number of packets (1000) and varying the delay between the packets. This test aims to measure the number of packets correctly received and processed.
2. The system is stressed by transmitting a variable number of packets with a fixed delay (500ms). This test also aims to measure the number of errors in the receiving and processing phases.

The results for the first test are reported in Figure 7. The graphic lists the number of lost packets (on the ordinate) for a delay varying from 200 ms to 0.001 ms. It can be observed that the number of lost packets is almost independent of the delay; moreover, the percentage of error is 1.3% in the worst case.

Results from the second test are reported in Figure 8. The graphic lists the number of lost packets (on the ordinate) for a number of transmitted packets varying from 150 to 6000 ms. It can be observed that the number of lost packets is negligible for a number of transmitted packets less than 1500, while the error is about 10% for 6000 packets transmitted.

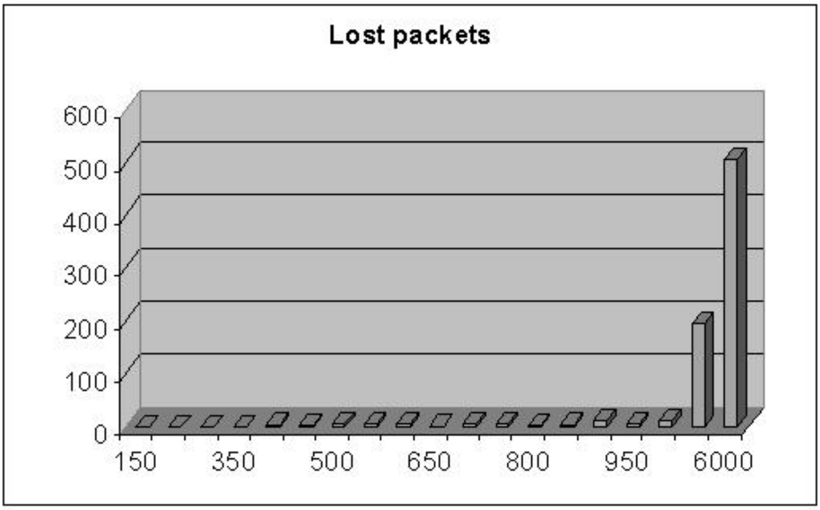


Figure 8:Test 2- lost packets.

Conclusions

Mobile devices, and in particular PDAs, are quickly changing the way information can be retrieved and visualized. Computational power and display size of a PDA are still two factors that limit the application of these new technologies when large data repositories have to be managed. On the other hand, Network Intrusion Detection Systems add a new level of visibility into the nature and characteristics of the network traffic, identifying threats from unauthorized users, back-doors, and hackers.

This paper proposes architecture able to manage and visualize the large amount of data produced by an intrusion detection program. Every time an attack or an intrusion is detected an effective visual event is sent to the user's PDA. An ad-hoc tool allows the user to organize the building to be monitored by a set of different levels of detail. Machines are placed at the leaf level of the hierarchy and for each machine it is possible to specify the user, the IP address, the operating system, etc. This hierarchical spatial data organization allows the user to efficiently and intuitively control large buildings obtaining a global view of the whole system, as well as detailed information concerning any incoming attack.

Acknowledgements

We want to thank Ing. Luca Lamorte for his support in implementing IMoViS.

Notes:

1 Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, *Readings in Information Visualization Using Vision to Think* (San Francisco, CA: Morgan Kaufmann Publishers Inc., 1999).

2 Roy Want and Gaetano Borriello, “Survey on Information Appliances,” *IEEE Computer Graphics & Applications* 20, 3 (2000): 24-31.

3 M Fairuz Shiratuddin, J. L. Perdomo, and W. Thabet, “3D Visualization Using the Pocket PC,” in *Proceedings of ECPPM 2002* (Portorož, Slovenia, 9-11 September 2002).

4 Simon Stegmaier, Marcelo Magallón, and Thomas Ertl, “A Generic Solution for Hardware-Accelerated Remote Visualization,” in *Procceedings of EG/IEEE TCVG Symposium on VisualizationVisSym’02* (Barcelona, Spain, 27-29 May, ACM, New York, 2002), 87-94.

5 Fabrizion Lamberti, Claudio Zunino, Andrea Sanna, Antonio Fiume, and Marco Maniezzo, “An Accelerated Remote Graphics Architecture for PDAs,” in *Proceedings of Web3D 2003 Symposium* (Saint Malo, France, 9-12 March, 2003).

6 Reinhard Oppermann and Marcus Specht, “Adaptive Support for a Mobile Museum Guide,” in *Proceedings of Workshop on Interactive Applications of Mobile Computing (IMC'98)* (Rostock, Germany, 24-25, November, 1998), available at http://www.rostock.igd.fhg.de/veranstaltungen/workshops/imc98/Proceedings/imc98-SessionMA3-2.pdf.

7 Vassilios Vlahakis, Nikos Ioannidis, John Karigiannis, Manolis Tsotros, Michael Gounaris, Didier Stricker, Tim Gleue, P. Daehne, and Luis Almeida, “Archeoguide: An Augmented Reality Guide for Archeological Sites,” *IEEE Computer Graphics & Applications* 22, 5 (September 2002): 52-60.

8 Andrew Johnson, Jason Leigh, Bryan Carter, Jim Sosnoski, and Steve Jones, “Virtual Harlem,” *IEEE Computer Graphics & Applications* 22, 5 (September 2002): 61-67.

9 Bernhard Preim, Andreas Fänger, Marcel Goetze, and Rainer Michel, “Guiding Travelers by a Mobile Information System for Public Transportation,” in *Proceedings of the Workshop on Adaptive Design of Interactive Multimedia Presentations for Mobile Users* (Sitges, Spain, 7 March 1999).

10 Robert S. H. Istepanian, “Modeling of a GSM-Based Mobile Telemedical System,” in *Proceedings of the 20th Conference of the IEEE Engineering in Medical and Biology* (Hong Kong, 29 October - 1 November, 1998), 926-930.

11 Robert S. H. Istepanian and Arthur A. Petrosian, “Optimal Zonal Wavelet-Based ECG Data Compression for a Mobile Telecardiology System,” *IEEE Transaction on Information Technology in Biomedicine* 4, 3 (September 2000): 200-211.

12 Bryan Woodward, Robert S. H. Istepanian, and C. I. Richards, “Design of a Telemedicine System Using a Mobile Telephone,” *IEEE Transaction on Information Technology in Biomedicine* 5, 1 (March 2001): 13-15.

13 Fabrizio Lamberti, Bartolomeo Montrucchio, Andrea Sanna, and Claudio Zunino, “A Web-based Architecture Enabling Multichannel Telemedicine Applications,” in *Proceedings of Systemics, Cybernetics and Informatics'02*, Volume **XIII** (Orlando, Florida, 14-18 July, 2002), 257-262. Also published in the online *Journal of Systemics, Cybernetics and Informatics*, Volume 1, No. 1.

[14] Andrew E. Johnson, Thomas G. Moher, Yong-Joo Cho, Ya-Ju Lin., Dave Haas, and Janet Kim, "Augmenting Elementary School Education with VR," *IEEE Computer Graphics & Applications* 22, 2 (March/April 2002): 6-9.

[15] Stephen Northcutt, Donald McLachlan, and Judy Novak, *Network Intrusion Detection - An Analyst's Handbook* (New Riders, 2nd Edition, 2001).

[16] Van Jacobson, Craig Leres, and Steven McCanne (The Tcpdump Group), *pcap- Packet Capture* (Berkeley, CA: The Lawrence Berkeley Laboratory, University of California Library, 1988-1994-2000).

[17] Fulvio Risso and Loris Degioanni, "An Architecture for High Performance Network Analysis," in *Proceedings of the 6th IEEE Symposium on Computers and Communications* (Hammamet, Tunisia, July 2001), 686-693.

[18] Steven McCanne and Van Jacobson, "The BSD Packet Filter: A New Architecture for User-Level Packet Capture," in *Proceedings of the Winter 1993 USENIX Conference* (USENIX Association, San Diego, CA, January 1993), 259- 270.

[19] B. L. Kwang, G. Roger, and J. Watt, "Zoomable User Interfaces for Personal Digital Assistants," (To be published in *the Journal of IEEE Transactions on Professional Communication*, 2003).

ANDREA SANNA graduated in Electronic Engineering in 1993, and received a Ph.D. degree in Computer Engineering in 1997, both from Politecnico di Torino, Italy. Currently he is Assistant Professor at the 2nd Engineering Faculty. He has authored and co-authored several papers in the areas of computer graphics, virtual reality, parallel and distributed computing, scientific visualization and computational geometry. Andrea Sanna is currently involved in several national and international projects concerning grids, peer-to-peer networks, and distributed technologies. *Address for correspondence:* Dipartimento di Automatica e Informatica, Politecnico di Torino, corso Duca degli Abruzzi 24, I-10129 Torino (Italy) Tel.: +39-11-564.7035 FAX: +39-11-564.7099 *E-mail:* andrea.sanna@polito.it

CLAUDIO FORNARO graduated in Electronic Engineering in 1994 and received a Ph.D. degree in Computer Engineering in 1998, both from Politecnico di Torino, Torino, Italy.Currently he is lecturer at the Politecnico and is a member of the PoliTo's Computer and Network Security Group. His research interests include Public Key Infrastructures, Intrusion Detection, and operating system and network security.