Autonomous Hybrid Honeypot as the Future of Distributed Computer Systems Security

Peter Fanfara, Marek Dufala, Ján Radušovský

Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice
Letná 9, 04001 Košice, Slovak Republic
peter.fanfara@tuke.sk, marek.dufala@tuke.sk, jan.radusovsky@tuke.sk

Abstract: Computer security presents one of the fastest-evolving segments in the Information Technologies (IT) area. The traditional system security approach is slightly focused on defence but more attention has been drawn to aggressive forms of defence against potential attackers and intruders. The advanced decoy based technology called Honeypot is a similar form of protection against intrusion. The paper is focused mainly on the proposal of the autonomous hybrid Honeypot and its features in cooperation with the Intrusion Detection System (IDS). The weakness of the detection mechanism is a major IDS shortcoming that can be minimized by using the hybrid Honeypot technology. The proposed architecture can be used as a solution for a rapid increase a security with the autonomous behaviour model in a distributed computer system.

Keywords: Honeypot; Hybrid Honeypot; Intrusion; Intrusion Detection System; Types of Honeypots

1 Introduction

People are able to find information and send messages quickly and easily due to the rapid spread of Internet and Web technologies. However, if we do not put a sufficiently high priority on basic system security at the same time, hackers can take over computers using malicious code through the existing system vulnerabilities and program weaknesses. A major damage to most of companies and to personal property will be caused by the attackers' invasion, destruction, theft and falsification of information. Nowadays, due to these potential threats, there is a growing interest in improving information security as well as intrusion detection.

The beginnings of intrusion detection have brought some complications. There still exists a gap between the theoretical and practical level of intrusion detection. Well-established defence of a network/system is based on using a firewall and an intrusion detection system (IDS). Once the attackers are aware that the firewall has allowed an exception for the external security service, they are able to use this service to gain access to the internal servers through the firewall. Subsequently, this can result in another attack. The IDS cannot provide additional information about the detection of enemy attacks and cannot reduce losses caused by those attacks [1].

A conventional approach to the security is considerably focused on defence, but the interest is increasingly devoted to more aggressive defence forms against the potential attackers and intruders. The protection against intrusions based on the bait by using a Honeypot is an example of this form [2].

Honeypot is an advanced decoy-based technology that simulates weak points of system security and unsecured system services. The potential attackers focus on system vulnerabilities and very often attack the system weakest points, which are simulated by Honeypot. This feature represents the nature of system security. Some Honeypot solutions like Honeyd or Honeynets are already used to increase the system security [3].

The proposed client-server architecture uses a specific hybrid Honeypot that mainly consists of existing tools such as Dionaea, Sebek and Snort for rapidly increasing security in the distributed computer systems. The proposed Honeypot has an autonomous feature that enables its use in a random deployment environment. This Honeypot will auto-configure itself on the basis of system parameters obtained via a passive fingerprint method.

The following chapters describe system security using IDS with the detection mechanism based on the advanced Honeypot technology.

2 Intrusion Detection System

The IDS can be defined as a tool or software application that monitors the activities of the computer system and/or network due to the potential occurrence of malicious activities or breaches of security policy. The IDS produces reports for the control station. It is primarily focused on identifying and recording information about any events as well as reporting similar attempts [4, 5].

2.1 Classification of Intrusion Detection System

In view of the various environment applications, the IDS can be classified into two general types [1]:

- *Host-based* – this consists of an agent located on host computer that is used for the continuous monitoring of information from the system audit data or network activities logs. This IDS sensor type typically includes a software agent. If there are unusual circumstances, the system automatically generates and sends a warning.

- *Network-based* – this is an independent platform for intrusions identification using direct capturing of transmitted network packets and monitoring several computers. Detection sensors are placed in network bottlenecks for capturing all network traffic and analyzing individual packet contents looking for dangerous operations.

On the basis of the detection method, the IDS can be divided into three types below [1]:

- *Anomaly detection* – refers to the pattern found in the data set that is inconsistent with normal behaviour. The anomaly detection provides basic performance for normal network traffic. An alarm sounds only if the current network traffic is above or below standard parameters.
- *Misuse detection* – collects previous hacker attack characteristics and patterns, which are then saved to knowledge attack database. Consequently, it can identify attacks with the same patterns and characteristics as those of previously stored attacks. The IDS cannot trigger an alarm if the hacker uses a new attack method that has not been previously reported or detected.
- *Hybrid mode detection* – represents attack detection using a combination of previous two types, resulting in a reduction of false alarms.

2.2 IDS Structure and Architecture

The IDS consists of several elements illustrated in Figure 1 where the main element is a sensor, the mechanism for analysis, responsible for intrusion detection. This sensor contains a mechanism that makes decisions regarding a breach. The sensor receives data from three main sources of information: the IDS knowledge database, system logs and audit trails. System logs may include for example file system configuration and user permissions. This information forms the basis for further decision on intrusion detection.

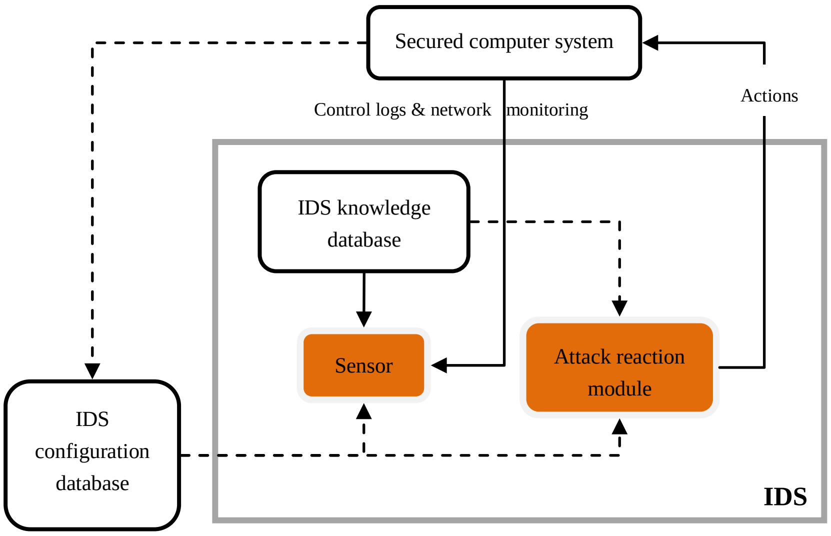

Figure 1
Intrusion detection system structure [6]

The sensor in the IDS elements illustrated in Figure 2 is integrated together with the component that is responsible for data collection, the events generator. The data collecting method is set by the policy of the events generator, which defines the filtering method for events information notifications. The events generator (operating system, network & application) in accordance with security policies produces sets of events (system logs, control records or network packets). These occurrences may be stored together with information policy either in a protected system or outside it. In some cases they are not stored, e.g. when events streams are directly transmitted to the analyzer, especially network packets [7, 8].

The role of the sensor is to filter information and to discard any irrelevant data obtained from the event file related to the protected system and to detect suspicious activity. For this purpose, the sensor uses the detection policy database, which is composed of the following parts: pattern attack, normal behaviour, profiles and necessary parameters. The database contains the IDS configuration parameters and communicating method with the reaction module. The sensor has a custom database that also includes a dynamic history of potential intrusions [7].

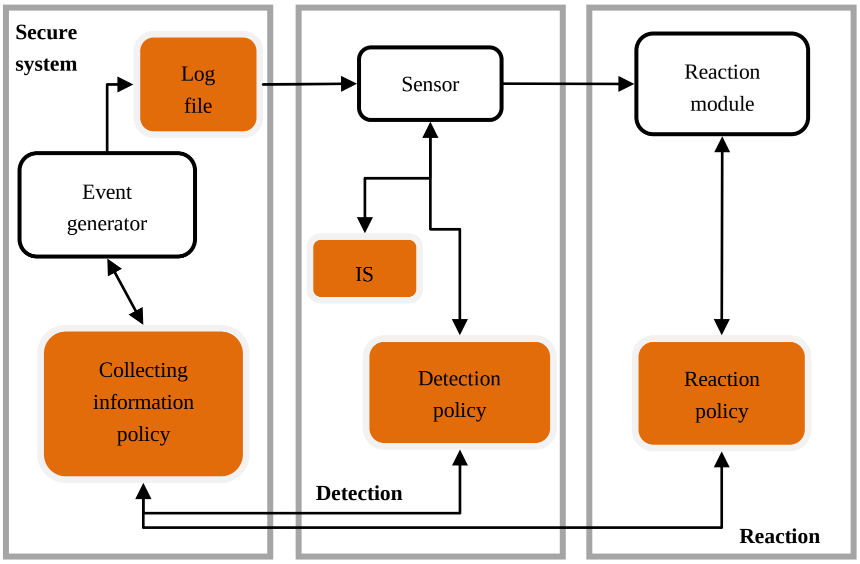

Figure 2
Intrusion detection system elements [1]

2.3 Intrusion Detection Tools

Nowadays, many IDS exist (i.e. Snort [9], SAX2, etc.), and all are specific to the system of deployment. A Snort is the most commonly used tool, one that has excellent additional conditions for usage in order to enhance the distributed system security in combination with Honeypots.

Snort represents an open-source IDS that can detect and warn of attacks (e.g. against the Honeypot). It can also capture packets and network load given by packets included in the attack. The collected information may be critical for analyzing the attacker's activities. Snort uses a modular architecture and rules based language. It combines the abnormal behaviour, detection signature and different protocol detection methods [10].

The methodology of lying and cheating by providing the emulation of some system services was domesticated in order to be able successfully monitor hackers' activities in distributed computer systems. At first sight this system appears to be legitimate. It is possible to record and monitor all hackers' activities due to the penetration and clarification of the various attackers' tactics. This idea was developed by using an advanced security tool called Honeypot.

3 Honeypot

Honeypot is a closely monitored network decoy available in different shapes and sizes, serving various purposes. It can be placed in a computer network with the firewall, in front of it and/or behind it. These points of deployment are the most frequent sites for attackers obtaining access to the system. These sites provide the best solution for acquiring the maximum amount of information about the attackers' activities. The main aim of Honeypot is to collect information by compromising the system data in the way that any system infiltration would be unfeasible to do in the future.

The main benefit of Honeypot is detection. It can address IDS shortcomings, by minimizing the amount of false positive and false negative alerts generated. There are several situations in which IDS cannot generate a warning of attack: if the attack is too short or if the appropriate security rule refers too many false alarms or detects excessive network traffic and thus drops packets. One solution is to use Honeypot, since it has no way to affect system functions. Honeypot implementation uses an unused IP address, which means that all incoming communication is almost certainly unauthorized, i.e. there are no false positive or false negative alarm warnings or large data files to be analyzed [6].

The data obtained from the Honeypots can be used to create better protection and countermeasures or system reconfiguration against future threats.

3.1 Types of Honeypots

Honeypots can be classified in different ways. Classification according to purpose and level of interaction is the most frequent one.

3.1.1 Purpose Honeypots

This basic classification divides Honeypots based on the area of deployment.

- Research Honeypot – this type is used merely for research. The main objective is to obtain as much information as possible about an intruder in a way that allows full infiltration and penetration of security system. It is used to obtain information and detect new methods and types of tools used to attack other systems as well as to analyze the hacker's traces, their identity or modus operandi. Another option in research is that the Honeypot can be used to discover potential risks and information vulnerabilities in enterprise systems [11].

  The primary function is to examine how attackers proceed and lead their attacks, which usually means understanding their motives, behaviour and organization. Research Honeypots are complex in terms of deployment, maintenance and the capturing huge amounts of data. On the other hand, they are highly useful security tools in the field of development and in enhancing forensic analysis capabilities.

In addition to the information obtained from research, the Honeypot can be used to improve prevention against attack. By improving the detection and response to attacks this Honeypot type contributes to direct security only by a small amount [12].

- Production Honeypot – it is used in organizations for protection and to help to reduce level of risk. It provides immediate enhancing of the system security [3]. Since it does not require as much functionality as the research Honeypot, its development and deployment is usually much easier. Nevertheless, it can identify various attack methods. The production Honeypot provides less information about the attacker than the research one. It is possible to determine where the attackers come from and what specific actions was performed, but it cannot determine the intruders' identities, how they are organized or which tools were used.

  The production Honeypot has minimum value as a prevention mechanism. The best way to implement this Honeypot is to use well-firewalled system, an IDS, and mechanisms for locking and fixing the system [12].

3.1.2 Level of Interaction

All Honeypots are based on the same concept: nobody should interact with Honeypot. The level of interaction can be defined as a maximum range of options available to attack allowed by Honeypot. Therefore, any transactions or interactions based on definition become illegitimate. Honeypots can also be categorized according to the level of interaction between intruders and the system. This classification helps in choosing the correct type for deploying in system [12].

- Low-interaction – does not contain any operating system (OS) for communication with the attacker. All tools are installed purely for emulation of OS and services that cannot be used to gain full access to the Honeypot. Emulation is set up to cooperate with the attacker and malicious code, resulting in radical risk reduction. Attackers can only scan the Honeypot and connect to several ports. Low-interaction Honeypots are characterized by the possibility of easy deployment and maintenance. Honeyd is an example of a low-interaction Honeypot.
- Medium-interaction – this type is more sophisticated than the previous one but still does not have installed any OS. The medium-interaction Honeypot only provides an illusion of real OS to the attacker because it contains a number of emulated services the attacker can interact with. This type is able to detect automated attacks and extract information about malware binaries. Malicious software can be automatically downloaded and analyzed. The Dionaea tool and Honeytrap are the examples of this Honeypot type.

- High-interaction – the most advanced Honeypot. On the other hand, it represents the most complex and time-consuming design with the highest rate of risk, because it implies the functional OS. It gives the attacker the ability to communicate with the real OS where nothing is simulated, emulated or restricted. This Honeypot allows for collecting the highest amount of information because it can detect and analyze all performed activities. Main focus is set to obtain valuable information about intruders by making available the entire system or even allow handling with it.

3.2 Architecture of the Hybrid Honeypot

The Hybrid Honeypot represents a combination of two Honeypots with different levels of interaction. The combination is a secure solution because it is possible to take advantage of both Honeypot types, which complement each other and thus limit their disadvantages, shown in Table 1. The ideal solution is to use a low-interaction Honeypot with a high-interaction one. The low-interaction Honeypot acts as a lightweight proxy, which relieves the high-interaction Honeypot and allows focusing on processing all IP address space network traffic [3].

Table 1
The essence of hybrid Honeypot

Low-interaction Honeypot	**High-interaction Honeypot**	**Hybrid Honeypot**
+ fast	- slow	+ fast
- no possibility to detect unknown attack	+ possibility to detect unknown attack + 0 false produced warnings	+ possibility to detect unknown attack + 0 false produced warnings
+ resists to time-bomb + handles interaction with attackers	- unable to resist time-bomb and can't handle interaction with attackers	+ resists to time-bomb + handles interaction with attackers
+ cheap	- expensive	+ relatively expensive
+ simple to set up and maintain	- complicated to set up and maintain	- complicated to set up and maintain

It is impossible for each proposed Honeypot not to use the implementation tools that have considerable importance in improving system security.

Dionaea is a modular architecture using a low-interaction Honeypot. It is able to simulate the server's main services and vulnerabilities due to attracting attacker/attack attention or the withdrawal of the malicious code [9].

Sebek is the most advanced tool for comprehensive data collection, aiming to capture as much information about the attackers' activities as possible from the Honeypot by stopping specific system calls (*syscalls*) on the kernel level [13].

3.3 Advantages and Disadvantages

All security technologies have some risk margin. If knowledge and experience represent the power of attackers, they also provide advantages for security professionals. By knowing the Honeypot risks, it is possible to use knowledge to mitigate them and reduce the disadvantages [11].

Honeypots have several unique advantages that are unique to this advanced technology [3]:

- Small data sets – Honeypots can monitor only the traffic that comes directly to them. They collect small amounts of data, but on the other hand, they may contain high value information.
- Minimal resources – Honeypots require minimum system resources for capturing harmful activities. Systems with low-end specifications will be enough to run a Honeypot.
- Discovery of new tools & tactics – Honeypots capture everything that starts interactions with them.
- Encryption or IPv6 – Honeypots can also operate in encrypted or IPv6 environments/systems.
- Simplicity – Honeypots are very easy and flexible to operate, so they do not need complicated algorithms to function properly.

The decoy-based technology, like other security solutions, also has its own disadvantages, which are described bellow [3]:

- Risk of takeover – if an attacker takes control of the Honeypot, he can exploit it to attack other systems inside or outside the system of deployment.
- Limited vision – Honeypots can only monitor the traffic that comes directly to them.
- Discovery and fingerprinting – Honeypots have some expected characteristics or behaviours. If the attacker uses some fingerprinting tool, he can identify the working Honeypot in attacked system. Even a simple error, such as a misspelled word in the emulated service, can act as a Honeypot signature.

4 IDS Architecture Using a Sophisticated Hybrid Honeypot

The main IDS weakness lies in the ability to detect new attack types. The use of different attack strategies or new tools cannot be detected by IDS. These new attacks need to be registered in the IDS configuration database and only then is it possible to detect them. The proposed IDS uses a hybrid Honeypot with an autonomous ability to reduce the risk of detection failure, and it provides extensive data collection. The Hybrid Honeypot also affords the opportunity to design safety features of distributed systems through the captured data. It also minimizes any system intrusion threats. Hybrid Honeypot combines several tools: Snort, Dionaea and Sebek. The proposed system, illustrated in Figure 3, analyzes all captured various data formats due to the rapid response to attacks. It also serves as a warning reporting system to the system administrator via web interface when interaction with Honeypot occurs.

The proposed intrusion detection system contains existing client-server detection architecture and its arrangements for using the proposed sophisticated Honeypot technology.

The architecture consists of several clients and a central server. Clients collect information about an attack and the captured malware is sent back to server. The server records and analyzes the received data, issues a warning and displays the overall information to the system administrator via web interface. The architecture is designed to achieve the effect of centralized distributed information management and to build complex distributed system of early warning for distributed computer systems.

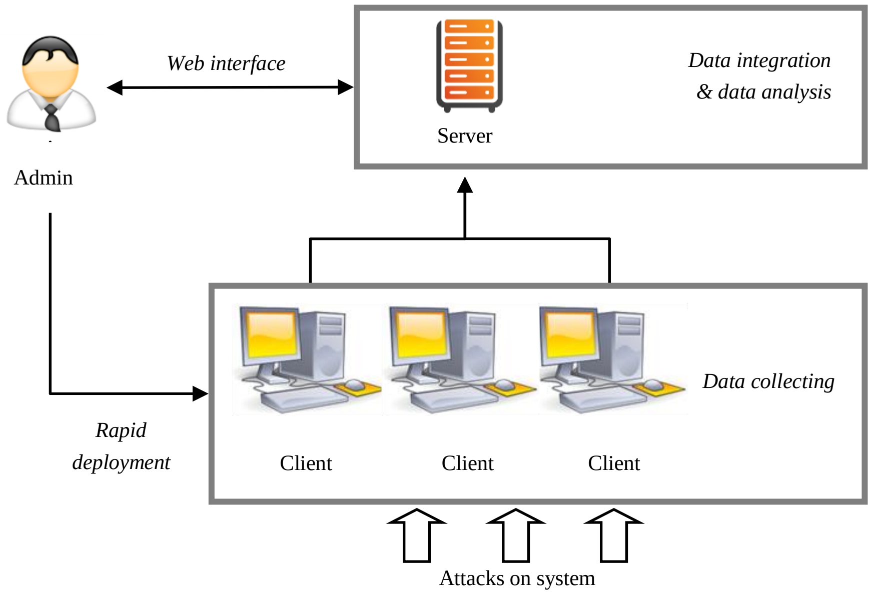

Figure 3
Architecture of proposed detection system

4.1 Client Architecture

The clients are installed in the same domain because of the data-gathering activities during the attack. Diverse system components for collecting data sets are activated depending on the different type of cyber-warfare activities. Then the data sets are sent to server for further analysis and they subsequently update the system security. The client architecture, shown in Figure 4, consists of three components:

- Snort – monitors and filters packets during intrusion detection. It identifies the patterns and characteristics of specific attacks, information and warning messages.
- Dionaea client – simulates general services and vulnerabilities that attract the attackers. It captures malware patterns and characteristics.
- Sebek client – records the attacker behaviour during interaction with Honeypot into the log files.

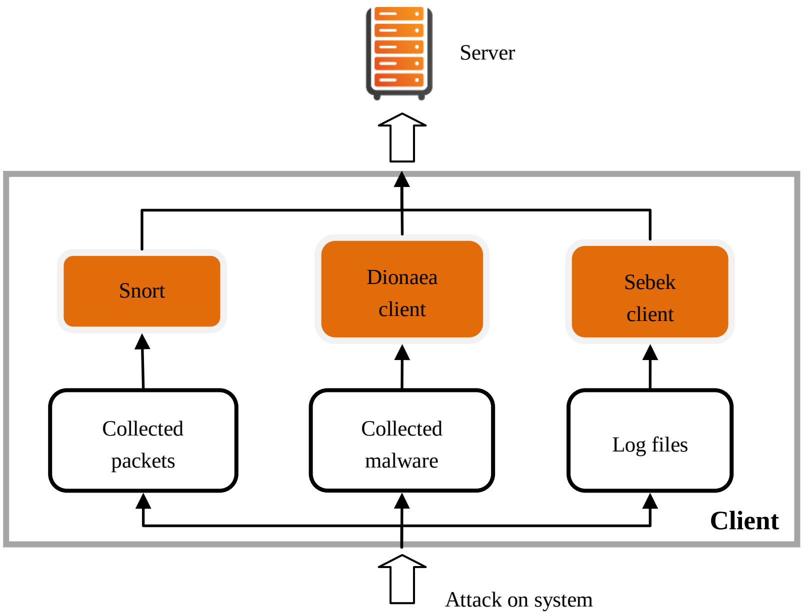

Figure 4
Client architecture

4.2 Server Architectur

At the same time, the server is connected to multiple clients owing to centralization of the collected data, and it is set to receive all outgoing messages, which are then stored in the database. The server architecture is shown in Fig. 5.

It indicates that the attacker's intention is targeted to extensive computer or scanning attacks by using individual interconnection reports. The architecture of the server consists of three main parts, the outputs that are normalized before they are stored into the database:

- Dionaea server – receives malware patterns sent by the Dionaea client component.
- Sebek server – simultaneously receives and filters multiple data sources representing the instruction or cohesion of data sent to be stored.
- Verification – modular design of open-source hybrid system for detecting an intrusion using standard communication format. The verification part can receive the data from many clients and integrate disparate data formats.

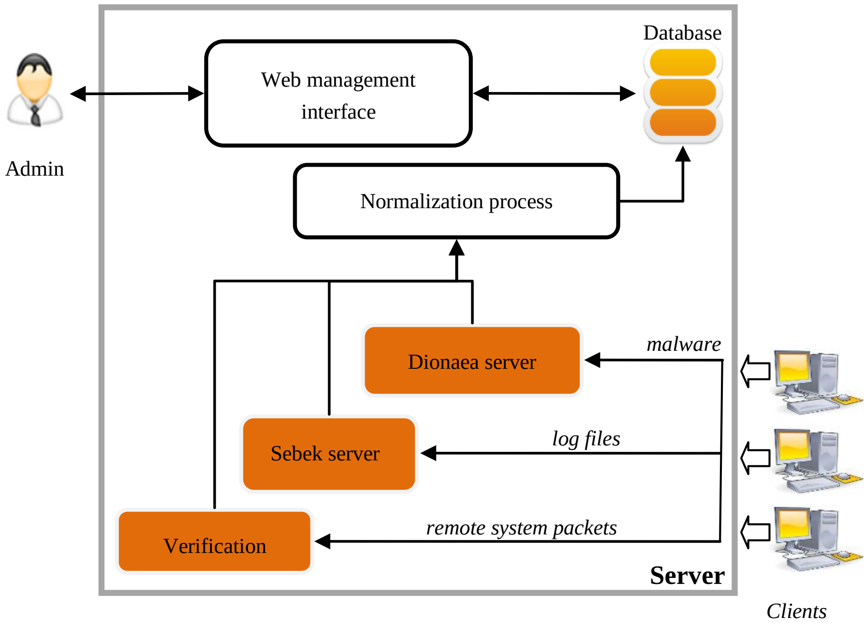

Figure 5
Server architecture

Web server interface displays all attack analyses obtained from the database. At the same time it monitors the attack patterns and occurrences of unusual circumstances. In the event of their occurrence, the specific messages are highlighted via web management interface for the correct and in time response.

4.3 Sophisticated Hybrid Honeypot

4.3.1 Desired State

The sophisticated hybrid Honeypot is a suggestion to address the current state of the security system that works on the plug-&-play principle. The optimum state will occur when the Honeypot runs a complete configuration process right after plug-in. For example, after installing the Linux OS to the distributed computer system, we will have the Linux Honeypot, or after removing any of services, the related service will be removed from the Honeypot list of emulated services. When replacing routers in the system, for example changing routers from Hewlett-Packard vendor to Cisco routers, the existing Honeypot, pretending to be a router, will immediately and autonomously auto-reconfigure and update itself. The solution is a device that simply connects to the network/system and learns the topology autonomously without any external support. Upon completion of the scanning process, the device will accurately determine the number of Honeypots with their configuration and it will be able to adapt quickly to any modifications in the system.

4.3.2 Trouble

The most critical component of the sophisticated Honeypot is the method how the Honeypot gets the information about the deployment network, for example, what systems are used and how they are used in the current environment. An example of heterogeneous distributed computer system is shown in Figure 6. The Honeypot will be able to sophistically map and promptly respond to the current system environment after obtaining the network parameters. One of the simplest possible ways is an active probing and thus determining the system and type of used services. The use of the active method of data mining also has some shortcomings in terms of increased network load; there is a risk to the running system functionality.

The sophisticated Honeypot would have to constantly scan all active environments of deployment to remedy the described lack. This solution is not the most appropriate approach.

4.3.3 Solution

The solution to the drawbacks of active system scanning is a passive approach, specifically the passive fingerprinting and mapping method. The passive fingerprinting method is not new. The idea is to obtain the system overview via mapping the current environment. The difference vis-a-vis the active method is that it has a different mapping approach. This approach is based on obtaining information through passively capturing network traffic, analyzing it and then determining the system identity based on the unique system fingerprints. The passive method uses the same method as the active one but in different ways.

Tools, such as Nmap [14], create a signature database that contains known operating systems and services. All searching tools actively broadcast packets that require a response from destination devices right after creating a signature database. Incoming responses are unique to most operating systems and services. Responses are simply compared with the data in the signature database due to a clear identification of the operating system and used services.

Passive fingerprinting uses the same approach as the signature database, unless the data are obtained passively. Instead of actively probing the system, the passive fingerprinting method intercepts network traffic and analyzes the captured packets, which are then compared with a signature database. After the analyzing process ends, the concrete operating system should be known. Passive fingerprinting is not limited to use only with the TCP protocol, which allows for the use of other protocols. The usage of the passive method represents several advantages: less likelihood of damage or shutting down of the system or service, and the ability to identify systems using a firewall. The passive method is continuous, which means changes in the network structure are captured in real time [8]. This advantage becomes a critical feature in maintaining a realistic Honeypot over a longer time period. The only disadvantage of the passive method is the correctness of functioning through the routed networks; the most effective usage of passive obtaining parameters method is in local area networks.

4.3.4 Concept

The proposed Honeypot data obtaining mechanism is based on the concept of the passive fingerprinting method. The Honeypot is deployed as an independent device that is physically connected to the computer network of a distributed computer system. The tracking and learning phase starts after connecting to a network device. In this phase, the Honeypot learns the topology and plans the deployment of other Honeypots. The duration of learning phase is variable and depends on the system topology. The proposed Honeypot can determine the number of used systems, the types of operating systems, and the running services via passive analyzing of the network traffic. It also has the feature to determine with whom and how often a concrete system or service communicates. This information is used for mapping and obtaining knowledge about the deployment network.

Once the Honeypot collects all the necessary information, it can start with the Honeypot deployment illustrated in Figure 7. The created Honeypots are designed to mirror the real system and decrease the risk of the attack. Honeypots with the ability to look and behave in the same way as the production environment can easily blend with their surroundings. Their identification and tracing by attackers is much more difficult.

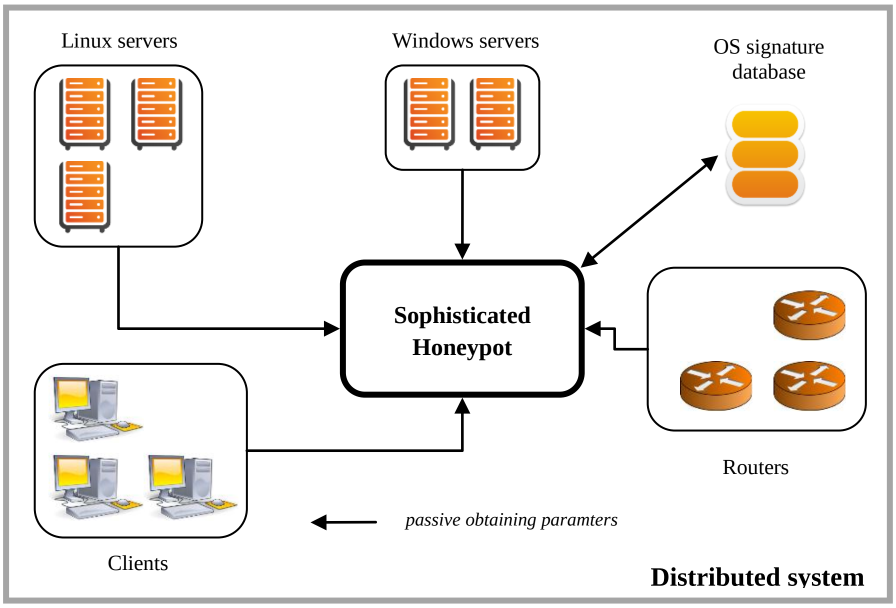

Figure 6
Passive obtaining system parameters via sophisticated Honeypot during determination process of deployment virtual Honeypots

Passive acquisition of information does not end but rather is continuous. It monitors the entire network system and increases its flexibility. Any change is identified in real time and the necessary steps (the system deployed Honeypots) are realized in the fastest possible time.

The proposed sophisticated Honeypot considerably reduces the work necessitated by configuration and administration in a constantly changing environment.

4.3.5 Deploying Honeypots in a System

The traditional solution to the issue of implementing the Honeypot in the system requires the physical placement of a new computer for each monitored IP address. The physical Honeypot deployment represents considerable time and work. An autonomous and simpler solution, for example, fire-&-forget, is not to implement a physical Honeypot but rather a virtual type, which, if in sufficient quantity, can monitor all the unused IP addresses. Virtual decoys pursue identical IP address space as the system itself. All virtual decoys are designed, located and managed by only one physical device, the proposed sophisticated Honeypot illustrated in Figure 7.

Whereas virtual Honeypots monitor unused IP addresses in computer networks, it is almost certain that any activity detected on the monitored IP addresses is most likely a malicious or unauthorized behaviour. Using information gathered through passive mapping of the environment can determine the quantity and type of Honeypot deployment.

The ability to dynamically create and deploy virtual decoys already exists. An open-source solution with a low interaction Honeypot called Honeyd [15] allows for deploying virtual environment decoys throughout the organization. It is possible to realize the design of a sophisticated autonomous Honeypot with dynamic creation and deployment of virtual decoys that minimizes the risk of detection and identification of intruders by merging the surrounding environment with a combination of options, such a Honeyd solutions capabilities and passive fingerprinting.

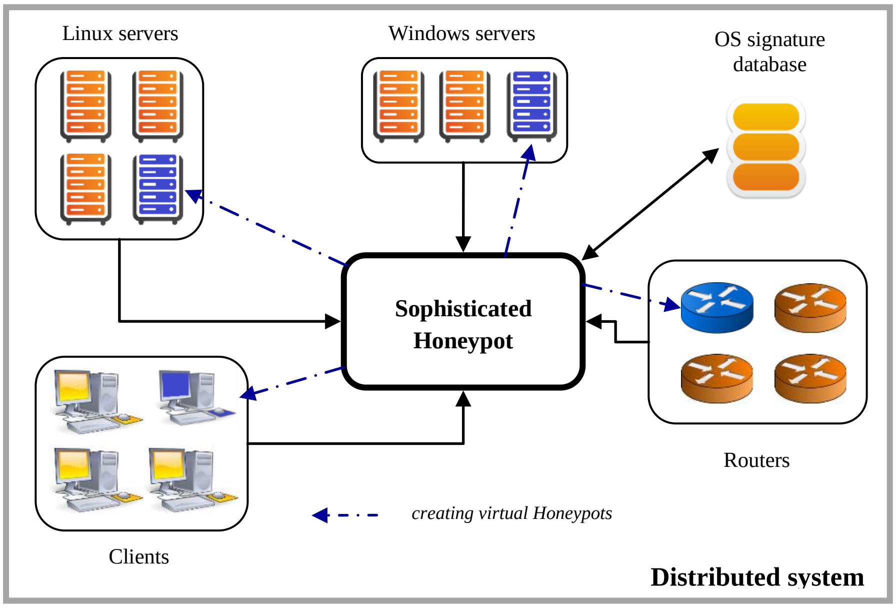

Figure 7
Deployment of virtual Honeypots based on obtained parameters

Conclusion

The security of information technologies is essential in a society that depends on information. Therefore, considerable emphasis is placed on data and information source protection in the systems development processes [16, 17]. The protection of access, availability and data integrity represents the basic safety features required for information resources. Any disruption of these properties will end in penetration into the system and would increase security risk. One way of defense is a system that detects unusual and suspicious behavior, called the IDS. IDS major risk is represented by undetected penetration problem.

An advanced technology called Honeypot has huge potential for the security community and it can also achieve several objectives of other technologies, which makes it almost a universal solution. The usage of Honeypots represents a cost-effective answer to improvements in the organization security status.

Honeypots, like any new technology, also have some shortcomings that need to be overcome and removed. The use of decoy-based technology represents a cost-effective solution to increase the security status of the organization. Therefore, they are being deployed in systems at an increasing rate, but mostly as a passive device. Many system administrators monitor the situation in the system via Honeypot, and if a production environment is attacked once, administrators analyze and implement solutions manually; the Honeypots' capabilities are not used at all, or they are used minimally. Despite the many advantages of Honeypot, it is not a panacea for all breaches of system security. Since it is used for gathering information about attacker and other threats, it is useful as an IDS detection mechanism.

The future of Honeypots and cyber security intrusion detection lies in sophisticated (autonomous hybrid) decoys. They have a radical revolutionary assumption in autonomous deployment and maintenance. They are becoming a highly-scalable solution due to their capability to study and monitor the network real time. Deployment and management becomes more cost-effective and also provides better integration into the system of deployment. Another advantage of the proposed Honeypot lies in minimizing the risk of human errors during manual configuration. The merger surrounding environment also minimizes the risk of being identified by the attacker.

Acknowledgement

This work has been supported by the Slovak Research and Development Agency under the contract No. APVV-0008-10 (20%). This work is also a result of implementing the Project: Development of the Centre of Information and Communication Technologies for Knowledge Systems (ITMS project code: 26220120030) supported by the Research & Development Operational Program funded by the ERDF (80%).

References

[1] J. McHugh, A. Christie, J. Allen: "Defending Yourself: The Role of Intrusion Detection System," IEEE Software, IEEE Computer Society, pp. 42-51, October 2000

[2] F. G. Lyon: "Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning," [online], Nmap Project, USA, ISBN 978-0979958717, January 2009. Available on: <http://nmap.org/book>

[3] S. Karthik, B. Samudrala, A. T. Yang: "Design of Network Security Projects Using Honeypots," Journal of Computing Sciences in Colleges, 2004

[4] L. Vokorokos, N. Ádám, A. Baláž: "Application of Intrusion Detection Systems in Distributed Computer Systems and Dynamic Networks," Computer Science and Technology Research Survey, Košice, 2008, pp. 19-24, ISBN 978-80-8086-100-1

[5] L. Vokorokos, N. Ádám, A. Baláž, J. Perháč: "High-performance Intrusion Detection System for Security Threats Identification in Computer Network," Computer Science and Technology Research Survey, Košice, 2009, pp. 54-61, ISBN 978-80-8086-131-5

[6] R. Baumann, C. Plattner: "White Paper: Honeypots," Swiss Federal Institute of Technology, Zurich, 2002

[7] L. Vokorokos et al: "Architecture of Intrusion Detection System Based on Partially Ordered Events", Computer Science and Technology Research Survey, Vol. 2, 2007, pp. 80-91, ISBN 9788080860714

[8] L. Vokorokos, A. Pekár, N. Ádám: "Data Preprocessing for Efficient Evaluation of Network Traffic Parameters," INES 2012: IEEE 16th International Conference on Intelligent Engineering Systems, 2012, Lisbon, Portugal, pp. 363-367, ISBN 978-1-4673-2695-7

[9] Snort [online]. Available on: <http://www.snort.org>

[10] M. Tomášek, M. Čajkovský, B. Madoš: "Intrusion Detection System Based on System Behavior", SAMI 2012: 10th IEEE Jubilee International Symposium on Applied Machine Intelligence and Informatics: proceedings: Herľany, Slovakia, 2012, pp. 271-275, ISBN 978-1-4577-0195-5

[11] L. Spitzner: "The Value of Honeypots, Part One: Definitions and Values of Honeypots," Security Focus, 2001

[12] L. Spitzner: "Honeypots: Tracking Hackers," Boston, USA: Addison-Wesley, Pearson Education, 2003, ISBN 0-321-10895-7

[13] Dionaea catches bug [online]. Available on: <http://dionaea.carnivore.it/>

[14] R. Chandran, S. Pakala: "Simulating Network with Honeyd," [online], Technical paper, Paladion Networks, December 2003. Available on: <http://www.paladion.net/papers/simulating_networks_with_honeyd.pdf>

[15] N. Provos: "Developments of the Honeyd Virtual Honeypot," [online]. Available on: <http://www.honeyd.org>

[16] Cs. Szabó, L. Samuelis: "Notes on the Software Evolution within Test Plans," Acta Electrotechnica et Informatica, Vol. 8, Issue 2, pp. 56-63, 2008, ISSN 1335-8243

[17] Cs. Szabó, L. Samuelis: "The A-shaped Model of Software Life Cycle," SAMI 2007, Slovakia, pp. 129-135, ISBN 9789637154560

[18] Sebek [online]. Available on: <http://www.honeynet.org/tools/sebek/>

[19] P. Jakubčo, E. Danková: "Distributed Emulation Using GPGPU," Electrical Engineering and Informatics 2: Proceeding of the Faculty of Electrical Engineering and Informatics of the Technical University of Košice, Slovakia 2011, pp. 284-287, ISBN 978-80-553-0611-7