Collaborative architecture for malware detection and analysis

Michele Colajanni, Daniele Gozzi, and Mirco Marchetti

Abstract The constant increase of malware threats clearly shows that the present countermeasures are not sufficient especially because most actions are put in place only when infections have already spread. In this paper, we present an innovative collaborative architecture for malware analysis that aims to early detection and timely deployment of countermeasures. The proposed system is a multi-tier architecture where the sensor nodes are geographically distributed over multiple organizations. These nodes send alerts to intermediate managers that, in their turn, communicate with one logical collector and analyzer. Relevant information, that is determined by the automatic analysis of the malware behavior in a sandbox, and countermeasures are sent to all the cooperating networks. There are many other novel features in the proposal. The architecture is extremely scalable and flexible because multiple levels of intermediate managers can be utilized depending on the complexity of the network of the participating organization. Cyphered communications among components help preventing the leakage of sensitive information and allow the pairwise authentication of the nodes involved in the information sharing. The feasibility of the proposed architecture is demonstrated through an operative prototype realized using open source software.

1 Introduction

It is pointless to repeat once again that the global network is growing steadily and fast, and that the attached hosts are becoming more and more tightly connected (for example with the shift from dial-up PSTN connections to broadband xDSL and cable-TV connections). The increasing phenomenon of botnet infections is a real threat to organizations which rely heavily on their web presence. Some evidences

Department of Computer Engineering, University of Modena and Reggio Emilia, e-mail: michele.colajanni@unimore.it, e-mail: daniele.gozzi@unimore.it, e-mail: mirco.marchetti@unimore.it

Please use the following format when citing this chapter:

Colajanni, M., Gozzi, D. and Marchetti, M., 2008, in IFIP International Federation for Information Processing, Volume 278; *Proceedings of the IFIP TC 11 23rd International Information Security Conference*; Sushil Jajodia, Pierangela Samarati, Stelvio Cimato; (Boston: Springer), pp. 79–93.

have been found that connect large botnets with organized crime, so voiding the influence of this type of worms is not any more just a matter of computer security.

The present defense mechanisms against the most virulent forms of malware and botnets are clearly inadequate. Some estimates [1] show that the Storm worm reached two million machines, thus giving its owner a computing power theoretically higher than the world's top supercomputers. Other recent data concerning worm spread can be obtained from the ShadowServer site [2]. Any domestic personal computer is heavily targeted by self-replicating malware when it is connected to an ADSL line. (We recorded through the honeypot Nepenthes [3] 10464 infection attempts over a 178 hour period, which is about one attempt per minute on average.)

The usual defense mechanisms aim to apply patches on demand well past the first attack attempt and the discovery of a new vulnerability. For a safer diffusion of the Internet-based services, we think it is important to move from independent and late defenses to coordinated, timely and possibly preventive countermeasures.

We present an innovative collaborative architecture that aims to anticipate malware detection, analysis and related countermeasures. The cooperation between heterogeneous and geographically distributed networks can be especially useful to fight autonomously spreading malware (i.e., worms) that represents the main focus of this paper. For example, most negative effects of malware and botnet spreading can be mitigated by simple packet filtering policies that must be activated as soon as possible. Unfortunately, each network implements this type of countermeasures in an individual fashion, without any knowledge of what is happening in other networks. In our proposal that allows different networks to cooperate, all information about a new malware type (threats, how is spreading, which kind of vulnerabilities it exploits, which software application or operating system can be affected, countermeasures) gathered by one sensor is propagated in a fast, reliable and trusted way with the goal of preventing the infection in other not yet touched networks.

The proposed architecture has several innovative features. Unlike the few existing collaborative systems that are mainly oriented to spam fighting, the proposed system is oriented to malware detection, analysis and communication of security threats. Its flexibility and scalability is intrinsic in the architecture design that is based on a decentralized communication scheme and a multi-tiered hierarchy of geographically distributed components. The proposed solution is general and takes advantage of the knowledge of each participant on its network part while requiring a very unobtrusive trust scheme. Cross-organization security initiatives are rarely seen even if some interesting solutions for partial information disclosure have been presented [4].

Current initiatives which require the cooperation of many users to collect malware represent an appreciable start, but most of the analysis work is still manual and there is a strict separation between anti-malware research and deployment of countermeasures. On the other hand, the proposed architecture needs a limited or null human intervention that differentiates it from analogous solutions in similar fields. Finally, it is worth to observe that we take advantage of honeypot properties to share information without raising privacy concerns between different organizations.

The remainder of this paper is structured as follows. In Section 2, we evidence the contribution of this paper with respect to the literature. In Section 3, we give

an overall description of the cooperation architecture and the details of the main components. In Section (4) we describe the implementation of a prototype version of the proposed architecture. In Section 5, we present the results of some experimental tests. Finally, in section 6 we state our conclusions and outline future research work opened by this paper.

2 Related work

A cooperative multi-tiered hierarchy of geographically distributed components for fighting malware represents an original proposal. However, other interesting cross-organization security initiatives exist. We can cite the DNS black lists (DNSBL) [5], Botnet investigation [6, 2], IDS alert correlation frameworks [7, 8], partial information disclosure [4].

DNS black lists (DNSBL) are one of the existing automated facilities for limiting the activities of suspicious hosts, but they are still highly focused on a single service (email). Our approach is general and avoids blacklists, since many home Internet connections have dynamic IP addresses and the inclusion of such addresses would be detrimental to the efficacy of the blacklist and to the user experience.

Investigation of botnets is still highly manual. The existing efforts are more oriented to help law enforcement agencies, but not so much to limit malware spread. Instead, we aim mainly to counter worm infections and botnet activities from a technical point of view. The absence of just one common countermeasure deployment approach is an intentional goal of the designed architecture that intends to avoid overreactions and prevent denials of service induced by attackers with specially crafted malware.

Current NIDS alert correlation frameworks are rarely (if at all) seen in a cross-organization deployment. We insist once again on the benefits of intrusion information sharing and we propose a cooperative architecture where each participant has to trust only two other parties (its Manager and the Collector, as evidenced in section 3) and communication is authenticated and encrypted. The pairwise trust scheme and cyphered communications among the components represent other interesting novel features of the proposal, although similar solutions have been suggested in other contexts [9, 10, 11]. The benefits of cooperation are not indirect: each participant is notified timely of security threats collected from any cooperating organization. The shared data does not need any modification for increased anonymity, since the involved hosts are an attacker and a honeypot.

Malware collection organizations, such as the *mwcollect Alliance* [12], focus on grasping the dynamics of infection spreading and detecting new malware types and variants. This operation is carried out mainly through manual analysis of binary code and extraction of call graphs. We seek to obtain a substantial reduction of the human interaction required for identifying new malware.

The system design guarantees an intrinsic flexibility and scalability that, to the best of our knowledge, cannot be found in any existing proposal as a priority architecture requirement.

3 Architecture design

The growing proliferation of Internet worms is mainly due to their non-stopping evolution of the spreading and replication mechanisms, which have traversed several stages:

1. automated infection of vulnerable network services on servers (e.g. [13]);
2. automated infection of vulnerable collateral network services on desktop computers (e.g., [14]);
3. email spreading, where the attack code is activated by unaware users (e.g., [15]);
4. email spreading, where infection is based on a mail user agent vulnerability (e.g., [16]);
5. infection of vulnerable Web applications, often carried out through XSS techniques; the search for vulnerable targets is made through web search engines (e.g., [17]);
6. infection of the client side execution environment (JavaScript or Flash) in rich Web applications, such as the lOrdOfthenOOse and EricAndrew worms [18].

The main types of malware that the present version of our architecture is targeting are those of types 1 and 2. The scheme in Figure 1 describes the main components of the proposed multi-tier system: sensors, managers, collector. In this section we describe each component and show how this solution can be scaled to become a massive geographically distributed network of cooperating honeypots.

3.1 Sensors

We define a cooperating network as a honeypot sensor installation in a remote location. The sensor is able to collect infection attempts from its location and collect the payloads of the offending worms. Ideally, each machine connected to the Internet has the same chance of being targeted by a worm, however the presence of firewalls in some organizations internal networks has the effect of slowing the infection because some protocols are blocked for inbound connections. The capillary distribution of honeypot sensors grants a thorough monitoring of malware spread, but the locally stored malware payloads have to be transferred to a collection point where they are further analyzed through some behavior and safe supervision. Malware collection should abstract from the topology of the underlying networks, and a single point of connection between the cooperating networks is aimed at preventing the disclosure of the the internal network structures.

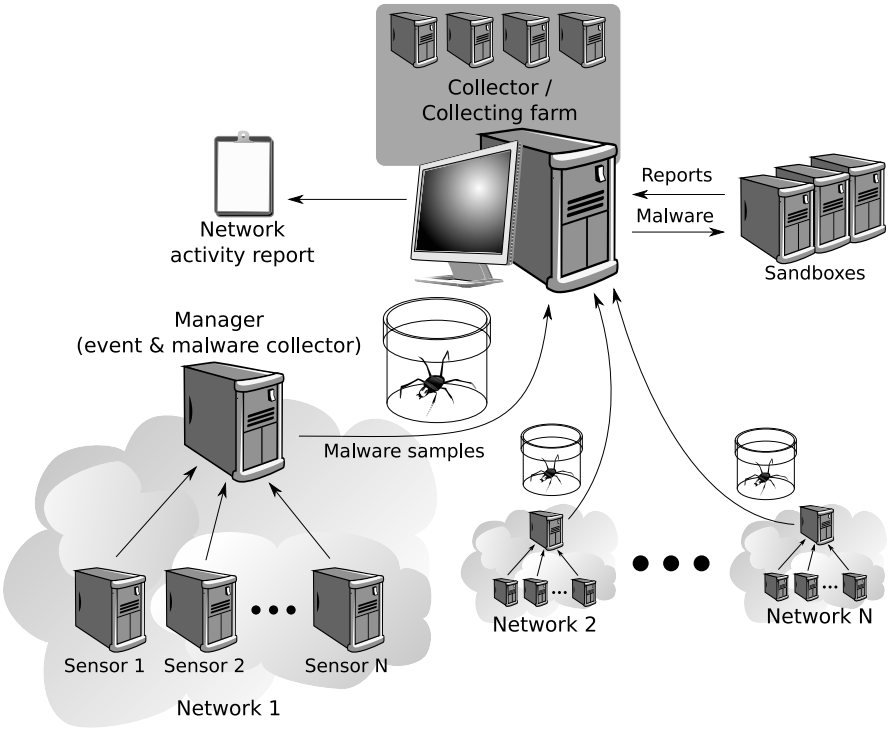

Fig. 1 Cooperative architecture for malware detection and analysis

The edge level of the proposed architecture is composed of *sensors*, which could be every type of IDMEF [19] event generator. However, low interaction honeypots, such as Nepenthes [3], are the best suited sensor type for our purpose, as they are able to collect copies of the malware payload while guaranteeing a continuous operation.

Upon collection, the malware payload samples are sorted on the basis of their MD5 hash. This solution prevents the collection of duplicate binaries, although the chance of hash collisions is not null. Novel malware is marked as different from known malware because its MD5 hash is unknown. Polymorphic malware spans over many hashes while having actually the same behavior and the operation needed for correlating the different hashes of the same malware as the mwcollect Alliance does is currently manual. We will describe in section 3.3 how to address this issue.

3.2 Managers

Managers are the architecture nodes which collect alerts and payloads from the set of sensors. A manager installation is composed by a Manager process (e.g., Prelude)

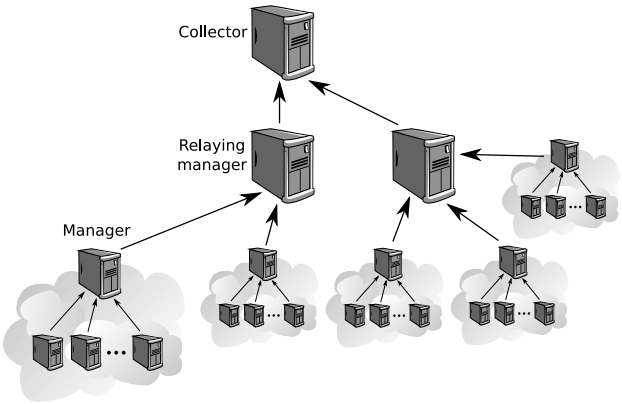

Fig. 2 Hierarchical organization of managers

and a polling agent which retrieves unknown malware payloads from the controlled sensors. The ideal location for the malware payload would be a *base64* encoded IDMEF *AdditionalData* element [19], however the current honeypots tested as sensors do not provide a payload transfer facility inside IDMEF messages. To solve this problem, we utilize a script which periodically lists captured payloads on sensors and retrieves those with an unknown MD5 hash.

The managers can be configured in a relaying way. As Figure 2 shows, the alert and payload collection nodes are connected as a hierarchy. This topology maximizes the collection capability while keeping low the number of transferred payloads, because the payloads are transferred to a higher level manager only if they are not already present there. This solution guarantees the transmission of each new malware variant to the collector .

The flexibility of the architecture is guaranteed by the possibility of having any number of manager levels. In such a way, small organizations can connect its sensor(s) directly to a remote manager; complex organizations can have multiple levels of managers that are controlled locally and connected to one or multiple remote managers.

3.3 Collector

The *Collector* is the top element of the hierarchical architecture. Each cooperating network contributes to the collection of malware hosted by the Collector. For each incoming malware sample, the collector runs an automated thorough analysis by means of local and external tools. The result of the analysis is stored and utilized to classify the malware. In our test case, the collector forwards the malware to some remote sandbox services and sends an email to the administrators of all cooperating networks with the analysis results that evidence also the ports and the protocols

involved in the malware remote controlling. It is important to observe that, to avoid system bottleneck and single points of failure, the collector is one logical component that actually runs on a cluster of machines.

Cyphering and polymorphism mechanisms may be applied to the worm code both if the worm is spread in a binary form or as a script: binaries can be altered by polymorphisms, while scripts may be obfuscated. These attacker strategies makes extremely difficult the worm classification as a single phenomenon, and this problem affects mostly the reverse engineering of the worm code. Worm detection is usually done through signature checking which becomes harder because of an increased number of signatures in the case of polymorphic malware. However, the *perfect* polymorphic engine (one that can change all the malware code at each attack) has yet to come. After a first classification through the MD5 hash, the mwcollect Alliance currently employs custom hash functions to classify the variants of the same worm with polymorphic transformations. This approach is indeed effective, but a special hash function has to be designed for each different polymorphic worm and the internals of this function have to remain undisclosed, otherwise the worm author could easily prepare an immune variant.

In order to solve the problems of payload cyphering and polymorphism, our malware classification is done in two steps. During the first step, the malware is analyzed by many different antivirus engines. If the payload cannot be identified just by a signature detection, the analysis proceeds to the second step. Here, the malware is executed in a protected environment and its effects are monitored (this technique is known as *sandboxing*).

While the behavioral analysis is not as precise as signature analysis since two different worms may have similar behaviors, it is the only way to collect any data for identifying the structure of botnets. This sandbox-backed analysis, although not exact, is the way of classifying the polymorphous variants of the same malware which fits best the purpose of fighting malware spread. This method has the benefit of being completely automatic, while not as exact as signature detection. All connections to honeypots are by default intrusion attempts, so assuming that the analyzed binary is harmful, it is perfectly legitimate. The knowledge of the real activities carried out by the malware if preferable to an exact classification.

3.4 Activity report

When using multiple remote sandboxes for analysis, the corresponding results will be dis-homogeneous, often unstructured and not directly comparable. In order to allow the cooperating networks to take advantage of the report information, all the results are adapted by the collector. In particular, the endpoints of the observed network connections which are related to the malware activity are highlighted. After the adaptation phase, the reports are sent to all the cooperating networks, even to those not yet reached by the malware. In this way, all the components receive the information needed for deploying defensive countermeasures, such as blocking

certain network connections, closing ports or patching some software applications. The exchange of information between the cooperating networks is the most effective measure against the spread of malware. By knowing timely how a worm is being transmitted, the administrators can deploy adequate countermeasures or at least plan some attack mitigation actions. Furthermore, the knowledge about the infection vector may be shared very early with the vendor of the targeted software, which can start correcting the problem when very few machines have been attacked. The increased bandwidth and availability of Internet connections facilitate the patch deliver in a short time.

One of the countermeasures that can be adopted against malware is the interruption of connections towards the malware distribution servers and the *Command & Control* (C&C) servers, if the payload is downloaded from a remote location or the malware has a control infrastructure. Usually these countermeasures are activated after the malware has begun spreading, not on a preventive basis. Each network implements this type of countermeasures in an individual way. On the other hand, if different networks cooperated, the information describing the way a new malware is spreading could be used to prevent the infection in networks not yet touched. It may even be feasible to deploy preventive measures before the working hours. For example, if the C&C servers are identified, it is possible to put into place new firewalling rules which prevent any infected host behind the firewall to connect and download further instructions from the malware controller; it also becomes possible to write blacklists of known C&C servers (although some recent worms such as Storm use decentralized communication systems).

3.5 Communication security

Is is essential to prevent a single node from polluting the set of collected data when we are aggregating information from many sources. The data may be misleading due to a malfunctioning or because a malicious user joined the network of cooperating sensors. In both instances, we need a way to trace back every alert to its origin. We use a public key cryptographic scheme to address this issue. The collector and the managers are provided with a public and private key pair, which are used for authenticating all the information exchanged among the architecture components. Each component knows in advance the keys of its communicating components. This choice guarantees the traceability of communications, together with the certainty that only registered managers can communicate with higher level managers and with the collector. Public key cryptography also provides confidentiality to the communication. The proposed communication scheme takes advantage of cooperation without the need of exchanging data directly between peers, since all the communications occur vertically. The only necessary trust relationship is pairwise between a sensor and a manager, or different lines of managers or between a top manager and the collector.

3.6 Malware collection

A capillary distributed network of sensors allows us to collect a set of data statistically relevant, that can be used for practical and research purposes. Most security products vendors have a similar infrastructure, but the proposed decentralized network has the advantage of being **vendor-agnostic** and possibly larger.

4 Prototype implementation

The feasibility of the proposed architecture is demonstrated through the realization of a prototype based on open source software and custom integration scripts. The prototype has been validated experimentally in controlled conditions as a whole and in its individual components through known malware. Furthermore, the prototype has been deployed in live operation, and it has been able to collect previously unknown malware. In this section we describe the technical details of the most important and novel components.

4.1 Malware collection

Malware collection is carried out by Nepenthes [3] instances. This software is a modular daemon which mainly has the following functions:

- socket binding and listening for incoming connections
- identification of targeted vulnerability
- analysis of the exploit code for the extraction of information necessary for downloading the worm payload
- payload retrieval
- logging and issuing alerts, potentially to a remote server through the IDMEF protocol

Each of these functions is implemented in a separate module, which in the Nepenthes source code is prefixed by a descriptive prefix, such as *dnsresolve-*, *download-*, *module-*, *log-*, *shellcode-*, *shellemu-*, *sqlhandler-*, *submit-*, *vuln-*. For example, modules whose name starts with ”vuln” contain the vulnerability simulation logic which is needed to reply correctly to attacks so as to retrieve the payload location.

Malware distribution can occur in many ways, and the honeypot software has to support as may method as possible. Sometimes the shellcode opens a remote connection with a TCP or UDP stream from which it transfers subsequent commands; otherwise a TFTP, FTP or HTTP download is used to transfer the worm payload. Nepenthes computes a SHA512 or MD5 hash of the malware and it stores a copy of the harmful binary. Such binaries can be moved to remote servers with different

methods or be submitted[1] to organizations that search for new malware, like the mwcollect Alliance [12], or Norman [20]. Nepenthes also provides a *libprelude* output module which can be used for issuing alerts to a Prelude Manager. In the prototype which we prepared, sensors use the Prelude output module to inform their respective manager of new infection attempts, while managers periodically call a script which looks in the archived payload directory of each controlled sensor. Since it would be preferable to collect malware samples as soon as possible, we plan to integrate alert issuing and payload uploading in the future.

4.2 Communication infrastructure

The critical component of the proposed distributed architecture is the communication infrastructure. Its requirements are:

- forwarding of alerts and malware binaries from sensors to managers and between managers
- transfer of unknown malware samples to the collector
- authentication of all exchanged messages
- confidentiality

In addition to these functional requirements, it is essential for the format of the exchanged messages to be a recognized standard and allow the maximum interoperability between heterogeneous threat detection systems. These considerations brought ourselves to choosing Prelude [21] as the alert management framework.

4.2.1 Prelude: a hybrid IDS

Prelude is an Open Source software which allows the deployment of a hybrid intrusion detection system - an aggregate of sensors employing different technologies and approaches to detect attacks. A typical use case is the integration of Host and Network IDSs in large networks. The format of the exchanged messages is IDMEF [19]. Prelude offers a library (*libprelude*) that security-related software can use to issue alerts and communicate with Prelude managers. Communications are encrypted using public key cryptography and relaying of messages is supported by managers, so it becomes possible to build a hierarchical network of malware-collecting nodes.

4.2.2 Transferring captured malware

A key requirement or the proposed architecture which is not natively supported by the hybrid IDS Prelude is the submission of the binaries downloaded by Nepenthes

[1] submission is done though the GOTEK protocol or with custom solutions which are different for each collection service

to a collection node. The sole propagation of IDMEF alerts is not sufficient for this. The Nepenthes developers are currently working on the integration of their work with the malware submission services of some sandboxes (CWSandbox and Norman) and online antivirus engines (VirusTotal), and they have developed the GOTEK malware distribution protocol which is actively employed by the mwcollect Alliance.

Our implementation is based on a script which is executed periodically on the collector and manager nodes. The script spawns a remote shell to each machine that is managed on the immediately lower level of the architecture, lists the collected binaries and retrieves the unknown ones. For example, the collector examines the caches of the highest level managers, which in turn collect the malware binaries from their respective subordinate managers. The lowest level managers collect binaries from their pool of honeypots.

4.2.3 Malware analysis

One of the advantages of the proposed architecture is the independency from a single malware analysis tool. It is possible to employ locally installed antivirus engines or sandboxes as well as remote public malware analysis services.

Our prototype is able to submit malware to three different remote services:

- Virustotal [22], via SMTP submission
- Norman Sandbox [23], using a custom HTTP POST request
- CW Sandbox [24], also using HTTP POST

The process of submitting the malware samples is handled by a custom modular software which is easily extendable to support other analysis services. By combining traditional signature detection and behavioral analysis (both from different vendors) we can identify clearly the actions performed by malware. Exact classification of the malware is only marginally useful since we know that the analyzed binary comes indeed from a worm, having collected it with a honeypot.

4.2.4 Report generation and sending

The results of different analysis services have heterogeneous formats and are typically semi-structured texts. Before sending the results to the administrators of the cooperating networks, the reports are tagged semantically so that an automated response may be prepared from each network accordingly to the local policies. The most important data are the location of the malware payload and the retrieval method, the address of command & control servers, the IP addresses of known infected hosts and the protocol being exploited for the infection to occur.

We remark that those reports can be easily used to generate and deploy automatic countermeasures without human intervention, thus greatly reducing the time required to react a network attack

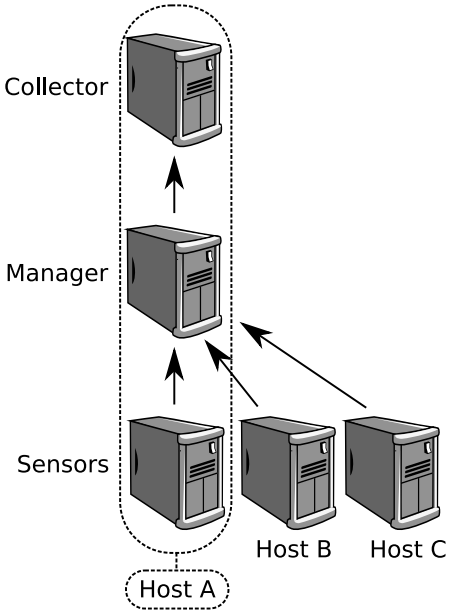

Fig. 3 Test setup

5 Experimental results

The described prototype has been validated experimentally in controlled conditions with the deployment scheme shown in Figure 3.

A single host is being used as a sensor, manager and collector, and two sensors have been installed in other machines. The manager is collecting alerts and malware samples from three sensors, and the collector is doing the same on a single manager. With this setup we can simulate:

1. the collection of binaries performed by the manager
2. the forwarding of alerts from the sensors up to the collector
3. the analysis of binaries performed by the collector
4. the collector generating a report and sending it to all the cooperating network administrators

A first test has been performed by sending known malware to the sensors. Nepenthes managed to collect the binaries and correctly issued alerts that were forwarded by the manager to the collector. Accordingly, the binary payload of the malware was transferred from the sensor to the manager and then to the collector, which proceeded with the analysis, since the hash of the binary is unknown. The payload was sent to the Norman sandbox analysis tool and the analysis outcome was interpreted and emailed to the administrators of the three simulated networks. The report included all the information gathered from the online analysis tool and was delivered timely to all the interested parties.

In order to verify the behavior of the system in a real setup, some Nepenthes sensors have been installed on home ADSL connections. This experiment led to the issuing of 3866 alerts and to the collection of 52 distinct binaries over a span of about eight hours. Seven of the collected binaries were previously unknown to our collector, and they were submitted to the available online scanning services. In many cases the behavioral analysis performed in CWSandbox has lead to the classification of malware as a worm-bot, and to the identification of several C&C hosts.

The following is a sample of the the report produced by the sandbox service:

```
0995104827bee951abc4fcc93cdf85ee :
INFECTED with W32/Malware
(Signature: W32/Malware.LNH)
    * Connects to "j4m4lz.B3D3RPIERO.INFO"
      on port 6137 (TCP).
    * Connects to IRC Server.
    * Possible backdoor functionality
      [Authenticate] port 113.
Network Activity:
  Opened listening TCP connection on port: 113
    * C&C Server: 69.64.36.188:6137
    * Server Password:
```

The worm bot tries to connect to a C&C server and opens a backdoor on port 113.

```
13ff667bebcc58253faba2313dce7b89 :
INFECTED with W32/Kut.gen1
(Signature: W32/Poebot.ADT)
  * C&C Server: 140.116.199.57:8998
Network activity
  * Server Password: PING
```

In this case it has been possible to intercept the password use by the malware for *authenticating* to its C&C servers.

```
03fb1ecf2cbcfb74ab5c29dcd247e132 :
INFECTED with W32/Endom.A (Signature: Allaple.gen1)
    * Sends data stream (76 bytes) to remote
        address "124.86.6.4",
      port 139.
    * Connects to "124.86.6.4" on port 445 (TCP).
    * Sends data stream (76 bytes) to remote
        address "124.86.8.6",
      port 139.
    * Connects to "124.86.8.6" on port 445 (TCP).
      * Sends data stream (76 bytes) to remote
        address "124.86.10.8", port 139.
    * Connects to "124.86.10.8" on port 445 (TCP).
    * Connects to "124.86.6.4" on port 9988 (TCP).
       * Sends data stream (255 bytes) to remote
         address "124.86.6.4", port 9988.
```

This result demonstrates that the proposed solution allows us to detect and block malware communications even if they rely on a complex, multi-tier control network,

as this bot does. Such solutions make it difficult to block the malware communications by only inspecting network traffic anomalies, because of the multiple servers and the different TCP ports. However, by examining the malware behavior we know at least the entrance points of the C&C network, and by blocking them we can prevent newly infected machines from joining the botnet.

6 Conclusions

This paper describes an innovative architecture to automate malware collection and classification with the purpose of implementing just in time countermeasures. It aims to benefit from the cooperation of multiple sensors spread over geographically distributed networks. The architecture is highly scalable and flexible because the number of component tiers can be adapted to the network characteristics of each participating organization.

We envision this proposal as a possible evolution of the existing malware collecting infrastructures, whose benefits are still dependent on a predominantly manual analysis of the collected samples.

The automatic analysis carried out on the collected malware allow the architecture to defeat most concealing techniques used by virus writers, since it includes the execution of the malware payload in a sandbox. This behavioral analysis avoids most hiding techniques found in modern malware. The related computational cost is reduced thanks to the collection of malware from multiple networks and to the rapid classification of duplicate binaries based on their MD5 hash.

Much attention has been paid to the security of the architecture that utilizes pairwise trust between the close components and ciphered communications. However, the necessary theoretical and practical validation of the security level of the architecture and consequent possible adjustments are left to future work.

We should also observe that we have implemented a prototype for the validation of the main ideas that are behind the proposed architecture. All experiments have obtained the expected results. On the other hand, a large scale deployment of the proposed architecture over the networks of different organizations lacks because of practical obstacles. Nevertheless, preliminary contacts with other academic

7 Acknowledgements

The authors would like to thank Saverio Verrascina [2] for his valuable help with the prototype implementation.

[2] saverio@weblab.ing.unimo.it

References

1. Sharon Gaudin (2007), Storm Worm botnet more powerful than top supercomputers, Information Week, available at http://www.informationweek.com/software/showArticle.jhtml?articleID=201804528
2. ShadowServer Foundation homepage, available at http://www.shadowserver.org
3. Nepenthes, available at http://nepenthes.mwcollect.org/
4. Xu D and Ning P (2005), Privacy-Preserving Alert Correlation: A Concept Hierarchy Based Approach, 21st Comp. Sec. App. Conf.
5. Jaeyeon Jung J and Sit E (2004) An empirical study of spam traffic and the use of DNS black lists, IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement
6. Freiling FC, Holz T, and Wicherski G (2005) Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks, ESORICS 2005: Proceedings of the 10th European Symposium on Research in Computer Security
7. Valeur F, Vigna G, Kruegel C, and Kemmerer RA (2004) A Comprehensive Approach to Intrusion Detection Alert Correlation, IEEE Transactions on dependable and secure computing, Jul-Sept 2004, Vol. 1 pp.146-169
8. When-Yi Hsin, Shian-Shiong Tseng, Shun-Chieh Lin (2005) A study of alert based collaborative defense, Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN05)
9. Zhu S, Setia S, Jajodia S (2003) LEAP: efficient security mechanisms for large-scale distributed sensor networks, CCS '03: Proceedings of the 10th ACM conference on Computer and communications security
10. Perrig A, Canetti R, Tygar JD, Song D (2000) Efficient Authentication and Signing of Multicast Streams over Lossy Channels, Proc. of the 2000 IEEE Symposium on Security and Privacy
11. Przydatek B, Song D, Perrig A (2003) SIA: secure information aggregation in sensor networks, SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems
12. mwcollect Alliance, homepage available at http://alliance.mwcollect.org/
13. Robert Tappan Morris (1988), The Morris Worm, homepage available at http://www.morrisworm.com/. Cited 17 Jan 2008.
14. Internet Storm Center (2004), Sasser Worm, LSASS exploit analysis, available at http://isc.sans.org/diary.html?date=2004-04-30
15. Computer emergency Response Team (2000), CERT®Advisory CA-2000-04 Love Letter Worm, available at http://www.cert.org/advisories/CA-2000-04.html
16. Symantec™(2004), W32.Wallon.A@mm worm description, available at http://www.symantec.com/security_response/writeup.jsp?docid=2004-051112-0815-99
17. US-CERT (2004), Technical Cyber Security Alert TA04-356A (Santy worm), available at http://www.us-cert.gov/cas/techalerts/TA04-356A.html
18. Wikipedia (2007), Timeline of notable computer viruses and worms, available at http://en.wikipedia.org/wiki/Timeline_of_notable_computer_viruses_and_worms#2006
19. IETF Intrusion Detection Working Group (2007) The Intrusion Detection Message Exchange Format (IDMEF), available at http://tools.ietf.org/html/rfc4765
20. Norman ASA, homepage available at http:/www.norman.com/
21. Prelude Hybrid IDS project, homepage available at http://www.prelude-ids.org/
22. Virustotal, a malware analysis service offered by Hispasec Sistemas, available at http://www.virustotal.com/
23. Norman SandBox Information Center, available at http://sandbox.norman.com
24. CWSandbox, Behavior-based Malware Analysis remote sandbox service, homepage available at http://www.cwsandbox.org/