상황 지식 축적에 의한 알려지지 않은 위협의 검출*

박길철** · 하미드 쿡*** · 김양석*** · 강병호*** · 육상조** · 이 극**

요 약

알려지지 않은 불분명한 위협을 검출하는 내는 것은 모순이다. 존재하는 것이 알려지지 않았다면 어떻게 찾아 낼 것인가? 그것은 알려지지 않은 위험을 아주 짧은 시간 안에 위협을 정의(identification)을 할 수 있으면 가능 할 수 있을 것이다. 본 논문은 위험 검출 기법을 만들어 네트워크상의 알려지지 않은 위험에 대해 유연하게 대처할 수 있는 시스템 개발에 도움을 줄 수 있게 하기 위해 연구되었다. 이 시스템은 알려지지 않은 위험을 탐지 하기위하여 동적이고 유연한 상황 지식을 가진 로직을 가지고 시스템을 감시한다. 시스템은 새로운 위협의 검출뿐만 아니라 빠르고 효과적인 방법으로 위협에 대처할 수 있다.

Unknown Threats Detection by Using Incremental Knowledge Acquisition

Gil-Cheol Park** · Hamid B. M. Cooke*** · Yangsok Kim***
Byeong Ho Kang*** · SangJo Youk** · Geuk Lee**

ABSTRACT

Detecting unknown threats is a paradox ; how do you detect a threat if it is not known to exist? The answer is that unknown threat detection is the process of making a previously unknown threat identifiable in the shortest possible time frame. This paper examines the possibility of creating an unknown threat detection mechanism that security experts can use for developing a flexible protection system for networks. A system that allows the detection of unknown threats through monitoring system and the incorporation of dynamic and flexible logics with situational knowledge is described as well as the mechanisms used to develop such a system is illustrated. The system not only allows the detection of new threats but does so in a fast and efficient manner to increase the available time for responding to these threats.

Key words : Threats Management, Network, MCRDR(Multiple Classification Ripples Down Rules)

---

* This work was supported by a grant from Security Engineering Research Center of Korea Ministry of Commerce, Industry and Energy.
** Professor of Hannam University
*** University of Tasmania, Australia

1. Introduction

Unknown threat detection has become a highly investigated area of the Internet security community [1-3]. Unknown threat detection is the process of identifying a previously unknown attack by finding the factors that will make it identifiable for every occasion it appears. This process seems paradoxical ; how can unknown threats be identified if they are unknown? How can these unknown threats be detected if they are not known to exist? If an unknown threat is created perfectly, it can spread without detection until it is time to strike. However, just as flawless software cannot be created ; there is no perfect unknown threat. The unknown threat will have some weakness within its operation that will indicate that something suspicious is occurring. The activities of the threat appear, analogous to a human illness, as symptoms on the host machine. After careful examination these symptoms will identify the existence of a problem and hence the unknown threat.

An intrusion detection system (IDS) is a device that monitors the activity of a network in an attempt to identify malicious activity [4]. Generally IDS can be classified into two types : signature based and anomaly based. The signature based IDS is less complex than the anomaly based IDS and so it can be implemented successfully in both perimeter based and host based security strategies[5]. The anomaly-based IDS is complex and processor intensive and is typically seen in the perimeter based security strategy[7].

This research focuses on reducing response times and decreasing the time it takes to classify an unknown attack. In addition to the decreasing identification time, the unknown threat detection system will ensure that while it identifies unknown threats, normal conditions are not identified as threats and that a threat is not classified as normal traffic. This paper, in Section 2, the aims of the research system are described with reference to the background and the issues pertaining to the domain. Experiment results and their analyses are discussed in Section 3. Finally conclusions will be provided in Section 4.

2. Methodology

2.1 Aim of the System

To be successful in detecting unknown attacks the system will need to possess the following features :

Firstly, the system is able adapt to the dynamic environment of networks. This caters for the fact that traffic on networks is always changing due to factors such as new network protocols and software. In addition to this the attacks that are occurring are changing dynamically as old attacks mutate and new attacks are formulated. To facilitate this feature the system uses an expert system in order to develop a constantly evolving knowledge base. The evolving knowledge base requires a system to provide easy maintenance of the knowledge it contains. To provide this feature for the knowledge base the system will use the multiple classification ripples down rules (MC RDR) knowledge acquisition technique[8, 9].

Secondly, a monitoring system that creates too many false positives will induce a dismissive attitude in the security expert. As this monitoring

system produces more false alarms the security expert will be more likely to ignore each alarm as another false positive. This repeated generation of false alarms within a monitoring system occurs because of over generalized logics that are used for identifying an attack. In addition the thresholds for the network activities could be set too low. The system solves this problem by using a flexible logic engine where the thresholds can be dynamically adjusted and the logics adapted to better identify the threats.

2.2 MCRDR Knowledge Acquisition

2.2.1 Rule Tree

In the implemented system a tree data structure was created in addition to the functions for interacting with the rule structure. The process of rule addition, refinement and manipulation within the system are transparent to ensure that the knowledge base does not get corrupted and to not burden the expert with undue operations.

2.2.2 Inference

The process of knowledge acquisition in terms of the security monitoring system is as follows : The packet capturing module is started and it begins acquiring network traffic data. The case time period elapses and the case data is created. The case data is fed through to the inference engine and the MCRDR process begins. Firstly each rule on the first level is compared with the case data to see whether the conditions are satisfied. If the rule is satisfied it is fired and the children of this rule are considered. This process repeats with the children and so forth until none of the fired rule's children has fired.

2.2.3 Knowledge Acquisition

The expert determines whether the conclusion was correct or not by observing the case data and what conclusions where produced with which rule. If it was correct, process the other case/s otherwise the conclusion was not correct and so the expert selects which rules concluded incorrectly or to adds a completely new classification. If the expert has determined that a rule has concluded incorrectly, the system show the difference list between one of the cornerstone cases that exist for the incorrect rule and the current case data. The expert can select the cornerstone cases that he/she wishes to let match this rule's conditions. If the expert selects all of the remaining cornerstones to let match then the new rule is valid and is added as a refinement (child) of the incorrect rule with the selected conditions and the current case data as the cornerstone. Otherwise the expert is presented with the differences between the current case and one of the remaining cornerstones. This process continues until there are no cornerstones in the list or the expert select the all of the remaining cornerstone case data that will be allowed to match. Then the new rule is added as a refinement of the incorrect rule with the conditions selected and the current case data as the cornerstone.

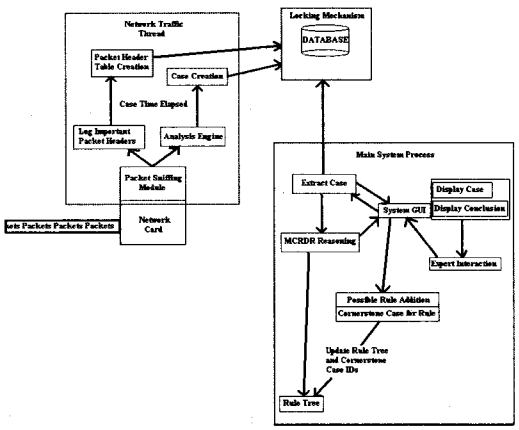
2.2.4 Validation Process

In addition to the validation process ensuring that the new logics do not incorrectly fire an existing logic the system also checks to see whether the conditions are valid. This is because the conditions are displayed but they are able to be edited. If the edited condition does not satisfy the existing case data then it is deemed invalid and

must be corrected. The invalidation continues until the edited condition satisfies the existing case data.

2.3 System Design and Implementation Issues

(Figure 1) illustrates the abstract design of the system. The thing to observe is that the network traffic thread, and hence packet capturing, is executed in parallel with the main system thread. Parallel execution allows the real-time execution of the monitoring system without packet loss. The following issues were raised while implementing the system.



(Figure 1) Overall System Design

2.3.1 Packet Loss

In order to detect attacks in progress the system must collect every packet that is received or sent on the host network. If any packets are not captured then it will impede the knowledge acquisition process. The lost packets may contain information that is vital to an attack pattern being detected. With this issue in mind the system was designed such that the packet capturing engine is run in parallel with the other system processes.

2.3.2 Case Window Size

The time interval between cases being created is the window size. The window size determines the amount of network traffic is inputted into the analysis engine before the case is created. The selection of the window period will affect the overall system in a number of ways. Having the window size set to a small value will mean that there won't be much data to build a case from. A small window size may also mean that anomalous patterns may be misclassified as normal traffic as the expert will not see all of the information relevant to an attack. Having the window size set to a large value will mean that there will be a significant amount of data processed before a case is created. A large window size forces the system to deal with a significant amount data and the resultant memory requirements. In the experiments the window size was set to twenty. The packet capturing module obtains information from the network for twenty packets and then it analyses the data, builds summary information of that data, and then creates a case. While the analysis engine computes the summary information the packet capturing module continues to gather network information for the next case.

2.3.3 Storage of the Captured Data

Every packet has headers that contain important information necessary for transmission through a network medium. Once collected the network header data that the packet capturing

module gathers is stored in a database. This information is stored in the database in two parts, the important packet information and the case data created from the network header data. In addition to the storage of packet information in the database every single packet and most of the header information is stored in a separate text file. This is so that the expert can get an extremely detailed analysis of the traffic.

2.4 Experiment Design

In order to test a system that identifies unknown threats it would be impossible to have an unknown threat attack situation. The assumption used for testing is that at some point in computing history the attacks were not known and had to be identified through some means. So testing of the system was carried out by identifying candidate threats and creating or downloading the source code for an attack. Meanwhile the expert will identify the patterns and subsequent conditions that best represent and identify the attack from the norm. The candidate threats were selected from a varying mix of complexity and what type of systems the threat would typically attack, such as mainly server oriented. The following two threats were used in the experiments.

2.4.1 Denial of Service

Two different forms of this threat were selected for testing. First a basic denial of service attack, ping flooding, which can affect both normal type hosts and servers was selected. Next a server oriented denial of service attack that is directed at E-Mail servers, mail bombing, was selected.

2.4.2 Worm Attack

Worms are a form of virus that can propagate using their own means. A Worm will scan for active vulnerable machines and then mobilizes to infect it. In this experiment a worm simulation was created and used by a monitored host to attack another monitored machine. The worm simulation provided was not identified ahead of time and the expert had to identify discriminating patterns from analysis. The worm displayed curious properties that are detailed in the results section.

3. Results

3.1 Rule Creation

The monitoring system captures the network data in real-time and buffers the cases in a database so that the expert can concentrate on the case that he/she is analyzing. What typically occurs is the expert will fall behind the current cases being collected. However when the monitoring system concludes correctly on a case, and the expert has confirmed this, then the expert will catch up to the current traffic analysis data. So the more correct conclusions the system produces, the closer the expert will be to approaching the currently captured data. The following sections illustrate the time it took the expert to create rules, in the peak threat and the incremental threat pattern types.

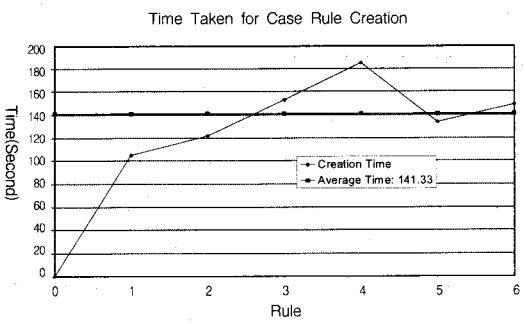
3.1.1 Peak Threat Patterns

The peak threat patterns are easier to discover and classify as these patterns contain conditions

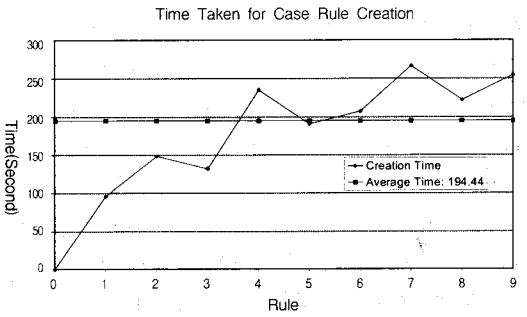
that exhibit obvious activity spikes when the host is under attack. The expert simply observes the spiking condition/s and creates rules based on these conditions. (Figure 2) (a) shows the times taken to add rules in the Ping Flood experiment. As can be observed the time taken to add rules to identify this Denial of Service attack averaged around 2 minutes 30 seconds.

3.1.2 Incremental Threat Pattern

The incremental threat patterns are more difficult to create rules for due to the more intensive analysis required by the expert to identify the pattern. The in-depth analysis is required because the expert wanting to ensure that correct conditions are being selected to identify the threat to reduce false positives. Also, due to the condition patterns being more difficult to identify, the expert will usually create more abstract rules. As more rules are created the average number of cornerstone cases from which to validate against increase. This has the effect of lengthening the validation/revalidation cycle. (Figure 2) (b) shows the times taken to add rules in the Worm experiment. As can be observed from the results the average time to add a rule in the Worm experiment was around 3 minutes 15 seconds. Also there exists a slight trend for the rule creation time to increase as the number of rules in the system increased and hence the number of cornerstone cases that need to be validated against also increased. Overall rule addition within the monitoring system averaged around three minutes. The majority of this time was used by the expert to ensure analysis all of relevant data so that no threat patterns were left undiscovered and normal traffic conditions misclassified. The process of rule addition within the monitoring system is trivial and the time taken for rule creation is comparable to other existing systems. Observations of experts maintaining the GARVIN ES1 system showed that the time taken to add rules for each case, whether refinements of previous rules or completely new classifications, averaged around ten per hour [10].



(a) Ping Flood



(b) Worm

(Figure 2) Rule Creation Time

3.2 Accuracy of Threat Detection

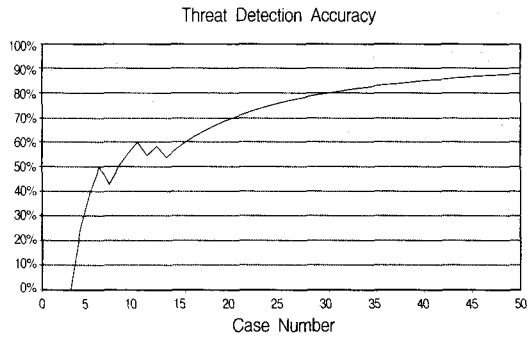
The accuracy of the monitoring system in threat detection is a true measure of the performance and capabilities of such a system. If the monitoring system was inaccurate in identifying threats then the usability of such a system rapidly diminishes. The accuracy of threat detection

in the system is dependent on the expert being able to establish the logics that will identify the threat now and in the future. The expert does not necessary have to know about every single threat, known and unknown, rather he/she has to apply an understanding of the situation aspects that will identify the threat symptoms.

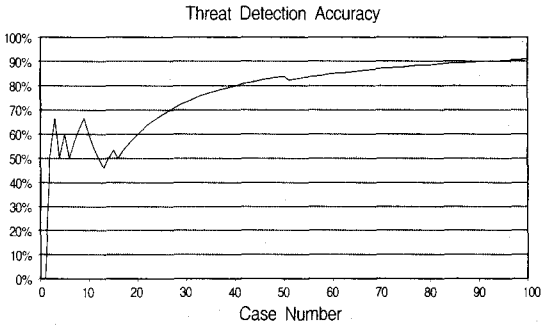
(Figure 3) (a) shows the accuracy of the monitoring system in identifying a Peak Threat Pattern ; ping flooding. The results show that at first the monitoring system cannot identify the threat at all. This is due to there being no logics and situational data to identify the threat. However, as the expert begins adding the logics and knowledge for threat identification the accuracy of the monitoring system improves dramatically. This improvement progresses to a point where the monitoring system is identifying the threat around 90% of the time. A similar scenario occurs for the threat detection accuracy for the incremental threat patterns.

(Figure 3) (b) shows that, although it takes longer to achieve consistent accuracy above 50% due to the expert creating more abstract logics, the monitoring system eventually reaches 91% accuracy in identifying the Worm threat. Results of this nature show that, even though it may take the expert some time to develop the correct logics for identifying any threat, in the long term the monitoring system will be able to adapt and identify any threat with a minimum of false alarms.

The results from the experimentation show that generalized threat patterns at the network level of a host system do exist. The results also indicate that the monitoring system is flexible and can dynamically adapt its logics and thresholds to identify threats on the individual hosts. In addition to these factors the monitoring system reduces the burden upon an expert in the task of creating and refining the rules within the system, to a point where adding rules becomes a trivial task that can be achieved in a minimal amount of time. The monitoring system also achieves these features and functions while still maintaining threat detection accuracy and keeping false alarms to a minimum.



(a) Ping Flooding



(b)Worm

(Figure 3) Detection Accuracy

4. Conclusion

This research focused on the production of a monitoring system that detects unknown network threats. The monitoring system is driven

by logics which allow the expert to correctly identify network threat patterns as they manifest on a host. The logics that are developed over time will give the expert a clear understanding of the way in which the unknown threats behave on the various hosts. The addition of these logics within the system is dynamic and flexible ; allowing the expert to create generalized rules that can be further refined or a series of quite specific threat defining rules. The results show that the addition of rules and situational knowledge within the monitoring system is a trivial task for the expert. This gives the expert more time to extract, adapt and apply the knowledge throughout the monitored network. This reduces the response time to the previously unknown threat on the monitored network. Having an effective response to an unknown threat requires that the system that alerts the expert must be accurate. The experiment results show that the accuracy of the monitoring system, after the addition of some of the threat defining logics, is high. The accuracy of the system is always improving as the system is run and the logics are refined. This accuracy will keep the number of false positives (false alarms) to a minimum so that the expert is not conditioned into ignoring the system alerts. Finally it can be concluded that the attainment of the previous aims has proven that the use of MC RDR within the network security domain was successfully applied and is valid within this domain.

References

[1] A. K. Ghosh, J. Wanken, and F. Charron, "Detecting anomalous and unknown intrusions against programs", in 14th Annual Computer Security Applications Conference (ACSAC '98), 1998.

[2] P. W. Hodgson, "The threat to identity from new and unknown malware", BT Technology Journal 2005, Vol. 23, No. 4, pp. 107-112, 2005.

[3] S. Singh, et al., The EarlyBird System for Real-time Detection of Unknown Worms, 2003, UCSD.

[4] R. Janakiraman, M. Waldvogel, and Q. Zhang, Indra : A peer-to-peer approach to network intrusion detection and prevention in Proceedings of the Twelfth International Workshop on Enabling Technologies : Infrastructure for Collaborative Enterprises 2003 IEEE Computer Society, p. 226, 2003.

[5] S. Lorimer, A Real-Time IDS Monitoring Multiple Gateways, in Computing, University of Tasmania : Hobart. p. 73, 2003.

[6] C. P. Pfleeger and S. L. Pfleeger, Security in computing, 3rd Int ed. 2003, Upper Saddle River, N. J. : Prentice Hall PTR. xxix, 746, 2003.

[7] S. Axelsson, Intrusion Detection Systems : A Survey and Taxonomy, in Department of Computer Engineering, Chalmers University of Technology : Goteborg, p. 27, 2000.

[8] B. Kang, P. Compton, and P. Preston. Multiple Classification Ripple Down Rules : Evaluation and Possibilities, in 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, University of Calgary, 1995.

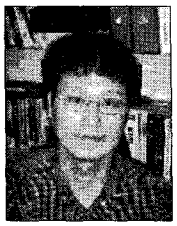
[9] B. H. Kang, W. Gambetta, and P. Compton, Verification and validation with ripple-down

rules. International Journal of Human-Computer Studies, Vol. 44, No. 2, pp. 257-269, 1996.

[10] P. Compton, and R. Jansen, A philosophical basis for knowledge acquisition. in 3rd european knowledge acquisition for knowledge based systems workshop, 1989.

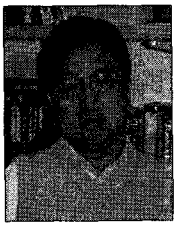
Gil-Cheol Park

1979년 ~ 1983년 HanNam Univ. (BA)
1983년 ~ 1985년 SungSil Univ., Graduate School (MA)
1994년 ~ 1998년 SungKunKwan Univ., Graduate School (Ph.D)
1985년 ~ 1990년 SamSung Advanced Institute of Technology
1991년 ~ 1996년 DaeKyo Computer Co., LTD.
1996년 ~ 1998년 HanSeo University, Professor
2005년 Visiting Professor of UTAS., Australia
1998년 ~ Now HanNam Univ., Professor



Yang Sok Kim

1987년 ~ 1995년 University of Seoul
1994년 ~ 2001년 Hyundai Information Technology Co., Ltd
2001년 ~ 2002년 E-2 Corporation
2002년 ~ 2004년 University of Tasmania (MA)
2005년 ~ Now University of Tasmania, Australia, PhD Student



Byeong Ho Kang

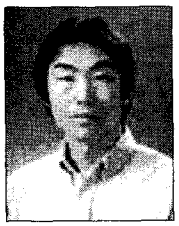
1982년 ~ 1988년 Pusan National University
1988년 ~ 1990년 University of Tasmania, Australia (MA)
1990년 ~ 1995년 New South Wales Univ., Australia (PhD)
1995년 ~ 1996년 Hitachi Advance Research Lab, Japan, Research Fellow
1996년 ~ 1999년 Hoseo University, Assistant Professor
2000년 ~ Now University of Tasmania, Australia, Senior Lecturer

Hanmid B. M. Cook

2002년 ~ 2005년 University of Tasmania
2005년 ~ 2006년 University of Tasmania (MA)

Sang Jo Youk

1983년 ~ 1990년 HanNam Univ.
1992년 ~ 1994년 HanNam Univ., Graduate School (MA)
1997년 ~ 2004년 HanNam Univ., Graduate School (Ph.D)
2004년 ~ 2005년 YoungDong Univ, Professor
2006년 ~ Now HanNam Univ., Professor



Geuk Lee

1978년 ~ 1983년 KyungPook Univ.
1984년 ~ 1986년 Seoul Univ., Graduate School (MA)
1988년 ~ 1993년 Seoul Univ., Graduate School (Ph.D)
1995년 ~ 1996년 Visiting Professor., Univ. of CA. at Irvine
1986년 ~ Now HanNam Univ., Professor