IEICE TRANS. INF. & SYST., VOL.E92–D, NO.4 APRIL 2009

LETTER

A Bio-Inspired Approach to Alarm Malware Attacks in Mobile Handsets

Taejin AHN[†a)], *Nonmember and* **Taejoon PARK**[††b)], *Member*

SUMMARY With proliferation of smart handsets capable of mobile Internet, the severity of malware attacks targeting such handsets is rapidly increasing, thereby requiring effective countermeasure for them. However, existing signature-based solutions are not suitable for resource-poor handsets due to the excessive run-time overhead of matching against ever-increasing malware pattern database as well as the limitation of detecting well-known malware only. To overcome these drawbacks, we present a *bio-inspired* approach to discriminate malware (non-self) from normal programs (self) by replicating the processes of biological immune system. Our proposed approach achieves superior performance in terms of detecting 83.7% of new malware or their variants and scalable storage requirement that grows very slowly with inclusion of new malware, making it attractive for use with mobile handsets.

key words: *bio-inspired approach, biological immune system, mobile immunity, mobile handsets, mobile malware*

1. Introduction

The traditional voice-call-only cellular phones are rapidly evolving into mobile handsets supporting complicated multimedia and e-commerce applications as they become smarter and capable of high-speed Internet. This trend, however, inevitably invites malware (viruses, worms, Trojans and spyware) attacks targeting specifically at mobile handsets as manifested by their history dating back to the first mobile malware appearance in 2004 and ever since witnessing considerable increase in the number of mobile malware [1], [2]. As such, the mobile malware will soon become a serious threat to the users as well as to the handset manufacturers and network operators since it can steal personal data, remotely take over the handsets, propagate over Bluetooth/MMS and send bulky SMS traffic.

Existing anti-malware solutions [3] are basically based on pattern-matching scanning that stores a database of known malware patterns or signatures in the handset and compares all incoming data with this database to see if there's a match. But, unfortunately, they are not suitable for battery-powered handsets because they consume significant amount of battery energy as the database constantly builds up with the addition of new signatures, cannot detect new or polymorphic malware not listed in the database, and require manual operations of malware analysis by the human experts and of database update by the users.

Motivated by these limitations, we propose a novel *bio-inspired* approach to replicate the processes and capabilities of biological immune system that successfully protects human body from over 10^{31} viruses. This approach leads to the development of an anti-malware solution, called *mobile immunity*, for mobile handsets to efficiently detect and quarantine the incoming malware, addressing the above-mentioned drawbacks of existing solutions. The mobile immunity is built on top of a classification algorithm to distinguish *nonself* (malware samples) from *self* (normal handset programs), i.e., it determines if a specific binary code from the incoming data belongs to either self or nonself, and then declares the latter to be potentially harmful. Our experimental results show that it can detect, with high confidence (up to 83.7%), new or polymorphic malware that have evaded the signature-based techniques and achieves scalability of storage overhead that grows very slowly with addition of new malware samples.

The rest of the paper is organized as follows. Sections 2 and 3 overview the related work and the biological immune system. Section 4 describes our proposed approach while Sect. 5 presents the performance evaluation results. Finally, Sect. 6 concludes the paper.

2. Related Work

The mobile malware first appeared in June 2004 as a proof-of-concept virus that infects Symbian-based smart phones and spreads over Bluetooth and MMS. From then on, we have seen rapidly increasing number of mobile malware families and their variants that reached 35 and 186, respectively, by the end of 2006 [2].

Although mobile malware have not yet caused serious outbreak, their threats are far more real and handsets are expected to become targets of increasing number of mobile malware for the following reasons. First, there will be more and more software vulnerabilities/bugs to be exploited by the malware writers as the handsets get more intelligent. Second, as the mobile Internet services like WiMAX gain popularity, the handsets will face the huge volume of web-based attacks and worm propagations from the Internet. In addition, peer-to-peer or ad hoc communications, e.g., using Bluetooth, make malware outbreak to closely track human mobility patterns [4]. Third, massively inter-networked ubiquitous environments will make it easier for the malware writers to create crossover malware that can infect both

Manuscript received June 12, 2008.

[†]The author is with Samsung Electronics Co., P.O. Box 111, Suwon, Korea.

[††]The author is with Korea Aerospace University, Gyeonggi-do, 412–791 Korea.

a) E-mail: taejin.ahn@samsung.com

b) E-mail: tjpark@kau.ac.kr

DOI: 10.1587/transinf.E92.D.742

handsets and PCs and propagate across network borders. Fourth, while Symbian phones have been the main target of malware writers due to their large volume in the current market, their interests will quickly move to other popular OS's such as mobile Windows and Linux.

Several approaches have been proposed to defeat new or polymorphic malware, to which signature-based solutions get defenseless due to the lack of matching signatures in the database. A heuristic approach [5] uses n-gram (where $n = 3, 4$) appearing more than once in the malware as the feature for detection. Computer immune systems for Internet [6] draw inspiration from biology to make computing systems more biological in nature. Recently proposed behavioral detection approach for mobile handsets [7] monitors the run-time behavior of programs to tell their maliciousness. Our approach differs from the above approaches in that we precisely replicate the immune system of human body via the experimental study of feature selection and immunity database construction (similarly to the development of new antibody).

3. Biological Immune System

The biological immune system had evolved for over 2 million years to better protect humans from external attacks of infectious pathogens. The human cell contains 3×10^9 letters of DNA base pairs to encode protein information. Among them, 3×10^7 DNA letters can be converted to 1.5×10^6 protein letters, which collectively represent the self-originated material in the human body. Likewise, protein letters from pathogens constitute the nonself-originated material. That said, the basic principle of biological immune system is to distinguish nonself materials from self materials, which in turn provides a key function for its self-learning nature that enables people to naturally resist pathogens that they have never been exposed to.

As shown in Fig. 1 (a), the biological immune system consists of two stages, each processing molecular information from either self or nonself. The first stage is *antigen presentation*. It chooses and presents short fragments of proteins, called antigens, from both self- and nonself-originated proteins via the major histocompatibility complex (MHC) proteins that act as signposts for displaying the selected antigens on the cell's surface [8]. The antigen selection criteria of MHC are given by the regular expressions of two types [9]:

- **Type-1**: $*-*-\mathsf{Y}-*-[\mathsf{YF}]-*-*-*-[\mathsf{LMIV}]$
- **Type-2**: $*-*-*-*-*-*-\mathsf{N}-*-*-*-*-*-[\mathsf{LMIV}]$

To illustrate the notation, $*$ denotes any of the alphabet letters referring to amino acids, while [LMIV] means one of four letters, L, M, I, and V qualifies for the position. According to these criteria, the protein fragment ALGQNLGYV is a valid type-2 expression, while ALGQNLGYX is not because the last letter X violates the rule [LMIV].

The *negative selection* is the next stage of immune system. It recognizes the presented antigens and classifies them as either self or nonself by selectively training T cell receptors (TCRs). At the beginning, it randomly generates a large number (about 10^{14}) of different TCR types, where each TCR has a unique short protein code that reacts only to a certain (presented) antigen. It then uses reaction affinity to determine the fate of each TCR. That is, it kills a TCR if having too high affinity to the self proteins, which means only those TCRs with small amount of affinity against the presented self proteins survive. These survivors are called mature TCRs and play the role of classifiers that distinguish self proteins from nonself proteins. Thanks to the way mature TCRs are chosen, none of them exhibits high affinity to the self proteins. However, it is possible for a nonself protein to show high affinity to at least one of the mature TCRs (if unseen before). Hence, the mature TCRs consistently conduct surveillance and monitoring on the presented fragments of proteins to determine whether or not it is originated from nonself.

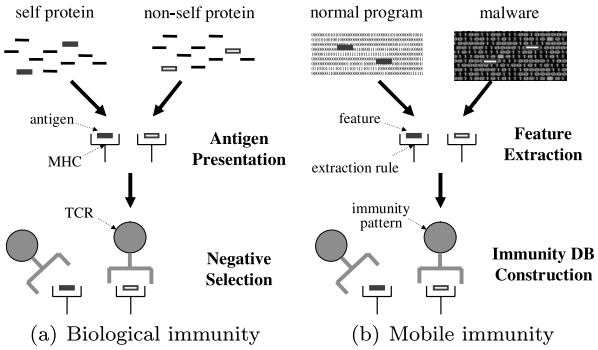

Fig. 1 Biological vs. mobile immunity.

4. The Proposed Approach

4.1 Motivation

We have focused on the capability of biological immune system to recognize the formerly unseen viruses or pathogenic bacteria to address the fundamental challenge of anti-malware research: "how to effectively defend against new malware and unidentified variants of existing malware?" To apply the biological principles, let us first make an analogy between biological and mobile immunity. An executable program code plays the role of biological protein sequence, and hence, self and nonself can respectively be defined as a collection of all legitimate handset programs that can be harmlessly installed/executed on handsets and a collection of all malware samples. We also use terms, a *feature*, an *extraction rule* and an *immunity pattern* to refer to the digital-world counterpart of biological terms, an antigen, MHC and TCR, respectively. The mobile immunity then answers the following two questions:

1. how to define a rule to filter short feature fragments from the program code (either a normal program or a malware sample), and

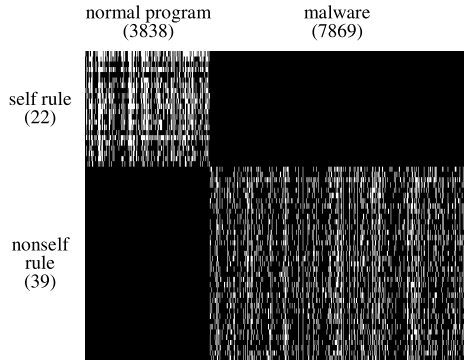

Fig. 2 Determination of self and nonself rules given the training data of 3838 normal programs and 7869 malware samples.

2. how to construct a classifier consisting of immunity patterns to distinguish if the thus-filtered features belong to the nonself category.

Accordingly, as shown in Fig. 1 (b), the mobile immunity is comprised of two stages, *feature extraction* and *immunity database construction*, which are analogous to the antigen presentation and negative selection stages of biological immunity.

4.2 Feature Extraction

The human body can have up to 18 MHCs and each MHC can extract 10^{10} possible protein fragments, resulting in a maximum of 1.8×10^{11} distinct protein fragments [10]. To replicate this characteristics, we define a feature as a 6-byte data sequence and design a set of feature extraction rules such that the maximum number of distinct features is similar to that of biological immunity. This condition is met by employing 450 distinct rules, each specifying the values of 2 bytes at the given positions of the 6-byte sequence, e.g. 0xBA and 0x64 in the first and fourth byte positions, which yields $450 \times 2^{4\times8}$ ($\cong 1.8 \times 10^{11}$) features.[†]

The feature extraction rules are further divided into self and nonself rules, which are tailored to extracting features pertinent to (or more frequently found from) the normal programs and malware, respectively. To determine the self/nonself rules optimized to the given training data, we experimented with 3838 normal programs and 7869 malware samples in the desktop environment as follows. For each of the candidate rules (with varying values of fixed bytes), we counted how many features are extracted from the normal programs and malware, and classified the rule as a self (nonself) rule if the number of normal program-originated features is much larger (smaller) than the number of malware-originated features. Consequently, as plotted in Fig. 2, we found 22 self rules and 39 nonself rules that respond solely to their matching training set while extract none from the opposite set. Note that we can easily extract up to 450 rules by expanding the size of training data and allowing the extraction of jitters (from the opposite set). This result shows the discrimination of self and nonself can be achieved by judiciously selecting the feature extraction rules.

4.3 Immunity Database Construction

We first generate a reasonable number of random (and unique) immunity patterns p_i and then perform the negative selection process. The random generation is essential to reproducing the natural resistance mechanism of human body and enabling each and every handset to have its own pool of immunity patterns. That is, if two handsets randomly generated immunity pattern profiles, $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A}$ might have patterns sensitive to an unknown malware while $\mathcal{B}$ might not. This means the handset with $\mathcal{A}$ has acquired an innate immunity against that malware.

In the negative selection, we first extract features f_j from the normal programs by processing them with thus-constructed rules. Then, for each p_i, we measure the affinity between p_i and f_j for all j by calculating the Euclidian distance $d(p_i, f_j)$ between them, and select p_i and add it to the immunity database only if the affinity values are lower than a certain threshold τ, i.e. $\max_j d(p_i, f_j) < \tau$. After testing all p_i's, the immunity database is built from the immunity patterns selected as above. Accordingly, the immunity database is a collection of patterns insensitive to the self-originated features while exhibiting high affinity to the nonself-originated ones.

4.4 Malware Detection and Cleanup

We present two detection schemes based on the extracted features and the immunity database as follows.

Frequency-based detection: extracts features from the input data by applying both self and nonself rules and counts how many features are produced by each of the rules. It then diagnoses the data as being potentially malicious if there are more nonself-originated features than self-originated features. This scheme is based on the premise that the former is coupled with malicious functionalities.

Immunity database-based detection: extracts features v_j from the input data under verification and calculates affinity values $d(p_i, v_j)$ between the immunity patterns p_i in the database and the features v_j for all i, j. It finally declares the data is suspected to be malicious if any of v_j's shows noticeable affinity with immunity database, i.e., $d(p_i, v_j) \geq \tau$. This scheme can be viewed as an anomaly detection approach that detects significant deviations from the normal profile.

Once detected, further inspection can be taken on the input data to make sure they are indeed malicious.

[†]Note that there can be many possible constructions of immune systems depending on the choices of the feature length and the number of rules.

5. Performance Evaluation

We evaluated the performance of our approach in terms of the capability of feature extraction and accuracy of detection. We experimented with a total of 19573 malware and 4286 normal programs in the Win32 desktop environment due mainly to the limited availability of handset applications and mobile malware.

We first determined a total of 200 self/nonself rules (100 each) from the malware and normal program samples and found they agree with Fig. 2. We also found 95.4% (18681/19573) of the malware samples produced more than one nonself features while the rest 4.6% didn't due to their very short length.

We next measured the capability of above-constructed rules to detect new malware and previously unseen normal programs. To this, we tested 646 malware samples and 646 normal programs that were not used in the generation of rules. Table 1 summarizes the test result. It shows 46.7% (302/646) of the new malware produced features matching the nonself rules only, and hence, could directly be classified as malware. Likewise, we could make sure that 25.5% (165/646) of the normal programs were indeed genuine. However, 44.5% and 73.9% of the malicious and normal programs generated features from both self and nonself rules, requiring deeper inspection, e.g., based on the frequency-based method. Note that we failed to extract features in 8.6% and 0.4% of the test samples.

Table 2 shows the performance of frequency-based detection against the same set of samples as that of the previous experiment. By employing the ratio of the count of nonself-originated features to the count of self-originated features, we achieved the malware detection rate of 83.7% (541/646) and the missed detection rate of 8.5%, which means we can detect up to 83.7% of the new or polymorphic malware having evaded the signature-based detection. By contrast, the false positive rate (declaring normal programs to be malicious) was as high as 20.6%. To mitigate this, we may simultaneously use an additional countermeasure such as immunity database-based detection.

Finally, the memory overhead is fairly small and doesn't grow significantly with addition of new malware for the following reasons. First, the memory requirement for the feature extraction rules is limited by 2.7 KB (in case of 450 rules). Second, the size of immunity database depends on the choice of τ and the number of immunity patterns initially generated, but not on the malware samples themselves.

Table 1 Classification of new malware and normal programs (646 each) not used in the rule construction.

	new malware	unseen programs
nonself-rules-only match	46.7%	0
self-rules-only match	0	25.5%
both-rules match	44.5%	73.9%
no match	8.6%	0.4%

Table 2 Performance of frequency-based detection against new malware and normal programs.

ratio of nonself- to self-originated features	new malware	unseen normal programs
> 1	83.7%	20.6%
< 1	8.5%	78.1%
= 1	0.9%	0.7%

6. Conclusion

In this paper, we presented a novel bio-inspired approach applicable to the resource-limited mobile handsets. The proposed approach offered rules for extracting features to discriminate self and nonself and a mechanism for constructing the immunity database, which together achieved the detection rate of 83.7% for new or polymorphic malware.

References

[1] S. Coursen, "The future of mobile malware," Network Security, vol.2007, no.8, pp.7–11, Aug. 2007.
[2] A. Gostev, "Kaspersky security bulletin 2006: Mobile malware," http://www.viruslist.com/en/analysis?pubid=204791922
[3] L. Tung, "Signature-based antivirus is dead: Get over it," ZDNet Australia, April 2008.
[4] J. Kleinberg, "The wireless epidemic," Nature (News and Views), vol.449, pp.287–288, Sept. 2007.
[5] W. Arnold and G. Tesauro, "Automatically generated win32 heuristic virus detection," Proc. Virus Bulletin Conference, Sept. 2000.
[6] A. Somayaji, S. Hofmeyr, and S. Forrest, "Principles of a computer immune system," Proc. New Security Paradigms, Sept. 1997.
[7] A. Bose, X. Hu, K.G. Shin, and T. Park, "Behavioral detection of malware on mobile handsets," Proc. ACM MobiSys, June 2008.
[8] D.R. Madden, "The three-dimensional structure of peptide-mhc complexes," Annu. Rev. Immunol., vol.13, pp.587–622, April 1995.
[9] K. Falk, O. Rotzschke, S. Stevanovie, G. Jung, and H.G. Rammensee, "Allele-specific motifs revealed by sequencing of self-peptides eluted from MHC molecules," Nature, vol.351, pp.290–296, May 1991.
[10] J.A. Borghans, A.J. Noest, and R.J. DeBoer, "Thymic selection does not limit the individual MHC diversity," Eur. J. Immunol., vol.33, no.12, pp.3353–3358, Dec. 2003.