# On-line Shared Platform Evaluation Framework for Advanced Persistent Threats

**Dongsik Sohn[1], Taejin Lee[2], Jin Kwak[3*]**
[1] ISAA Lab., Department of Computer Engineering, Ajou University, Suwon, South Korea
[e-mail: over0033@gmail.com]
[2] Department of Computer Engineering, Hoseo University, Cheonan, South Korea
[e-mail: kinjecs0@gmail.com]
[3]Department of Cyber Security, Ajou University
Suwon, South Korea
[e-mail: jkwak.security@gmail.com]
*Corresponding author: Jin Kwak

*Received May 14, 2018; revised July 26, 2018; revised November 10, 2018; revised December 19, 2018; accepted December 23, 2018; published May 31, 2019*

## Abstract

Advanced persistent threats (APTs) are constant attacks of specific targets by hackers using intelligent methods. All current internal infrastructures are constantly subject to APT attacks created by external and unknown malware. Therefore, information security officers require a framework that can assess whether information security systems are capable of detecting and blocking APT attacks. Furthermore, an on-line evaluation of information security systems is required to cope with various malicious code attacks. A regular evaluation of the information security system is thus essential. In this paper, we propose a dynamic updated evaluation framework to improve the detection rate of internal information systems for malware that is unknown to most (over 60 %) existing static information security system evaluation methodologies using non-updated unknown malware.

***Keywords:*** APT Evaluation, APT detection, Intrusion detection, APT evaluation framework, Detection Performance

**http://doi.org/10.3837/tiis.2019.05.021** **ISSN : 1976-7277**

## 1. Introduction

Most recent hacking attacks use Advance Persistent Threat (APT) with unknown vulnerabilities.[3],[22] In addition, the changes and evolution of the attacks are very fast. For this reason, the kind of framework to constantly evaluate the latest vulnerabilities and variant malicious codes is needed, rather than to verify detection capability by evaluating(identifying) the APT attack identification of information security system against past attacks.. That is, it is assumed that the requirement to verify the real function as the information security system in a real environment is premised, rather than receiving a license or a certificate. According to the study of the recent trends in hacking attacks, Jeong et al. stated that more than 90% of total APT attacks use malicious codes from e-mails. [10] Aditya K et al reported that the malicious code used here is created only for targets and target sites, and it increases the success rate of attacks by using customized malware not known to anyone. [2] For custom malicious code, in their articles, W. Liu et al. insist that an action-based attack method is used, [31] while DS, Sohn et al. and Lee, K, Lim, J claim that self-generated malware tried for the first time only on the specific sites is often used by hackers. [15],[23]

Malicious code is classified into executable and non-executable code. Executable malicious code is executable on a specific operating system or multiple operating systems. [10],[11],[23] The code attacks with customized files only for a specific site using anti-debugging or packing techniques in order to avoid the detection by the information security systems such as a vaccine which detects and protects the function in these operating systems. [24] Non-executable malicious code is not a direct execution but an indirect execution method, which infects targets using a PDF or MS Word document embedded with malicious code when it is loaded into a document file , and it uses a compressed file, such as ZIP format to attack[7],[31]. This method is widely used by hackers as the attacks are increasingly targeting software widely used by country (such as certain compression formats and document editing programs that can not be detected by software vaccines from other countries but used only in its own country) and it has the disadvantage of having difficulty in developing general detection technology because it is in the localization aspect. [6]

In addition, Marco Cova et al. said that these attacks use malicious URLs via DBD (Driveby download).[12],[16],[25] Researches on detecting APT malicious code have been based on the method for detecting botnets. For instance, AsSadhana and Mourab proposed an efficient method to detect the periodic operation of botnet traffic [4]. Dietrich et al. recognized the command channels of botnets, such as differences in transmission protocol, message length and sequence, and encoding differences of less than 50% within the network traffic and reduced the false positive rate by 0% to 1.93%.[5] Stevanovic et al. examined the efficient detection of botnets by analyzing three main channels for C&C (Command and Control) and traffic attacks.[18] Recently, studies have also been carried out on APT attack detection using machine learning.[28]

However, in the meantime, enterprises and government have introduced information security systems that have been verified and certified as information protection systems in specific certification authories (KISA, TTA etc.) or companies (NSS, Tolly etc.) to detect and

defend hacking attacks in general. The latest set of updated APT attacks listed above, represent a significant challenge to the existing information security systems in companies and organizations. Therefore, it is urgent to establish evaluation criteria for determining whether information security systems can clearly detect and block such attacks by reflecting the changed trend of the attack. [13],[19],[26],[27]. In this paper, we verify the existing evaluation criteria and environments and propose an evaluation framework that can continuously detect the newly changed APT attacks. [8],[14],[17],[32],[33]

This paper is composed as follows. Section 2 analyzes the performance evaluation models of existing information security systems and formulates the research questions. Section 3 proposes an on-line evaluation framework based on shared platforms proposed in this paper. Section 4 presents the test results for the system development and commercial malicious codes. Finally, section 5 summarizes the test results and describes the directions of future work.

## 2. Related Work

There are various methodologies of testing information security system to test the detection or vulnerability of external hacking attacks through a network. Nevertheless, evaluation of the information security systems has been accomplished within the detection range of well-known cyber-attacks and network performance.

The existing research institutes having methodologies to test information security systems include the TTA lab [21], which determines only Pass and Fail based on the required items by the certificate requester for the detection and performance evaluation, the Tolly lab [29], which measures the accuracy of malicious traffic detection and blocking, and the performance of the defenses against bypass attacks, and the Veritest lab, which does not have a specific evaluation criteria for APT but has evaluation items for measuring network virus anti-virus products, such as external file scan, saved file scan, compressed file scan, DoS viruses, Window viruses, file viruses, scripts, variant codes, virus generators, etc., using known malicious codes as detection performance measurement samples. These tests do not have the APT detection capability verification system itself or passively verify only the contents requested by the test examinees. Although, APT-related tests are possible in KISA lab [1], [21], ICSA lab [9] and NSS Lab [20] with a more advanced information protection system evaluation framework as shown in **Fig. 1**, in the closed network test based on off-line which is one time based on the measurement point, it is impossible to manage the continuous detection capability of a new or changed attack in the future due to limitation of physical and human resources including the system and frame work. That is, it is impossible to test vulnerabilities against unknown APT attacks.

In addition, some labs support the on-line evaluation framework as shown in **Fig. 2**, but there is a high probability of failure of the test due to the environment in which the APT attack should be waited for a long time. Even if an APT attack occurs, testers cannot sense undetected APT attack.

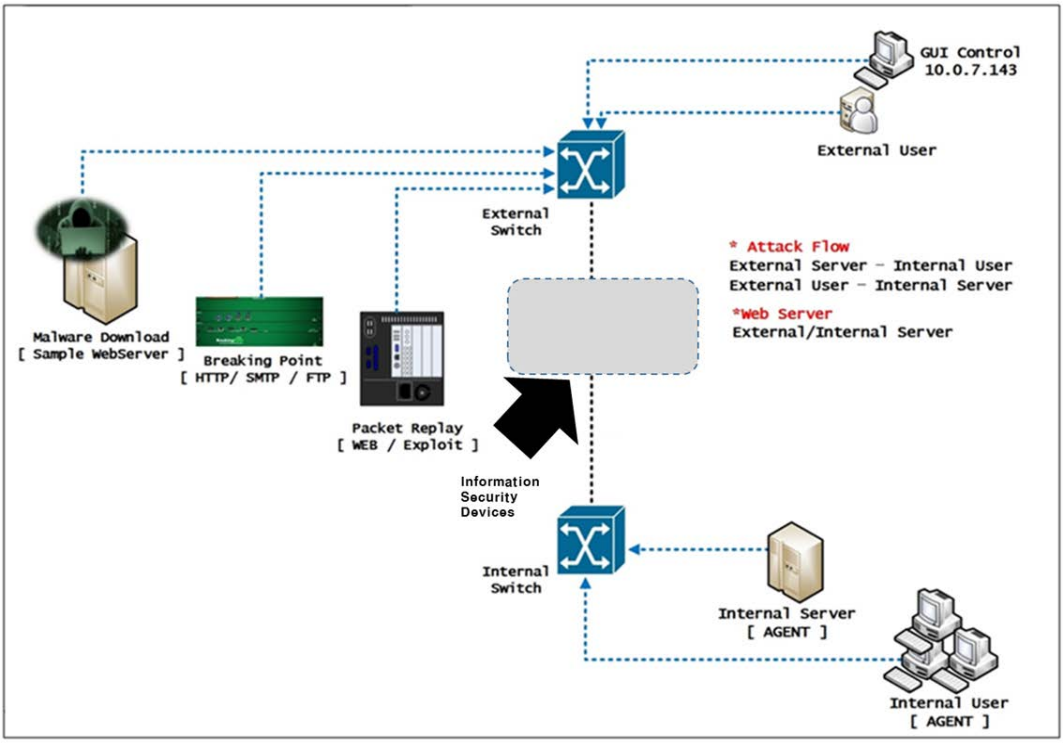


**Fig. 1.** Off-line Existing Evaluation Framework overview.

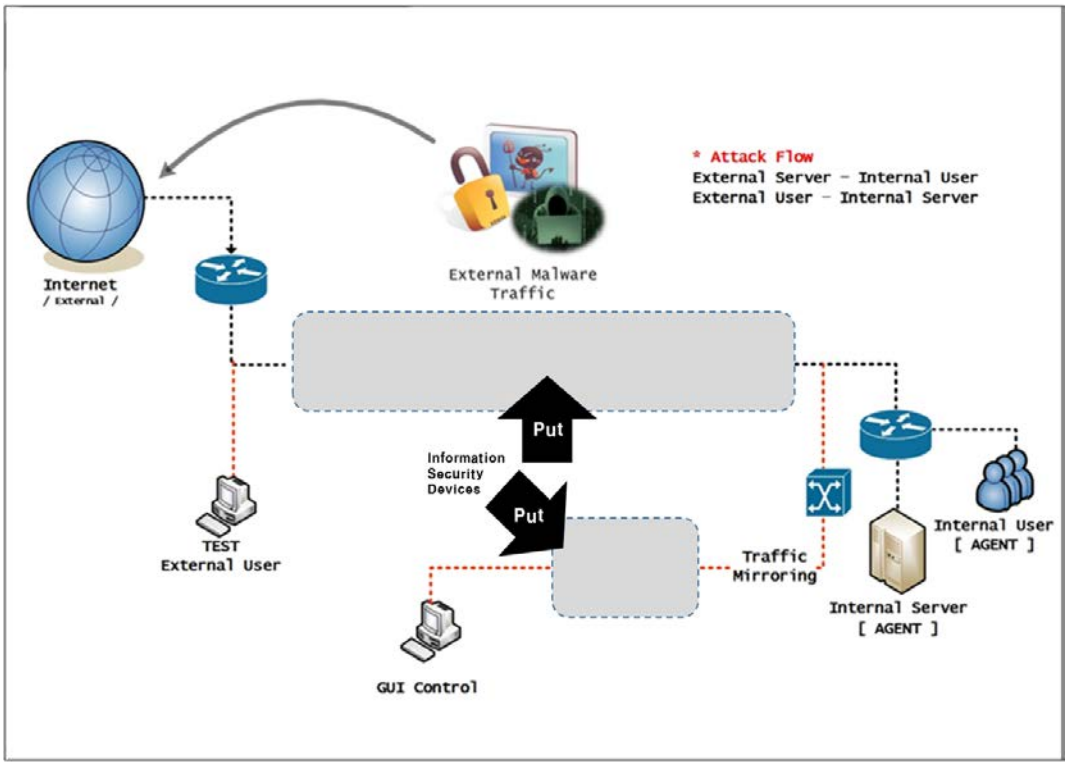


**Fig. 2.** On-line Existing Evaluation Framework overview

Therefore, a detection validity inventory on APT attacks for information security systems should be implemented to be able to test information security assets at all times by sharing malicious codes spread by different environments in a distributed network and performing attack replay with the detection abilities of the malicious code updated in real time. **Table 1** shows the APT attack test items for each of the information security system testing organizations. There is no environment or evaluation framework to satisfy all conditions and always test it. In addition, APT attacks that are not even detectable in each test set are being generated at this moment. This paper proposes an online evaluation framework in order to detect unknown attacks in each detection category.

**Table 1.** APT detection comparison for international testing laboratories

| APT Attacks | TTA | KISA | Tolly | ICSA | Veritest | NSS |
|---|---|---|---|---|---|---|
| E-mail APT | X | O | X | O | X | O |
| DBD APT | X | X | X | O | X | O |
| Messenger APT | X | X | X | O | X | O |
| Zeroday | X | O | X | X | X | △ |
| Executable code | X | O | X | O | O | O |
| Non-executable code | X | X | X | X | O | O |
| Customized code | X | X | X | X | X | X |

## 3. Proposed Scheme

### 3.1 Framework Overview

Most evaluation models of information security systems mentioned above, focus on functional requirements and packet processing capabilities, mainly using well-known attacks and malicious code samples collected in the past for intrusion detection capability testing. Although these test methodologies are not meaningless, in case of customized malicious code used for APT attack, reuse frequency is small or there is almost no similarity, moreover, the rebirth cycle of a new variant code is very short. Thus, a regular evaluation of information security systems is very important.

In this paper, we propose a framework to collect real-time traffic for the detection of APT attacks, identify APT attack source based on the collected, determine whether attacks are malicious through static analysis database and dynamic analysis sandbox, and in the case of malicious attacks, we generate virtual traffic only for the internal information protection system and always verify it (see **Fig. 3**). And then the results are linked to malicious code sharing with the the environment using the same framework, and support updating of evaluation models. In addition, in the actual network, internal information security systems are configured between the Attack Replayer and the Reflector in **Fig. 3** to prevent additional problems caused by the transition of traffic attacks to the inside.

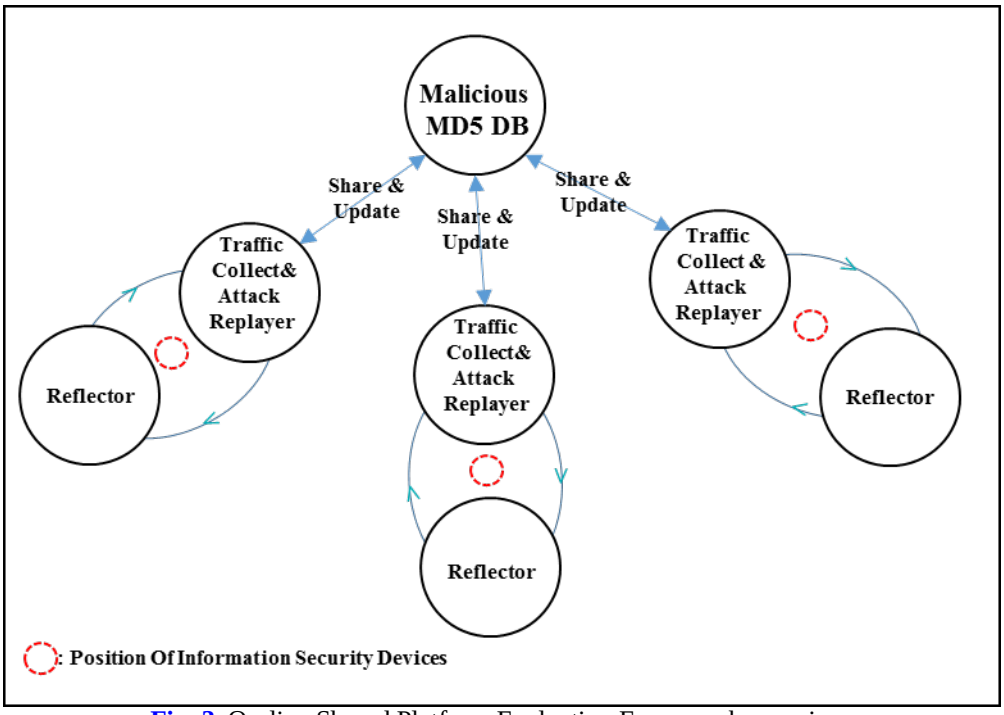


**Fig. 3.** On-line Shared Platform Evaluation Framework overview.

As shown in **Fig. 4**, the overall flow and steps to measure APT detection in Shared Platform Based On-line Evaluation Framework are as follows: 3.1 collection of link information after collecting and normalizing network traffic; 3.2 collection of emails; 3.3 file combination; 3.4 file collection and MD5 extraction; 3.5 MD5 comparison static analysis; 3.6 sandbox dynamic analysis; 3.7 APT attack replay and detection test. **Table 2** provides a general description of the functions and modules for configuring the processing flow of traffic collected through the network interface.

**Table 2.** The List of Function For Shared Platform Based On-line Evaluation Framework

| |
|---|
| MAIN: Main pseudo code for branching to each module in Collected Traffic |
| LINK_CHK: Link information collection pseudo code |
| EMAIL_CHK: E-mail collection pseudo code |
| FILE_CHK: Binary file analysis pseudo code |
| COMBINE_FILE: File combination pseudo code |
| MD5DB: Malicous code MD5 match pseudo code |
| DA: Dynamic analysis pseudo code |
| AR: Attack replayer pseudo code |
| K : Data buffer generated in traffic |
| Dk: DATA |

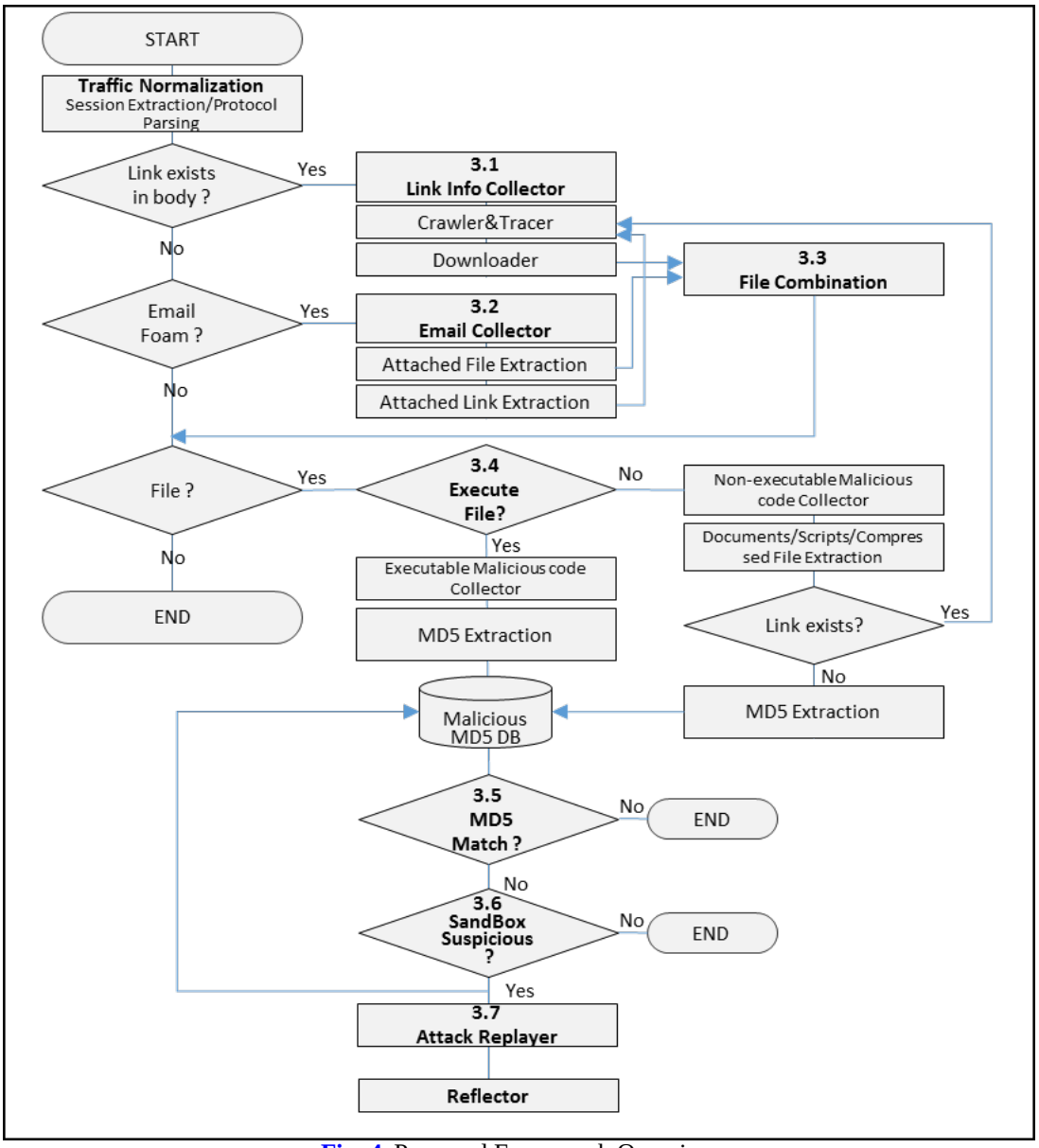


**Fig. 4.** Proposed Framework Overview

**Table 3** shows the main function for collecting, analyzing, and replaying APT attacks in Shared Platform Based On-line Evaluation Framework. This function calls up the data buffer created by traffic through the network interface and branches to each collection function up to 3-8 times and ends.

**Table 3.** The Main Function For Shared Platform Based On-line Evaluation Framework

```
pseudo code MAIN()
1. for k = 1 to K do
2.          Get Dk
3.          if Dk contain Link then
4.                    move to LINK_CHK();
5.          else if Dk contain Email then
6.                    move to EMAIL_CHK();
7.          else if Dk contain File then
8.                    move to FILE_CHK();
9.          else
10                   end{if}
11. end{for}
```

The collection of malicious codes in Shared Platform Based On-line Evaluation Framework is the key element in recognizing unknown APT attacks. It identifies whether codes are malicious by examining malicious links, emails, and files in the normalized traffic.

The first step 3.1 link information collection(**Fig. 4**) is the section for (2) crawling and tracing (1) when there is a link information in the data body branched from the buffer and (3) after downloading the file, (4) branching to the file combination function. When tracking the link information, it is necessary to re-extract the link from the webpage used as URL. In this case, Crawling and Tracer are used to trace the last link after the URI combination, and, if the file exists, the executable or non-executable file is downloaded using the downloader (see **Table 4**).

**Table 4.** Link Info Collector

```
pseudo code LINK_CHK()
1. foreach data ≠ empty do
2.          res1 <- Crawler and Tracer;
3.          res2 <- Downloader(res1);
4.     COMBINE_FILE(res2);
```

**Table 5** is 3.2 E-mail collection part, which extracts (2) attachment files and links if there is an email form in the data body branched from the buffer. (3) If the link exits, (4 )it branches to the LINK_CHK() function, and, if not, it branches (6) to the file combination function.

**Table 5.** Email Collector

```
pseudo code EMAIL_CHK()
1. foreach data ≠ empty do
2.          res <- Extract Attachment
3.     if res contain Link then
4.              move to LINK_CHK();
5.     else
6.          COMBINE_FILE(res);
```

**Table 6** shows 3.3 file combination step, which combines files as the result of branching from the e-mail collection function and the link information collection function, and branches to the binary file analysis function.

**Table 6.** File Combination.

```
pseudo code COMBINE_FILE()
1. foreach data ≠ empty do
2.        res <- combine files;
3.        FILE_CHK(res);
```

**Table 7** shows 3.4 file collection and MD5 extraction step. If the file extracted from the buffered data is executable, (2) extracts MD5 from the executable malicious code collector and (4) branches to the MD5DB function. If not (6), after extracting the script, the document file and the compressed file from the non-executable malicious code collector, (7) they are checked for links. If the link exists, (8) branches to the LINK_CHK() function; otherwise, after extracting MD5 (10) branches to the MD5DB function.

**Table 7.** File Collector & MD5 Extraction

```
pseudo code FILE_CHK()
1. foreach data ≠ empty do
2.        if a file is binary then
3.                  res <- Extract MD5;
4.                  move to MD5DB(res);
5.        else
6.                  res <- Extract Script, Document, Compressed file;
7.          if res contain Link then
8.                move to LINK_CHK();
9.          else
10.               res_md5 <- Extract MD5, res
10.               move to MD5DB(res_md5);
11.         end{if}
12.       end{if}
```

**Table 8** shows 3.5 MD5 comparison static analysis step. If the MD5 extracted from the binary file analysis function as the executable and non-malicious malicious code, matches the Malware MD5, (2) the process ends. Otherwise, (4) it branches to the sandbox dynamic analysis function, and determines whether it is malicious or not.

**Table 8.** MD5 Comparison Static Analysis

```
pseudo code MD5DB()
1. if a MD5 matching Malware Database
2.        end{if}
3. else
4.        DA(res);
5. end{if}
```

**Table 9** shows the sandbox dynamic analysis stage of the file that has not been analyzed through the static analysis step. MD5, which has not been identified as a malicious code in the MD5DB function, is determined whether it is malicious through the sandbox dynamic analyzer. If it is malicious, (3) it branches to the attack regenerator function; otherwise, (5) the process ends.

**Table 9.** Sandbox Dynamic Analysis

```
pseudo code DA()
1. foreach file ≠ empty do
2.          if file is Malware then
3.                    AR();
4.          else
5.                    end{if}
6.      end{if}
```

If a file is identified as malicious through the analysis process described above, the virtual traffic is simulated in the regenerator and reflector section to determine whether the information security system is detected without affecting the internal network of the actual environment in use. **Table 10** reproduces the APT attack against the test environment with the internal information security system if it is determined to be malicious in the sandbox dynamic analysis function.

**Table 10.** Attack Replayer and Reflector

```
pseudo code AR()
Description.
1. foreach file ≠ empty do
2.          firewall_chk;
3.          ips_chk;
4.          waf_chk;
5.          etc_chk;
6.      return Reflector;
```

## 4. Experimental Results

### 4.1 Experimental Environment

The core of this study is to propose countermeasures against hackers' unknown attacks in advance because if a single attack is successful, there will be serious information leakage, data corruption, and system failure on the victim's internal systems given the nature of the APT attack. In other words, our specific goal is to build a database using the proposed Evaluation Framework by analyzing APT attacks on its own, and to set up a preventive system to conduct simulated attacks through the built database. This is not just a one-time verification of the information security systems they will introduce simply through the existing test labs (NSS, Tolly, ICSA etc.) **[9]**,**[20]**,**[29]**, but it is to prove the effectiveness of the real-time APT advance response of information security systems through the constant Evaluation Framework.

The offline and online environment configurations of the constant evaluation framework for attacking and detecting APT are as shown in **Fig. 3** and **Fig. 4**. The difference between the two frameworks is the presence of the mutual updating function through the online Malicious MD5 DB. Information security systems, such as firewalls, intrusion prevention systems (IPS), and the APT prevention system, which are commonly tested for APT detection are installed in a series or in parallel in the gray boxes.

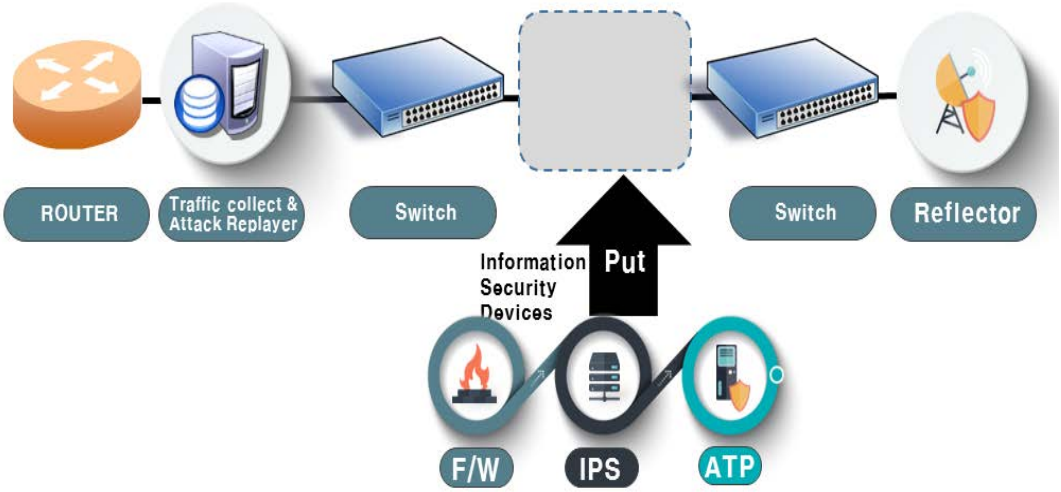


**Fig. 5.** Proposed Offline Evaluation Framework

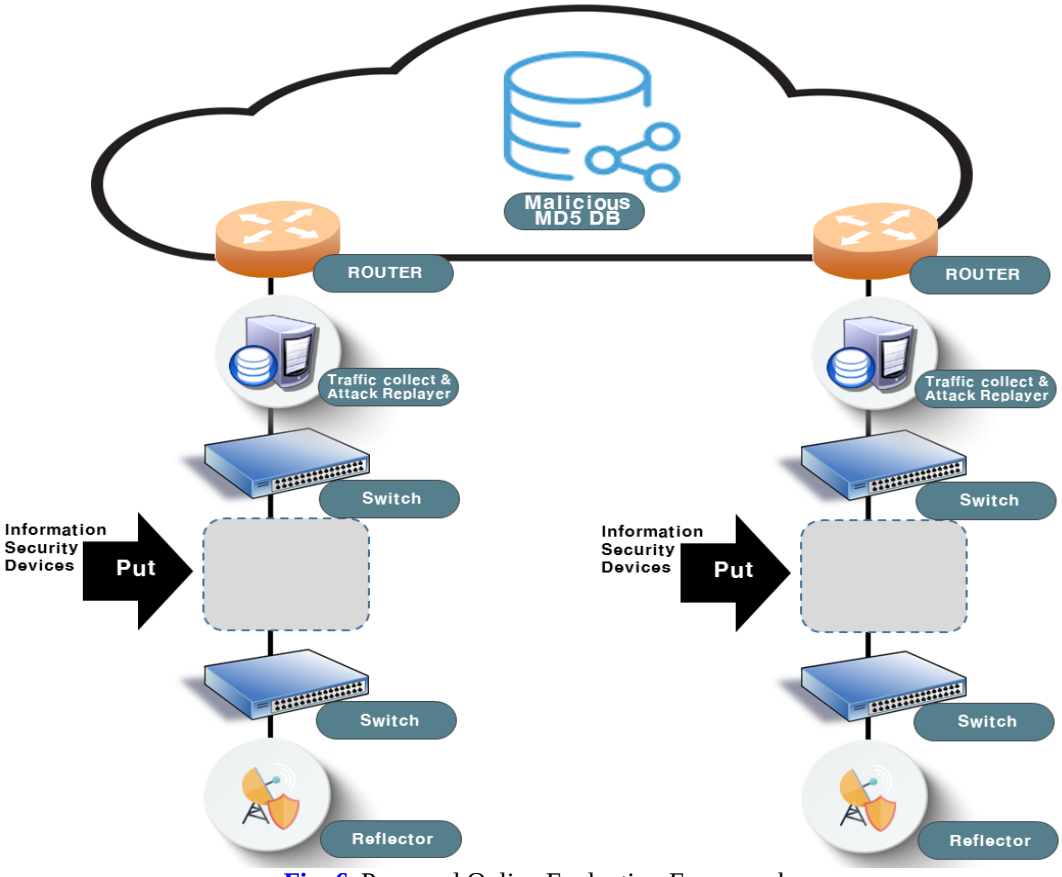


**Fig. 6.** Proposed Online Evaluation Framework

The information security systems for measuring the detection when the Evaluation Framework replays APT attacks are shown in **Table 11**. All test devices are equipped with the attack detection function of application layer.

**Table 11.** Information Security Devices for Testing

| | Name | A mounts of Signature | Specification |
|---|---|---|---|
| **F/W** | Sniper AF | 4000+∝ | NextGenerationFW(FW+VPN+IPS Modules) |
| **IPS** | Sniper IPS | 3500+∝ | Intrusion Prevention System |
| **ATP** | Sniper APTX | Flexible | Sandbox based Anti-APT System |

## 4.1 Experimental Results

This chapter proves the efficiency of the APT advance response of the proposed Evaluation Framework. The definition of proactive response is that the proposed framework detects the APT attacks in traffic flowing in the network, builds a database in the framework, based on this, it conducts simulated attacks on the information protection systems existing in the framework, and constantly checks for APT detection. To this end, **[30]** traffic including malicious code was collected from the commercial environment and Virustotal site from 2015 to 2017. The collected malicious codes were classified as well-known attacks if they were detected by an antivirus software and as APT attacks if they were not. A total of 3,000 samples of network traffic were divided into seven categories according to the attack characteristics. In order to test as much as in the actual environment, the traffic including well-known attacks was mixed with the traffic including APT attacks and used within the Evaluation Framework to attack. Seven APT attack categories and the distribution of malicious code samples by year are shown in **Table 12**.

**Table 12.** Malware sample distribution by year and Attack category

| Category for APT Attacks | | 2015 | 2016 | 2017 |
|---|---|---|---|---|
| 1.E-mail APT | Well-known | 140 | 140 | 140 |
| | APT | 10 | 10 | 10 |
| 2.DBD APT | Well-known | 140 | 140 | 140 |
| | APT | 10 | 10 | 10 |
| 3.Messenger APT | Well-known | 140 | 140 | 140 |
| | APT | 10 | 10 | 10 |
| 4.Zeroday | Well-known | 40 | 40 | 40 |
| | APT | 10 | 10 | 10 |
| 5.Executable code | Well-known | 270 | 270 | 270 |
| | APT | 30 | 30 | 30 |
| 6.Non-executable code | Well-known | 90 | 90 | 90 |
| | APT | 10 | 10 | 10 |
| 7.Customized code | Well-known | 90 | 90 | 90 |
| | APT | 10 | 10 | 10 |
| **Sum** | | 1000 | 1000 | 1000 |

In order to test the effectiveness of proactive response of the proposed Evaluation Framework, A and B in **Fig. 7** are connected online, and C is configured offline. And then, the 3,000 types of malicious network traffics were passed through to discriminate the traffic where Evaluation Framework is flowing and test whether the framework could build a database with APT attacks. The attack is in the following order: A-> B-> C network, and was performed only once due to the nature of APT attacks. If more than two tests are performed, the proactive response effectiveness of the proposed system will affect reliability by the Self-Learning function of the proposed system itself.

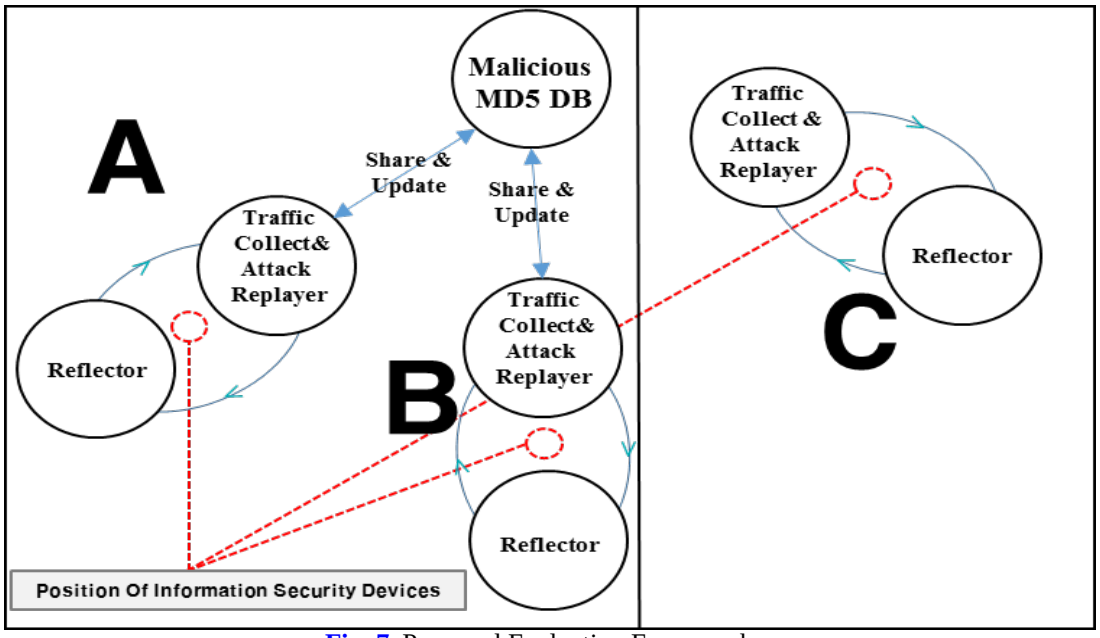


**Fig. 7.** Proposed Evaluation Framework

**Table 13.** Detection Rate of Proposed Online Vs Offline Evaluation Framework

| Category for APT Attacks | | Online | | | | | | Offline | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | A | | | B | | | C | | |
| | | '15 | '16 | '17 | '15 | '16 | '17 | '15 | '16 | '17 |
| 1.E-mail APT | Well-known | 88% | 94% | 82% | 100% | 100% | 100% | 88% | 94% | 82% |
| | APT | 92% | 90% | 90% | 100% | 100% | 100% | 92% | 90% | 90% |
| 2.DBD APT | Well-known | 92% | 92% | 92% | 100% | 100% | 100% | 92% | 92% | 92% |
| | APT | 90% | 94% | 92% | 100% | 100% | 100% | 90% | 94% | 92% |
| 3.Messenger APT | Well-known | 87% | 94% | 84% | 100% | 100% | 100% | 87% | 94% | 84% |
| | APT | 88% | 96% | 84% | 100% | 100% | 100% | 88% | 96% | 84% |
| 4.Zeroday | Well-known | 74% | 98% | 80% | 100% | 100% | 100% | 74% | 98% | 80% |
| | APT | 76% | 74% | 74% | 100% | 100% | 100% | 76% | 74% | 74% |
| 5.Executable code | Well-known | 88% | 92% | 80% | 100% | 100% | 100% | 88% | 92% | 80% |
| | APT | 94% | 98% | 84% | 100% | 100% | 100% | 94% | 98% | 84% |
| 6.Non-executable code | Well-known | 89% | 92% | 82% | 100% | 100% | 100% | 89% | 92% | 82% |
| | APT | 96% | 96% | 96% | 100% | 100% | 100% | 96% | 96% | 96% |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 7.Customized code | Well-known | 98% | 96% | 88% | 100% | 100% | 100% | 98% | 96% | 88% |
| | APT | 94% | 96% | 96% | 100% | 100% | 100% | 94% | 96% | 96% |
| **Sum** | Well-known | 88% | 94% | 84% | 100% | 100% | 100% | 88% | 94% | 84% |
| | APT | 90% | 92% | 88% | 100% | 100% | 100% | 90% | 92% | 88% |

As shown in **Table 13**, in the case of E-mail APT attacks, which is the mainstream of APT attacks, **[10]** it was possible to detect 92% of the attacks in 2015 for A configuration, 90% in 2016, and 90% in 2017. However, for B configuration connected online, 100% detection rate was possible by updating according to the detection results of A configuration.

The comparison of these figures states that in the evaluation framework A, the average 91% of E-mail APT attacks can be simulated on internal information security systems; 9% of actual hacker attacks using E-mail cannot be performed as an audit function for the detection of the internal information security systems. For the case of APT attacks, as described in the previous section, a single successful attack causes problems, such as serious information leakage, data corruption, and system failure on the victim's internal system. The administrators should assume 9% risk rate of the attack success.

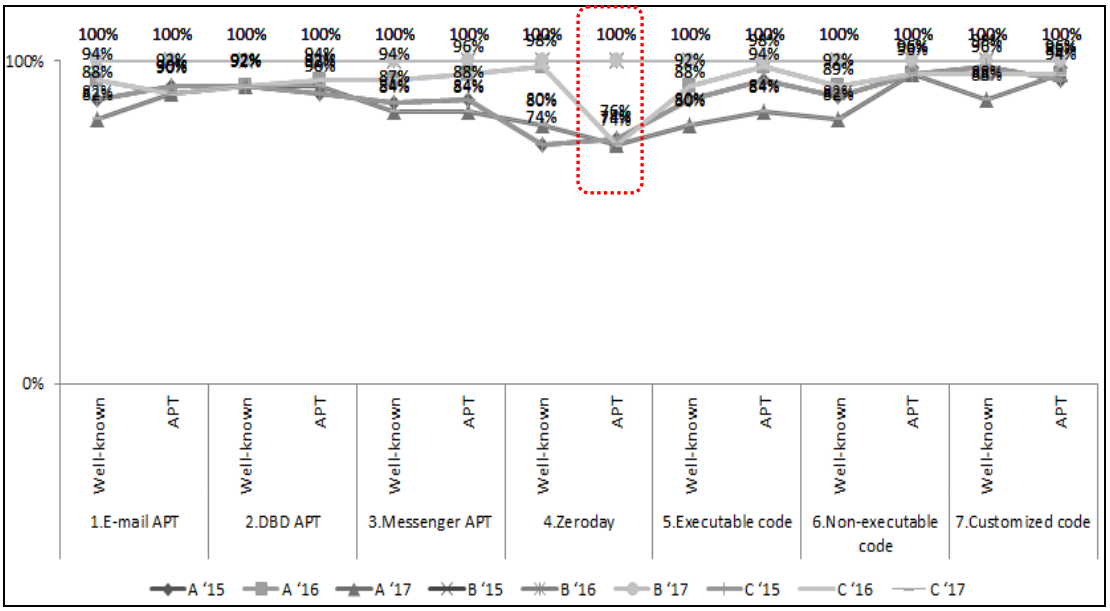


**Fig. 8.** Detection rate for the APT attack category

Furthermore, as shown in **Fig. 8**, Zeroday APT attacks,which shows the lowest detection rate, can be detected by 76% from 2015, 74% from 2016, and 74% from 2017. This means the administrators should take an attack succes risk of more than 25%. This implies that one of every four hacking attacks will be absolutely successful. This is fatal for internal systems.

By contrast, all Zeroday APT attacks were 100% detectable for B configuration connected online. That is, the effectiveness of proactive response increased perfectly by 25%, from 75% to 100% through the proposed Evaluation Framework. This demonstrates that the effectiveness of proactive response is perfect, as it does not allow even 1% of Zeroday APT attacks. The results of the analysis for configurations A and C are the same, and that is because they did not receive the information update.

Although A configuration was configured online, it was attacked first according to the test environment. Thus, it can be considered the same as C configuration. This means that for APT attacks, the proposed Evaluation Framework is also very effective in patching information security systems only with the updates and tests through an online shared platform.

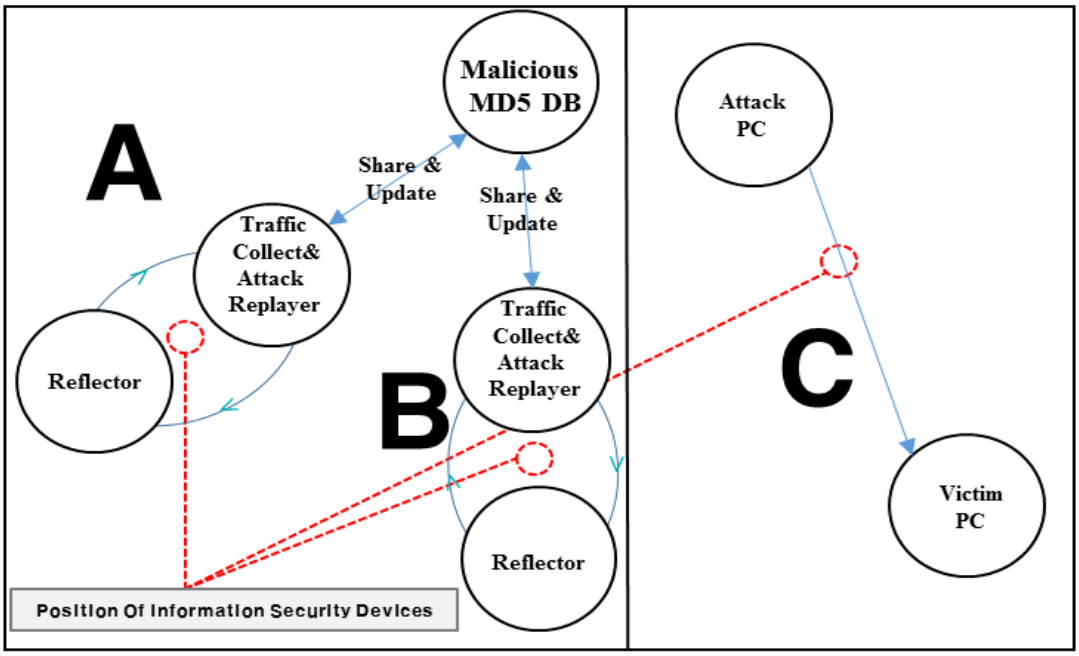


**Fig. 9.** Proposed evaluation framework Vs Existing evaluation framework

**Table 14.** Detection rate of the proposed online vs. existing evaluation framework

| Category for APT Attacks | | Proposed | | | | | | Existing(NSS,KISA) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | A | | | B | | | C | | |
| | | '15 | '16 | '17 | '15 | '16 | '17 | '15 | '16 | '17 |
| 1.E-mail APT | Well-known | 88% | 94% | 82% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 92% | 90% | 90% | 100% | 100% | 100% | 0% | 0% | 0% |
| 2.DBD APT | Well-known | 92% | 92% | 92% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 90% | 94% | 92% | 100% | 100% | 100% | 0% | 0% | 0% |
| 3.Messenger APT | Well-known | 87% | 94% | 84% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 88% | 96% | 84% | 100% | 100% | 100% | 0% | 0% | 0% |
| 4.Zeroday | Well-known | 74% | 98% | 80% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 76% | 74% | 74% | 100% | 100% | 100% | 0% | 0% | 0% |
| 5.Executable code | Well-known | 88% | 92% | 80% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 94% | 98% | 84% | 100% | 100% | 100% | 0% | 0% | 0% |
| 6.Non-executable code | Well-known | 89% | 92% | 82% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 96% | 96% | 96% | 100% | 100% | 100% | 0% | 0% | 0% |
| 7.Customized code | Well-known | 98% | 96% | 88% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 94% | 96% | 96% | 100% | 100% | 100% | 0% | 0% | 0% |
| **Sum** | Well-known | 88% | 94% | 84% | 100% | 100% | 100% | 100% | 100% | 100% |
| | APT | 90% | 92% | 88% | 100% | 100% | 100% | 0% | 0% | 0% |

In order to compare the advance response effectiveness of the the existing Evaluation Framework used in proposed Evaluation Framework, NSS lab or KISA, A and B are connected online, and C is configured offline as shown in **Fig. 9**. After that, flowing traffic was discriminated and tested to determine if it is possible to build database with APT attack. The existing evaluation framework does not have the function to analyze traffic, so direct comparison is difficult. Thus, the attack was manually forced to be performed to compare risk management figures. Well-known attacks are 100% detectable in both the proposed evaluation framework B and the existing evaluation framework C as shown in **Table 14**. However, since there is no APT identification function in C, 0% for the detection is shown. When comparing the attack results of B and those of C, 100% for the known attacks of B is auto-identified, and 100% for the known attacks of C is arbitrarily identified for testing. As such, this study has greatly contributed to identify unknown attacks related to APT and support risk management compared to the existing assessment frame work, like detection differences in **Fig 10**.

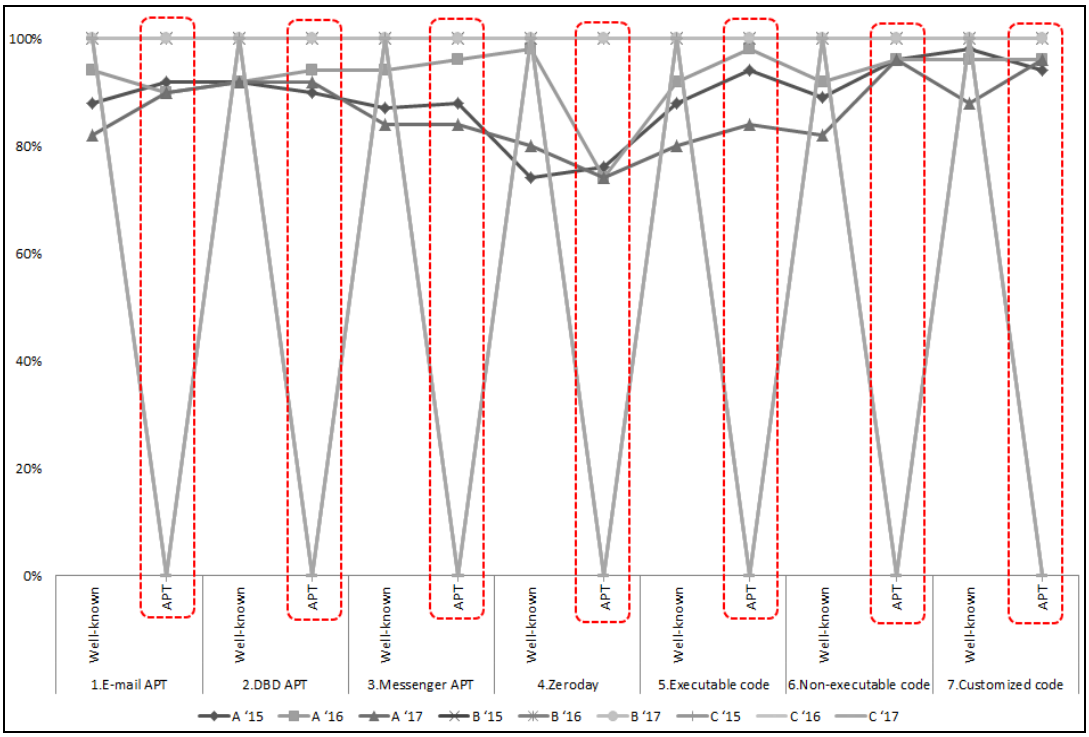


**Fig. 10.** Detection rate for Proposed evaluation framework vs. Existing evaluation framework

## 5. Contribution

In this paper, we use a dynamic on-line evaluation framework and try to give contributions in improving the detection rate of information security systems against more than 60 % of unknown hacking attacks and malwares compared to the static evaluation methodologies using the existing non-updated vulnerability information. Actual malicious code traffic was used to verify the validity of this model, and the accuracy of the demonstration was increased by dividing attack types into seven groups.

This research greatly contributes two things to BCP (Business continuity planning) for the survival of enterprises and organizations. First, it supports risk management for enterprise

survival by suggesting the measures to manage the efficiency of proactive response to external APT hacking attacks and presenting the threat index for various types of APT attacks to management.

Second, it can be a tool to audit information protection capabilities of the information security systems that are planned to be purchased or already purchased. It presents administrators or management with indicators to assess the efficiency and sustainability of investment in expensive information security systems. This means that this study replaces periodic auditing or supervisory frameworks to check whether proper information protection measures are in place to ensure that businesses continue their business.

## 6. Conclusions

A number of cyber-infringement accidents is constantly occurring nowadays. Given the nature of APT attacks, the success of just one attack causes problems, such as serious information leakage, data corruption, and system failure on the victim's internal system. Therefore, we propose the effectiveness of proactive responses to prevent unknown attacks by hackers in advance. In this paper, a one-time and passive malicious code evaluation test framework, which is specialized in APT detection tests and used in the existing research institutes, was improved and a regular evaluation framework was proposed. The results of testing 3,000 malicious codes demonstrated that the proposed model greatly enhanced the reliability level and increased the effectiveness of proactive response compared to the existing one.

Among the malicious codes appearing at the rate of 1 million per day around the world, APT attacks are mainly unknown attacks. Therefore, it is extremely difficult to identify whether a specific packet is related to APT attacks and most packets are not detected. Still, the proposed model shares the packet, and enables a quick identification of unknown attacks, which can be an important key to the quick response to APT attacks. Due to this, the proposed framework can improve the detection capability and security of network security devices.

## Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2017R1E1A1A01075110)

## References

[1] Andreas Dewald, Thorsten Holz, Felix C. Freiling, "ADSandbox: sandboxing JavaScript to fight malicious websites," in *Proc. of SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing* , ACM, pp.1859-1864, Mar 2010. Article (CrossRef Link).

[2] Aditya K. Sood and Richard J. Enbody, "Targeted Cyberattacks: A Superset of Advanced Persistent Threats," *IEEE Security and Privacy*, vol. 11, issue 1, pp. 54-61, Jan.-Feb., 2013. Article (CrossRef Link).

[3] Bhavik Thakar and Ahmedabad, "Advance Persistent Threat: Botnet," in *Proc. of ICTCS '16 Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, NY,* ACM, Article No. 143, 2016. Article (CrossRef Link).

[4] B. AsSadhana, Jose M.F. Mourab, "An efficient method to detect periodic behavior in botnet traffic by analyzing control plane traffic," *Journal of Advanced Research*, vol. 5, no. 4, pp. 435 – 448, 2014. Article (CrossRef Link).

[5] C. J. Dietrich, C. Rossow, and N. Pohlmann, "Cocospot: Clustering and recognizing botnet command and control channels using traffic analysis," *Computer Networks*, vol. 57, no. 2, pp. 475 – 486, 2013. Article (CrossRef Link).
[6] Dhruwajita Devi and Sukumar Nandi, "Detection of packed malware," in *Proc. of SecurIT '12: Proceedings of the First International Conference on Security of Internet of Things*, ACM, pp. 22-26, August 2012. Article (CrossRef Link).
[7] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effecti ve feature selection in machine learning based botnet detection approaches," *CNS14*, IEEE, pp. 247-255, Oct 2014. Article (CrossRef Link).
[8] Galal Hisham S, Mahdy YB, Atiea MA, "Behavior-based features model for malware detection," *Journal of Computer Virology and Hacking Techniques*, Vol. 12, Issue. 2, pp. 59–67, May 2016. Article (CrossRef Link).
[9] Testing Requirement of ICSA Lab, http://www.icsalabs.com, 2018. Article (CrossRef Link).
[10] Jeong, H., Kim, H. K., Lee, S., and Kim, E., "Detection of Zombie PCs Based on Email Spam Analysis," *KSII Transactions on Internet and Information Systems*, Vol. 6, No.5, May 2012. Article (CrossRef Link).
[11] Hyoung Chun Kim, Young Han Choi, Dong Hoon Lee, "JsSandbox: A Framework for Analyzing the Behavior of Malicious JavaScript Code using Internal Function Hooking," *KSII Transactions on Internet and Information Systems*, Vol. 6, No.2, February 2012. Article (CrossRef Link).
[12] Konrad Rieck, Tammo Krueger, Andreas Dewald, "Cujo: Efficient detection and prevention of drive-by-download attacks," in *Proc. of ACSAC '10: Proceedings of the 26th Annual Computer Security Applications Conference*, ACM, pp. 31-39, Dec 2010. Article (CrossRef Link).
[13] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big data analytics framework for peer-to-peer botnet detection using random forests," *Information Sciences*, vol. 278, no. Complete, pp. 488-497, 2014. Article (CrossRef Link).
[14] K. Shanthi and D. Seenivasan, "Detection of botnet by analyzing network traffic flow characteristics using open source tools," in *Proc. of ISCO15*, IEEE, pp.1-5, Jan 2015. Article (CrossRef Link).
[15] Lee, K. and Lim, J., "The Reality and Response of Cyber Threats to Critical Infrastructure: A Case Study of the Cyber-terror Attack on the Korea Hydro & Nuclear Power Co., Ltd.," *KSII Transactions on Internet and Information Systems*, Vol. 10, No. 2, 2016. Article (CrossRef Link).
[16] Marco Cova, Christopher Kruegel, and Giovanni Vigna, "Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code," in *Proc. of WWW '10 Proceedings of the 19th international conference on World wide web*, pp. 281-290, 2010. Article (CrossRef Link).
[17] Miao Q, Liu J, Cao Y et al., "Malware detection using bilayer behavior abstraction and improved one-class support vector machines," *International Journal of Information Security*, Vol. 15, Issue. 4, pp 361–379, August 2016. Article (CrossRef Link).
[18] M. Stevanovic and J. M. Pedersen, "An analysis of network traffic classification for botnet detection," in *Proc. of CyberSA15*, pp. 1-8, June 2015. Article (CrossRef Link).
[19] M. Sheikhan and Z. Jadidi, "Flow-based anomaly detection in high-speed links using modified gsa-optimized neural network," *Neural Computing and Applications*, vol. 24, no. 3-4, pp. 599-611, 2012. Article (CrossRef Link).
[20] NSS Labs, http://www.nsslabs.com, 2018.
[21] "Researching for evaluating on Information Security System," *KISA*, 2017.
[22] Ross Brewer, "Advanced persistent threats: minimising the damage," *Network Security*, vol. 2014, No. 4, pp. 5-9, 2014. Article (CrossRef Link).
[23] Sohn, DS., "The design on Circulation Detection &Blocking Architecture against Unknown Cyber Attack in IPS," *konkuk University*, 2015. Article (CrossRef Link).
[24] Sara Khanchi, Ali Vahdat, Malcolm I. Heywood, A. Nur Zincir-Heywood, "On botnet detection with genetic programming under streaming data, label budgets and class imbalance," in *Proc. of GECCO '18: Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ACM, pp. 21-22, July 2018. Article (CrossRef Link).

[25] S. Garcia, V. Uhlr, and M. Rehak, "Identifying and modeling botnet c&c behaviors," in *Proc. of the 1st International Workshop on Agents and CyberSecurity, ACySE'14*, ACM, pp. 1:1-1:8, 2014. Article (CrossRef Link).
[26] S.-C. Lin, P. S. Chen, and C.-C. Chang, "A novel method of mining network ow to detect p2p botnets," *Peer-to-Peer Networking and Applications*, vol. 7, no. 4, pp. 645 - 654, 2014. Article (CrossRef Link).
[27] S. Garca, M. Grill, J. Stiborek, A. Zunino, "An empirical comparison of botnet detection methods," *Computers and Security*, Vol. 45, pp. 100-123, 2014. Article (CrossRef Link).
[28] Sana Siddiqui, Muhammad Salman Khan, Ken Ferens, Witold Kinsner "Detecting Advanced Persistent Threats using Fractal Dimension based Machine Learning Classification," in *Proc. of IWSPA '16: Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics*, ACM, pp. 64-69, Mar 2016. Article (CrossRef Link).
[29] Tolly Group, http://www.tolly.com, 2018. Article (CrossRef Link).
[30] VirusTotal Site, https://www.virustotal.com, 2018. Article (CrossRef Link).
[31] W. Liu, P. Ren, K. Liu, and H.-X. Duan, "Behavior-based malware analysis and detection," in *Proc. of the 1st International Workshop on Complexity and Data Mining (IWCDM'11)*, IEEE, pp. 39–42, Sep 2011. Article (CrossRef Link).
[32] X. Liu, G. He, X. Wu, and D. Yu, "An abnormal network behavior detection system based on compound session," in *Proc. of 2014 Sixth International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 2, pp. 34-37, Aug 2014. Article (CrossRef Link).
[33] Youngjoon K, Eunjin K, Huy Kang K, "A novel approach to detect malware based on API call sequence analysis," *International Journal of Distributed Sensor Networks*, Vol. 2015, Jan 2015. Article (CrossRef Link).

**Dongsik Sohn** is a researcher at ISAA Lab., Department of Computer Engineering, Ajou University and SOC vice president of Wins corp, South Korea. He received the M.A. from Kunkuk University.
His research interests include APT attacks, malware & exploit analysis, network security, incident handing, network forensics and applied security mechanisms for Cloud and Big Data system and so on

**Taejin Lee** is a professor at Computer Engineering in Hoseo University. He received the Ph.D. from Ajou university. His research interests include malware analysis, network security and endpoint detection response.

**Jin Kwak** is a professor at Dept. Of Cyber Security in Ajou University, Korea. He received the Ph.D. degree from SKKU, Korea. His research interests include Cryptographic protocols, Applied security mechanisms for Cloud and Big Data system and so on.