KIPS Tr. Comp. and Comm. Sys.
Vol.5, No.11 pp.419~426 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2016.5.11.419

Automatic Creation of Forensic Indicators with Cuckoo Sandbox and Its Application

Kang Boong Gu[†] · Yoon Jong Seong[††] · Lee Min Wook[†††] · Lee Sang Jin[††††]

ABSTRACT

As the threat of cyber incident grows continuously, the need of IOC(Indicators of Compromise) is increasing to identify the cause of incidents and share it for quick response to similar incidents. But only few companies use it domestically and the research about the application of IOC is deficient compared to foreign countries. Therefore in this paper, a quick and standardized way to create IOC automatically based on the analysis result of malwares from Cuckoo Sandbox and its application is suggested.

Keywords : Incident Response, Indicators of Compromise(IOC), Digital Forensic, Cuckoo Sandbox

Cuckoo Sandbox를 이용한 포렌식 침해지표 자동생성 및 활용 방안

강 봉 구[†] · 윤 종 성[††] · 이 민 욱[†††] · 이 상 진[††††]

요 약

침해사고에 대한 위협이 지속적으로 증가하고 있는 가운데, 이에 대한 원인을 식별하고 해당 내용을 공유하여 유사한 침해사고에 대해 빠르게 대응하기 위한 침해지표(IOC, Indicators of Compromise)의 필요성이 증가하고 있다. 하지만, 국내의 경우 일부 업체에서만 이를 활용할 뿐 외국에 비해 침해지표의 활용 방안에 대한 연구가 많이 부족한 상황이다. 본 논문에서는 Cuckoo Sandbox의 악성코드 분석 결과를 바탕으로 빠르고 표준화된 침해지표 자동생성 방법과 이에 대한 활용 방안을 제안한다.

키워드 : 침해사고 분석, 침해지표, 디지털 포렌식, 쿠쿠 샌드박스

1. 서 론

최근 침해사고의 가장 대표적인 공격방법으로 APT 공격이 있으며, 제로데이 취약점을 활용한 APT 공격은 과거의 시그니처를 통한 탐지만으로는 식별이 매우 어렵다. 이를 해결하기 위해 최근에는 시그니처뿐 아니라 악성코드의 목적과 기능을 바탕으로 생성된 침해지표를 활용한다[1]. 이를 통해 침해사고를 탐지하고 이에 대한 지표를 공유함으로써 유사한 침해사고에 대해서도 식별 및 대처를 할 수 있다. 하지만 국내의 경우 일부 업체에서 활용하고 있을 뿐 침해지표의 활용 방안에 대한 연구가 많이 부족하다. 이에 반해 외국은 대표적으로 Mandiant의 OpenIOC DB를 통해 침해사고 관련 침해지표를 공유하고 이에 대해 토론하는 문화가 정착되어 있다. 하지만 침해사고의 식별을 위한 침해지표의 경우 침해사고의 흔적을 토대로 분석가에 의해 수기로 작성되는데, 이러한 특징으로 인해 침해지표를 생성하는데 많은 시간이 소요될 뿐 아니라 침해지표의 신뢰성이 분석가의 작업 숙련도에 의존적인 상황이다. 허니 넷 프로젝트에서 실시한 실험 보고서에 따르면, 모의 시스템에서 30분간 침투 실험의 피해 내용 및 상황을 조사하는데 전문가 1인당 평균 48시간이 걸렸다고 한다[2]. 만약 실제 침해사고에서 결과 분석 후 침해지표를 작성했을 때는 더욱 많은 시간이 소요될 것으로 예상된다. 이에 본 논문에서는 Cuckoo Sandbox의 분석 결과를 바탕으로 빠르고 표준화된 침해지표를 생성하는 침해지표 자동생성 도구와 이에 대한 활용 방안을 제안한다.

† 준 회 원 : 고려대학교 정보보호대학원 정보보호학과 석사과정
†† 비 회 원 : 고려대학교 정보보호대학원 정보보호학과 석·박사통합과정
††† 준 회 원 : 고려대학교 정보보호대학원 정보보호학과 석사
†††† 종신회원 : 고려대학교 정보보호대학원 교수
Manuscript Received : June 22, 2016
First Revision : August 28, 2016
Accepted : August 31, 2016
* Corresponding Author : Lee Sang Jin(sangjin@korea.ac.kr)

2. 관련 연구

침해지표(IOC, Indicators of Compromise)는 침해사고에서 호스트 또는 네트워크에서 식별할 수 있는 침해사고의 포렌식 아티팩트를 하나의 문서 형태로 정리한 파일이다[3]. 파일 이름, 해쉬값, 레지스트리 키, IP 주소 등으로 이루어진 침해지표를 통해 식별된 위협정보를 공유하여 침해사고에 대한 빠른 대응 및 피해 확산 방지를 위해 사용된다. 침해지표는 기계와 사람이 공통으로 이해할 수 있는 정형화된 포맷인 XML 형식을 주로 사용하며, 이를 활용한 대표적인 침해지표로는 OpenIOC와 CybOX가 존재한다.

OpenIOC(Open Indicators of Compromise)는 Mandiant에서 사이버 위협대응 및 침해사고 조기발견을 위해 개발한 XML 기반의 침해지표 포맷이다[4]. OpenIOC는 침해사고 또는 악성코드 감염 후 시스템에 남는 흔적정보를 바탕으로 AND, OR 논리 조합의 형태로 연결하여 침해지표를 작성한다. 무료로 제공하는 IOC Editor 프로그램을 통해 사용자가 GUI 화면에서 직접 침해지표를 작성할 수 있다. 또한, Finder, Redline 이라는 도구를 통해 피해가 의심되는 시스템에서 정보를 수집하고 기존에 작성한 침해지표와 비교하는 기능도 제공한다. 다만, 시스템에 남는 흔적의 종류가 제한되어 있어 침해지표를 생성하는데 한계점을 가지고 있으며 수기식 작성으로 인해 많은 시간이 소모되고 관련 도구에 대한 전문적인 지식이 필요하다.

CybOX(Cyber Observable eXpression)는 MITRE에서 공개한 XML 기반의 침해지표 포맷이다[5]. CybOX는 시스템에서 관측 가능한 정보를 Object로 표현하여 기술하며, Object는 레지스트리, 파일, 네트워크 정보 등 침해시스템에서 발견되는 다양한 정보들을 기술한다. MITRE에서 CybOX를 기반으로 STIX, TAXII 등의 특화된 침해지표 프로젝트를 제공하고 있으며, 악성코드 분석 결과에 대한 표준언어로 MAEC(Malware Attribute Enumeration and Characterization)를 제공한다. MAEC는 Cuckoo Sandbox 0.4 버전에서 기능이 추가되어, 1.0버전부터 MAEC 4.0.1버전의 포맷으로 악성코드 분석 결과를 옵션형태로 제공하고 있다. 하지만 근본적으로 CybOX는 포렌식 조사를 위해 설계되지 않아 AMCache, Shellbag과 같은 중요 포렌식 아티팩트를 Object로 명확하게 구분하고 있지 않다. 또한, MAEC의 경우 악성코드의 행위를 상세하게 설명하기 위한 목적으로 작성되어 호스트 내의 침해사고를 식별하기 위한 지표로는 부적절하다. 더욱이 침해지표를 작성하거나 활용하기 위한 자동화 도구를 제공하지 않아 활용이 제한적이다.

DFIOC(Digital Forensic Indicator of Compromise)는 침해사고 발생 시스템의 다양한 포렌식 아티팩트를 하나의 XML 포맷으로 통합하여 표현한다[6]. DFIOC는 총 3가지 항목으로 구성되어 있으며 침해사고 시스템에서 수집된 여러 포렌식 아티팩트 정보를 Evidence 항목에 입력한다. 침해사고 분석 결과를 바탕으로 수집된 증거들의 연관 관계를 Forensic Analysis 항목에 추가하며, 침해흔적을 Indicator 항목으로 나타낸다. 다만, Indicator 항목 작성 시 수집된 대용량의 Evidence를 바탕으로 수기로 작성하므로 침해지표를 생성하는데 많은 시간이 소요되는 단점이 존재한다.

3. 침해지표 자동생성 방법

본 논문에서는 침해지표를 자동생성하기 위해 샌드박스 형태의 가상 환경을 이용하였다. 샌드박스란 컴퓨터에서 어떠한 프로그램 또는 코드를 실행할 수 있는 환경을 제공해주는 격리된 공간을 의미한다. 가상의 컴퓨터 환경으로 구성된 샌드박스 내부에서 악성 파일을 동작시킴으로써 실제 환경에는 영향을 끼치지 않으며 빠르고 안전하게 분석이 가능한 장점을 가진다. 침해지표 자동생성 도구는 오픈소스로 악성코드 동적분석에 많이 사용되는 Cuckoo Sandbox 환경에서 악성코드를 동작시킨 뒤 악성코드 분석결과를 바탕으로 침해지표를 자동으로 작성하여 출력한다.

3.1 Cuckoo Sandbox 분석 결과 출력

Cuckoo Sandbox는 악성코드 분석 시스템으로써 사용자가 의심스러운 파일을 분석하기 위해 격리된 가상 환경에서 의심 파일에 대한 상세 분석결과를 제공한다[7]. 이를 통해 의심스러운 파일의 고유 행위만을 파악하는데 유용하다. Cuckoo Sandbox를 통한 악성코드의 분석결과는 크게 3가지로 구분된다. 파일 이름, 해쉬값, 사이즈 등의 입력된 파일의 기본정보, 악성 실행 파일을 구성하고 있는 Section, Resource 등 정적 분석정보, 마지막으로 악성코드가 실행되면서 생성하는 모든 프로세스가 호출하는 윈도우 API 정보, HTTP, TCP, UDP 등의 네트워크 통신 정보를 포함한 동적 분석정보이다. 추가로 생성 파일에 대해 별도의 항목으로 결과에 나타낸다. Table 1은 Cuckoo Sandbox에서 확인할 수 있는 분석결과 정보를 나타낸다.

Table 1. Cuckoo Sandbox's Analysis Result Items

Result Items	Sub Items	Description
basic info	machine info	Cuckoo machine info
	file info	file name, size, hashes
	signature	malicious signature info
static analysis	version info	submitted file's versiont info
	sections	submitted file's section list
	resources	submitted file's resource list
	imports	submitted file's import info
	exports	submitted file's export info
	strings	submitted file's strings
behavior analysis	process id	created process id
	process name	created process name
	calls	called windows api info
	process tree	created process tree
	summary	summary of behavior result
	dropped info	created file info
	network info	network communication info

입력된 파일의 분석결과는 파이썬 딕셔너리 형태로 존재한다. 다만 딕셔너리에 포함된 행위정보의 경우 포괄적인 내용만을 가지고 있으며, 각 프로세스에서 호출된 윈도우 API 정보는 별도의 경로에 BSON 파일로 저장하고 있다. Cuckoo Sandbox는 위의 2가지 형태의 결과를 기반으로 HTML과 MAEC의 포맷으로 악성코드의 분석 결과를 출력한다. Cuckoo Sandbox의 소스코드를 살펴보면 Reporting 모듈에서 results 변수를 상속받는 것을 확인할 수 있는데 이 변수를 통해 입력된 파일의 통합적인 분석결과에 접근할 수 있다.

3.2 침해지표 자동생성

침해지표 자동생성 도구는 Cuckoo Sandbox에서 출력된 분석결과를 바탕으로 침해지표를 생성한다. 침해지표의 포맷은 XML 구조인 DFIOC의 Indicator 포맷을 개선하여 사용하였으며, 이를 통해 DFIOC의 Evidence 포맷으로 수집된 다양한 포렌식 아티팩트에서 침해흔적을 식별할 수 있다.

1) 침해지표 포맷

침해지표 자동생성 도구는 DFIOC의 Indicator포맷에서 Content 엘리먼트 속성에 정규 표현식과 가중치 항목을 추가하고, 가중치의 범위를 0에서 1로 지정하였다. 추가로 Comment 엘리먼트에 Indicator의 설명을 입력하여 사람도 쉽게 이해할 수 있도록 개선하였다. Table 2는 Indicator와 Content 엘리먼트의 속성 정보에 대한 설명이다.

Table 2. The Information of Indicator's Attribute

Element	Attribute	Description
Indicator	Weight	Indicator's malicious weight
	Composition	Indicator's logical operation
	Indicator Category	automatically inspectable indicator - Inspectable - Uninspectable
Content	IsRegEx	use of regular expression
	Weight	Content's malicious weight
	Condition	ContentValue searching condition - Equals - Contains - StartsWith - EndsWith - GreaterThan - LessThan

Fig. 1은 자동 생성된 침해지표에서 악성 파일의 해쉬 정보를 나타낸 Indicator이다. 침해지표로써 파일의 해쉬값은 그 자체로 확실한 식별 정보이다. 따라서 Indicator 엘리먼트의 속성값 중 가중치는 1, 논리 연산은 OR이 된다. 침해지표 자동생성 도구로 생성된 Indicator는 모두 조사관의 추가 확인 없이 자동판단이 가능하므로 Inspectable로 설정된다. Content 엘리먼트의 속성은 수집된 Evidence 파일과의 매칭 설정 정보를 나타낸다. 예제에서 해당 Content는 1의 가중치를 가지며 정규표현식을 사용하지 않고 동일한 값을 검색하도록 되어있다. EvidencePath는 수집된 Evidence 파일에서 저장된 증거 경로를 나타내며, 예제에서는 SHA1 해쉬값이 저장되는 경로를 간소화하여 입력하였다. Cuckoo Sandbox의 분석결과 중 침해사고 식별값은 ContentValue 엘리먼트에 입력되며, 예제에서는 입력파일의 SHA1 해쉬값이 입력된 것을 확인할 수 있다.

2) 침해지표 자동생성 알고리즘

침해지표 자동생성 도구는 Cuckoo Sandbox에 입력된 파일의 분석 결과인 result dictionary를 사용하여 침해사고의 식별을 위한 포렌식 정보인 파일의 기본정보, 정적 분석정보, 동적 분석정보에서 중요한 식별정보를 추출한다. 이후 각 지표를 DFIOC포맷의 침해지표를 생성하며, 각 지표의 가중치 및 논리조합을 자동으로 구성한다. 최근에는 가상환경을 식별하여 가상 환경에서는 동작하지 않는 가상화 우회 기법이 적용된 악성코드도 존재한다. 이러한 악성코드의 경우 잘못된 침해지표를 생성할 수 있으므로 침해지표 자동생성 도구에서는 가상화 우회 방지 기법[8]을 적용하여 침해지표를 생성한다.

각 Indicator와 Content의 가중치 부여는 가장 확실한 정보는 1.0의 가중치를, 동적 행위정보에는 0.75의 정적 행위정보에는 0.5의 가중치를 부여하였다. 이는 종합 가중치 결과를 각각 1, 0.5 , 0.1을 출력하여 위험도를 명확하게 구분하기 위해 설정을 하였다. Table 3은 침해지표 자동생성 도구에서 생성하는 Indicator의 속성정보와 하위 Content 엘리먼트의 속성정보를 나타낸 표이다.

파일의 기본정보에서는 입력된 File Hash와 File Info 2개의 Indicator를 생성한다. 가장 확실한 식별정보인 해쉬값은 1.0의 가중치를 가지며 동일한 값만을 찾아내는 검색조건으로 Condition 속성에 Equals 값을 가진다. File Info는 오탐율을 줄이기 위해 File Name, File Size, Compile Time 3가

```
<M-IOC:Indicator Weight="1." Composition="OR" IndicatorCategory="Inspectable">
    <M-IOC:KeyEvidence>
        <M-IOC:Content IsRegEx="false" Weight="1." Condition="Equals">
            <M-IOC:EvidencePath>*/FileHashValue/SHA1Value</M-IOC:EvidencePath>
            <M-IOC:ContentValue>619dabe4ea05346730e041e116e11436de815c32</M-IOC:ContentValue>
        </M-IOC:Content>
        <M-IOC:Comment>FileInfo-FileHash-SHA1</M-IOC:Comment>
    </M-IOC:KeyEvidence>
</M-IOC:Indicator>
```

Fig. 1. The Example of Indicator

Table 3. The Table of Automatic Creation Indicators

Result Item	Indicator	Composition	Weight	Content	Weight	Condition
basic information	File Hash	OR	1.0	hash(SHA1)	1.0	Equals
	File Info	AND	0.5	File Name	0.5	Contains
				File Size	0.5	Equals
				Compile Time	0.5	Equals
Static Analysis	PE Version Info	AND	0.5	Internal Name	0.5	Contains
				File Version	0.5	Contains
				Product Name	0.5	Contains
				Original File Name	0.5	Contains
	Section List	AND	0.5	Section Name	0.5	Contains
				Raw Size	0.5	Equals
				Entropy	0.5	GreaterThan, LessThan
	Resource List	AND	0.5	Resource Name	0.5	Contains
				Resource Type	0.5	Contains
				Raw Size	0.5	Equals
				Sub-Language	0.5	Equals
Behavior Analysis	Create File	OR	1.0	hash(SHA1)	1.0	Equals
		AND	0.75	File Name	0.75	Contains
				File Size	0.75	Equals
	Create Registry	AND	0.75	Registry Key	0.75	Contains
				Registry Value	0.75	Contains
	Network Analysis	OR	0.75	Domain Name	0.75	Contains
				IP Address	0.75	Equals

지의 조건을 AND 논리연산으로 그룹화하여 Indicator를 생성한다. 이렇게 바뀌기 쉬운 여러 조건을 AND 연산으로 그룹화 함으로써 침해지표의 식별률을 증가시킬 수 있다.

정적 분석정보는 PE Version Info, Section List, Resource List 정보를 바탕으로 생성되며, 각 Indicator의 생성 조건은 SANS 문서[9]를 기반으로 한다. PE Version Info는 파일의 이름과 파일의 Original File Name이 같이 않고 Original File Name이 악성코드에서 주로 사용하는 기본 프로세스의 이름과 동일한 경우 Internal Name, File Version, Product Name, Original File Name 4가지 Content를 AND 연산으로 그룹화하여 Indicator 를 생성한다.

Section List는 오탐률을 줄이기 위해 특수한 Section Name에서 Entropy의 값이 1보다 작거나 7보다 큰 경우와 Section의 Raw Size의 값이 0인 경우 2가지에 대해서 Section Name, Raw Size, Entropy 3가지 Content를 AND 연산으로 그룹화하여 Indicator를 생성한다. Entropy의 검색 조건을 적용하기 위해 7보다 큰 경우 GreaterThan 조건을 1보다 작은 경우를 LessThan 조건을 적용하여 매칭한다. Resource List는 각 Resource의 Sub-Language의 값이 0, 즉 Neutral인 경우 Resource name, Resource Type, Raw Size, Sub-language 4가지의 Content를 AND 연산으로 그룹화하여 Indicator를 생성한다.

행위 분석정보는 Create File, Create Registry, Network Analysis를 바탕으로 Indicator를 생성한다. 행위분석결과는 입력된 파일의 실행과정에서 호출된 윈도우 API를 기반으로 결과를 출력한다. Create File은 입력된 파일의 동작하는 과정에서 생성된 파일의 해쉬값을 OR 연산으로 그룹화하여 Indicator를 생성한다. 그리고 생성된 File Name과 File Size 2개의 Content는 AND 연산으로 그룹화되어 Indicator를 생성한다. Create Registry는 RegCreateKey, RegSetValue 2개의 윈도우 api 호출 정보를 기반으로 한다. 2개의 api를 통해 생성이 확인된 Registry Key와 Registry Value 2개의 Content를 AND 연산으로 그룹화하여 Indicator를 생성한다. 네트워크 분석정보는 동적 행위 분석을 통해 확인된 Domain Name과 IP Address 2개의 Content를 OR 연산으로 그룹화하여 Indicator를 생성한다.

3.3 침해지표 가중치 연산 및 매칭 경로

침해지표의 가중치는 0에서 1사이의 값으로 악성 의심 정도를 나타내고 있다. 생성된 침해지표와 수집된 Evidence의 매칭 과정 중 종합 가중치를 계산하여 시스템 내의 침해사고 의심정도를 최종적으로 나타낸다. 종합 가중치 계산과정은 AND와 OR논리 연산에 따라 별도의 계산식을 사용한다.

Equation (1)은 각 논리연산에 따른 종합 가중치 계산 공식을 나타낸 것이다.

$$\begin{aligned} AND\,Calc\,Weight &= \sum_{n} Matched\,Content\,Weight \times \frac{Composition\,Weight}{Content\,Total\,Count} \\ OR\,Calc\,Weight &= MAX(Matched\,Content\,Weight) \times Composition\,Weight \end{aligned} \quad (1)$$

Evidence 엘리먼트는 기존 DFIOC의 Evidence 엘리먼트를 기반으로 세분화 및 추가하였다. 침해사고 피해 시스템에서 수집된 포렌식 아티팩트는 Evidence경로에 해당하는 엘리먼트에 입력되며 매칭과정에서 침해지표의 Content가 경로 내에 같은 아티팩트가 존재하는지를 확인하여 결과를 표시한다. Table 4는 자동 생성된 침해지표가 주로 매칭이 되는 Evidence 엘리먼트를 나타낸 표이다.

Table 4. Matching Evidence Element

Element	Sub-Element	Description
SystemSetting Info	Installed ApplicationInfo	Installed program info
	AutorunInfo	Automatical running program info
	Services	Running services info
	HostsFileInfo	Windows host file info
	Registry	Windows registry info
ProcessInfo	Process	Running process info
UserSystem Activities	DownloadInfo	Download file info
	ExternalDriveInfo	External drive access info
	ShellBagInfo	changed folder info
	PrefetchInfo	executed program info
	AMCacheInfo	executed program info
	CompatibilityInfo	executed program info
	IconcacheInfo	executed program info
	EventLogInfo	executed program info
	NtfsJrnlInfo	executed program info
	EvidenceFiles	additional file info
Network Activities	DNSInfo	DNS cache info
	NetworkSessions	network session info
	NetworkIPPorts	IP, Port info
	WebhistoryInfo	web page access info
	NetworkPacket	network packet info

4. 침해지표 자동생성 결과 검증 및 활용방안

이 장에서는 침해지표 자동생성 도구로 작성된 실제 악성코드의 침해지표를 검증하기 위한 실험과 그에 대한 결과를 서술한다. 실험은 2개의 유사 행위를 하는 악성코드를 이용하여 진행하였으며 2개의 파일 이름과 해쉬값은 Table 5와 같다. 실험에 사용된 악성파일은 패킹 및 난독화가 되어 있지 않은 트로이목마 파일로써 사용자에게 거짓 악성 정보를 전달하여 치료를 위한 가짜 안티바이러스 소프트웨어를 다운로드하게 하거나 결제를 유도한다.

Table 5. The Tested Malware Samples

File Name	SHA1
Nzogaa.exe	33b643308ea4bdc4070ec57fe4588cec7b42bea2
Niceware.exe	2bf267ef5a71ba056b918abe7a8b57cb50d7b586

실험은 3개의 과정으로 구성된다. 첫 번째는 'Nzogaa.exe' 악성코드를 역공학으로 분석한 뒤 수기로 작성한 침해지표와 침해지표 자동생성 도구로 생성된 침해지표의 작성시간 및 정확도를 비교하였다. 두 번째는 'Nzogaa.exe'의 자동생성 침해지표와 가상환경에서 동일한 악성코드를 감염시킨 뒤 수집한 Evidence 파일과 매칭 결과를 확인하였다. 세 번째는 유사한 흔적을 남기는 변종 악성코드인 'Niceware.exe'를 감염시킨 뒤 수집한 Evidence 파일과 기존 침해지표의 매칭 결과를 확인하여, 유사한 악성코드의 식별에 대한 실험을 진행하였다.

4.1 수기 및 자동 작성 침해지표 생성시간 비교

이 실험에서는 침해지표 및 작성에 대한 교육을 받은 4명을 대상으로 'Nzogaa,.exe' 악성코드를 분석한 뒤 수기로 침해지표를 작성하는 시간을 측정하였다. 참여자는 현업에서 악성코드 분석을 진행한 전문가 한명과 악성코드 분석교육을 받은 3명이 대표적인 디버깅 도구인 Ollydbg와 IDA를 사용하여 분석하였으며 DFIOC Indicator 포맷으로 침해지표를 작성하였다.

4명의 참여자가 수기로 침해지표를 작성하는데 적게는 6시간 많게는 10시간으로 평균 8시간이 소요되었다. 참여자들은 악성코드의 분석을 통해 파일 열람, 동일 파일 삭제 후 재생성 등 세세한 행위 정보를 확인하였지만, 시스템에 남는 흔적을 작성하는 침해지표의 내용에는 차이가 없었다. 또한 수기 작성 시 개개인의 숙련도에 의해 작성 완료시간의 차이가 컸으며 오타로 인한 침해지표의 오탐 가능성을 확인할 수 있었다. 이에 반해 침해지표 자동생성 도구는 초기 Cuckoo Sandbox 설정에서 약 3분으로 표준화된 침해지표를 생성할 수 있었다.

4.2 자동 생성된 침해지표와 감염 시스템 매칭 결과

이 실험은 자동 생성된 침해지표가 감염 시스템에서 식별 가능한 정보를 통해 효용성을 검증하기 위해 진행하였다. 이를 위해 침해지표와 감염 시스템에서 수집된 Evidence 파일의 매칭을 위한 별도의 침해 분석 도구를 개발하였다. Table 6은 침해 분석 도구의 결과화면에 나타나는 항목과 그에 대한 설명이다.

Table 6. The Result Items of IOC Analysis Tool

Result Item	Sub item	Description
Indicator Hit Summary	FileName	indicator's name
	IsExactMatch	the result of composition
	HitCount	number of matched contents
	TotalIndicatorCount	number of all contents
	WeightSum	calculated total weight
Indicator Hit Contents	Hit Result Tree	indicator composition tree
	Hit Value	matched content value
	Hit Evidence Path	matched Evidence Path

1) 'Nzogaa.exe' 감염 시스템 매칭 결과

Fig. 2는 자동 생성된 'Nzogaa.exe'의 침해지표를 보기 쉽게 나타낸 그림이다. 해당 침해지표는 4개의 Indicator로 구성되어있다. File Hash Indicator는 분석한 파일의 해쉬와 동적행위 과정에서 생성된 파일의 해쉬가 OR로 그룹화 되

Indicators	Conditaion	Value	Weight	Comment
Indicators				
OR			1.	
OR			1.	File Hash Indicator
*/...	Equals	33b643308ea4bdc4070ec57fe4588cec7b42bea2	1.	FileInfo-FileHash-SHA1
*/...	Equals	953850b55faac462f2ea1e0ce0f0d1cdc53d5a44	1.	DroppedFile-FileHash-SHA1
AND			0.5	File Info Indicator
*/...	Contains	Nzogaa.exe	0.5	FileInfo-FileName
*/...	Equals	200704	0.5	FileInfo-FileSize
*/...	Equals	2009-09-08T03:27:33Z	0.5	FileInfo-PETimeStamp
AND			0.5	Static Info Indicator
*/...	Equals	163840	0.5	StaticAnalysis-SectionList-Size_of_data
*/...	GreaterThan	7.34880849201	0.5	StaticAnalysis-SectionList-entropy
*/...	Contains	CODE	0.5	StaticAnalysis-SectionList-SectionName
AND			0.75	Behavior Info Indicator
*/...	Contains	{62C40AA6-4406-467a-A5A5-DFDF1B559B7A}.job	0.75	CreateFile-FileName
*/...	Equals	284	0.75	CreateFile-FileSize

Fig. 2. The Indicators of Nzogaa.exe

어 있다. File Info Indicator는 분석 파일의 기본정보로 생성된 Indicator이며, Static Info Indicator는 정적 분석정보에서 악성 행위의 특징으로 가능성이 높은 Section List 정보로 Indicator가 생성되었다. Behavior Info Indicator는 동적 행위 분석과정에서 생성된 파일 정보로 Indicator를 생성하여 총 10개의 Content로 구성된 침해지표가 자동생성 되었다.

Fig. 3은 수집된 Evidence와 침해지표와의 매칭 결과를 나타낸 그림이다. 요약 내용을 살펴보면 총 10개의 Content에서 9개가 매칭된 것을 확인할 수 있으며, Indicator의 논리 조합을 완전히 만족하기 때문에 IsExactMatch가 true로 나타난다. 또한 4개의 Indicator 중 수집된 Evidence 파일내에 악성코드의 해쉬값이 존재하므로 연산된 종합 가중치가 1인 것을 확인할 수 있다.

상세 내용을 살펴보면 악성파일은 바탕화면에 존재하였으며 파일의 실행 정보는 프로세스의 정보와 프리패치 파일의 생성 부분에서 확인할 수 있다. 추가적으로 레지스트리 자

Indicator Hit Summary

Total Indicator Count : 1 Matched Indicator Count : 1

File Name	IsExactMatch	HitCount	TotalIndicatorCount	WeightSum
MalwareIOC-a7226f0e-18b7-4bb9-9c97-2b4364b650aa	true	9	10	1

Indicator Hit Contents

Hit Result Tree	Hit Value	Hit Evidence Path
MalwareIOC-a72...		
IndicatorsType		
FileArtifact	33B643308EA4BDC4070EC57FE4588CEC7B42BEA2	EvidenceFiles/EvidenceFile/File/FileHashValue/SHA1Value
FileArtifact	33B643308EA4BDC4070EC57FE4588CEC7B42BEA2	PrefetchInfo/Prefetch/ExecutableFile/File/FileHashValue/SHA1Value
IndicatorsType		
FileArtifact	Nzogaa.exe	PrefetchInfo/Prefetch/Prefetch/ProcessEXE
FileArtifact	NZOGAA.EXE-7F650506.pf	PrefetchInfo/Prefetch/Prefetch/PrefetchFileName
FileArtifact	C:\USERS\KBG\DESKTOP\Nzogaa.exe	PrefetchInfo/Prefetch/RefernceFileList/ReferenceFile/FullPath
FileArtifact	Nzogaa.exe	EvidenceFiles/EvidenceFile/File/FileName
FileArtifact	C:\USERS\KBG\DESKTOP\Nzogaa.exe	EvidenceFiles/EvidenceFile/File/FilePath
FileArtifact	Nzogaa.exe	ProcessInfo/Process/File/FileName
FileArtifact	Nzogaa.exe	SystemSettingInfo/AutorunInfo/Autorun/ExecuteFile/FileName
FileArtifact	200704	EvidenceFiles/EvidenceFile/File/FileSize
FileArtifact	2009-09-08T03:27:33Z	EvidenceFiles/EvidenceFile/PEFile/PETimestamp
IndicatorsType		
FileArtifact	163840	EvidenceFiles/EvidenceFile/PEFile/SectionList/Sectioin/SizeInBytes
FileArtifact	7.34880849201	EvidenceFIles/EvidenceFile/PEFile/SectionList/Sectioin/entropy
FileArtifact	CODE	EvidenceFIles/EvidenceFile/PEFile/SectionList/Sectioin/SectionName
IndicatorsType		
FileArtifact	{62C40AA6-4406-467a-A5A5-DFDF1B559B7A}.job	EvidenceFiles/EvidenceFile/File/FileName
FileArtifact	C:\WINDOWS\TASKS\{62C40AA6-4406-467a-A5A5-DFDF1B559B7A}.job	EvidenceFiles/EvidenceFile/File/FilePath
FileArtifact	284	EvidenceFiles/EvidenceFile/File/FileSize

Fig. 3. The Indicator Matching Result (First Evidence)

Indicator Hit Summary

Total Indicator Count : 1 Matched Indicator Count : 1

File Name	IsExactMatch	HitCount	TotalIndicatorCount	WeightSum
MalwareIOC-a7226f0e-18b7-4bb9-9c97-2b4364b650aa	false	4	10	0.28125

Indicator Hit Contents

Hit Result Tree	Hit Value	Hit Evidence Path
MalwareIOC-a72...		
IndicatorsType		
FileArtifact	200704	EvidenceFiles/EvidenceFile/File/FileSize
IndicatorsType		
FileArtifact	163840	EvidenceFiles/EvidenceFile/PEFile/SectionList/SectoIn/SizeInBytes
FileArtifact	CODE	EvidenceFiles/EvidenceFile/PEFile/SectionList/SectoIn/SectionName
IndicatorsType		
FileArtifact	{62C40AA6-4406-467a-A5A5-DFDF1B559B7A}.job	EvidenceFiles/EvidenceFile/File/FileName
FileArtifact	C:\WINDOWS\TASKS\{62C40AA6-4406-467a-A5A5-DFDF1B559B7A}.job	EvidenceFiles/EvidenceFile/File/FilePath

Fig. 4. The Indicator Matching Result (Second Evidence)

동실행 프로그램에 자기 자신을 추가한 것을 확인할 수 있다. 악성 프로그램이 실행되면서 생성된 파일과 위치는 확인할 수 있지만 파일의 해쉬가 매칭이 안 된 것을 알 수 있다. 이것에 대해서 추가로 확인 결과 해당 악성코드는 동일한 파일 이름에 무작위 값으로 구성된 파일을 생성하는 것을 알 수 있었다.

2) 'Niceware.exe' 감염 시스템 매칭 결과

Fig. 4는 변종 악성코드인 'Niceware.exe'를 감염시킨 뒤 수집한 Evidence 파일과 기존의 침해지표를 매칭한 결과이다. 결과를 살펴보면 각 Indicator의 논리조합까지 만족하는 매칭 결과가 존재하지 않기 때문에 IsExactMatch는 false가 되며 총 10개의 Content 중 4개가 매칭된 것을 확인할 수 있다. 상세 내용에서는 정적 분석정보의 Indicator 중 일부 Content의 매칭 결과가 존재하는 것을 알 수 있다. 또한 다른 악성코드지만 동일한 이름의 파일이 생성된 것을 확인할 수 있다. 종합 가중치는 0.28125인데 동적 분석결과의 Indicator의 종합가중치이며, 상대적으로 낮은 정적 분석결과의 종합가중치는 영향을 끼치지 않는다. 분석결과를 통해 유사 악성코드의 감염을 의심해 보아야 하며, 해당 파일의 생성시간을 기준으로 다운로드 및 실행한 파일을 검사함으로써 의심되는 파일의 범위를 줄일 수 있다.

4.3 침해지표 자동생성 도구 활용 방안

본 논문에서 제안하는 침해지표 자동생성 도구의 활용방법은 2가지이다. 첫 번째는 실제 침해사고에서 침해지표 생성 방법의 간소화이다. 침해사고 발생 시 발생 PC의 포렌식 아티팩트를 수집 및 분석하여 의심스러운 파일을 추출한다. 추출한 의심스러운 파일들을 침해지표 자동생성 도구에 입력하여 임시 침해지표를 생성한다. 수집 Evidence와 매칭 및 침해사고에 대한 분석을 진행하여 더욱 정교한 최종 침해지표를 생성한다. 이 방법을 통해 기존의 방법보다 빠르게 침해지표를 생성하여 유사 침해 사고를 방지하고 식별할 수 있다.

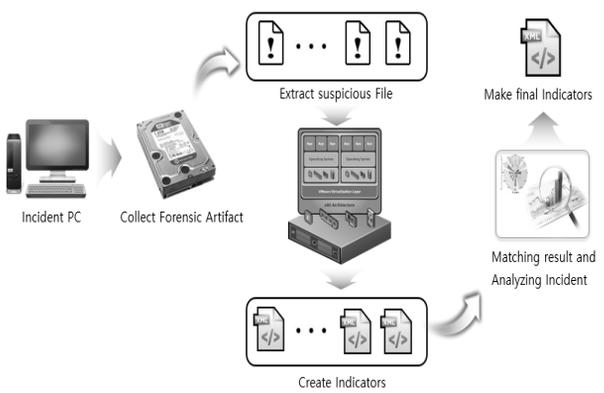


Fig. 5. Simplify Creation of Indicators

두 번째 방법은 별도로 악성코드를 수집하여 침해지표 자동생성 도구를 통하여 생성된 침해지표와 기존의 침해사고에서 제작한 최종 침해지표 리스트를 종합하여 침해지표 DB를 제작하는 것이다. 이를 통해 침해사고 발생 시 침해지표 DB에 존재하는 침해지표를 수집한 Evidence와 매칭을 하여 빠른 시간 내에 침해사고의 원인 및 경로에 대해서 식별이 가능하다.

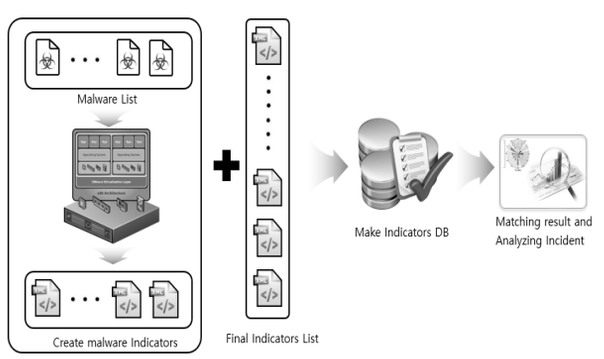


Fig. 6. Simplify Detection of Incidents

5. 결론 및 향후 과제

본 논문에서는 Cuckoo Sandbox를 이용하여 침해지표 자동생성 도구를 제작하고 이에 대한 검증 및 활용방안에 대해 제시하였다. 해당 도구를 통하여 Cuckoo Sandbox의 초기 분석 설정에서는 3~5분 이내에 의심스러운 파일에 대한 침해지표를 생성할 수 있다. 이를 통해 일반적으로 사용가능한 표준 형태의 침해지표를 기존의 수기식 방법보다 빠르게 생성할 수 있었다. 그뿐만 아니라 실험 결과를 통해 알 수 있듯이 자동 생성된 침해지표는 해당 악성코드가 감염된 시스템에서 감염 사실을 정확히 탐지할 수 있다. 반면 변종 악성코드의 경우, 행위 분석정보의 Indicator를 기반으로 유사 행위에 대해 탐지가 가능할 뿐 아니라 시간 정보를 통해 검사 범위를 줄일 수 있다.

침해지표 자동생성 도구는 유사 침해사고에 대한 빠른 식별을 위해 각 Content 및 Indicator 속성 정보에 가중치를 표현하고 논리 연산에 따른 종합 가중치의 계산방법을 제시하였다. 하지만 본 논문에서 제안한 방법은 현재 초기 버전으로 향후 더 다양한 악성코드 샘플을 활용하여 딥러닝과 데이터마이닝 기법을 통해 보다 정교한 가중치 및 계산 방법에 대한 개선이 필요하다.

References

[1] SANS, “Using IOC (Indicators of Compromise) in Malware Forensic.”

[2] The honeynet project, “Result of the Forensic Challenge” [Internet], http://old.honeynet.org/challenge/results/index.html.

[3] Wikipedia “Indicators of compromise” [Internet], https://en.wikipedia.org/wiki/Indicator_of_compromise.

[4] Mandiant [Internet], http://www.openioc.org.

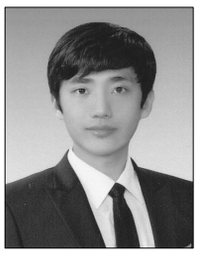
[5] MITRE [Internet], https://cyboxproject.github.io.

[6] Lee Min Wook, Yoon Jong Seong, and Lee Sang Jin, “Digital Forensic Indicators of Compromise Format(DFIOC) and Its Application,” *KIPS Tr.Comp. and Comm. Sys.*, Vol.5, No.4, pp.95-102, 2016.

[7] Cuckoo Sandbox [Internet], https://www.cuckoosandbox.org.

[8] Olivier Ferrand “How to detext the Cuckoo Sandbox and hardening it?” *22nd EICAR Annual Conference*, 2013.

[9] SANS, “Attributes of Malicious Files.”



강 봉 구

e-mail : bklove1618@korea.ac.kr
2012년 경희대학교 산업공학과(학사)
2015년~현 재 고려대학교 정보보호대학원 정보보호학과 석사과정
관심분야 : Digital Forensic, Reverse Engineering

윤 종 성

e-mail : yoonjs53@naver.com
2005년 공군사관학교 전산과학과(학사)
2013년~현 재 고려대학교 정보보호대학원 정보보호학과 석・박사통합과정
관심분야 : Digital Forensic, Information Security

이 민 욱

e-mail : zwmin1616@korea.ac.kr
2014년 경기대학교 전자공학과(학사)
2016년 고려대학교 정보보호대학원 정보보호학과 석사
관심분야 : Digital Forensic, Reverse Engineering

이 상 진

e-mail : sangjin@korea.ac.kr
1987년 고려대학교 수학과(학사)
1989년 고려대학교 수학과(석사)
1994년 고려대학교 수학과(박사)
1989년~1999년 ETRI 선임연구원
1999년~현 재 고려대학교 정보보호대학원 교수
2008년~현 재 고려대학교 디지털포렌식연구센터 센터장
관심분야 : Digital Forensic, Steganography, Hash Function