Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.27, NO.1, Feb. 2017 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2017.27.1.79

포렌식 준비도 제고를 위한 윈도우의 파일 시스템 감사 기능 설정 방안에 관한 연구

이 명 수,† 이 상 진‡
고려대학교 정보보호대학원

A Study on the Setting Method of the File System Audit Function of Windows for Enhancing Forensic Readiness

Myeong-Su Lee,† Sang-Jin Lee‡
Center for Information Security Technologies, Korea University

요 약

기업의 내부 정보 유출 감사 및 침해사고 사건에서 파일 처리 이력을 확보할 수 있다면 사용자의 행위를 좀 더 명확하게 추적하여 사건을 입증하는데 많은 도움이 될 수 있다. 윈도우에서는 파일 접근 이력을 확보할 수 있는 아티팩트들이 여럿 존재하나 부분적인 정보만 존재하거나 아티팩트의 특성상 오래 보존되어 있지 않아 사건 입증이 어려운 경우들이 많이 발생한다. 본 논문에서는 윈도우에서 제공하는 파일 감사 기능인 SACL(System Access Control List)을 활용하는 방법을 제안하고자 한다. 외부 솔루션을 도입할 수 없는 소규모 조직이라 하더라도 윈도우 설정을 강화하여 사고 발생 시 사건을 좀 더 명확히 입증할 수 있는 환경을 만들 수 있을 것이다.

ABSTRACT

If digital forensic investigators can utilize file access logs when they audit insider information leakage cases or incident cases, it would be helpful to understand user's behaviors more clearly. There are many known artifacts related to file access in MS Windows. But each of the artifacts often lacks critical information, and they are usually not preserved for enough time. So it is hard to track down what has happened in a real case.

In this thesis, I suggest a method to utilize SACL(System Access Control List) which is one of the audit functions provided by MS Windows. By applying this method of strengthening the Windows's audit settings, even small organizations that cannot adopt security solutions can build better environment for conducting digital forensic when an incident occurs.

Keywords: File Access Audit, System Access Control List, SACL, Digital Forensics, Forensic Readiness

I. 서 론

최근 기업들은 침해사고 및 내부 정보 유출, 개인정보 유출에 따른 피해를 실질적 위협으로 인식하고 있으며, 관련 법 준수를 위해서도 보안을 강화하고 있다. 기업들은 내부의 자산을 보호하기 위해 인터넷과 연결된 외부 경계부터 내부의 각 호스트 시스템에 이르기까지 각 구간에 걸쳐 DDoS 방어솔루션, 네트워크 방화벽, 애플리케이션 방화벽, IDS/IPS, Anti-Virus 등 다양한 보안 솔루션을 도입하고, 네트워크를 분리하는 등 많은 보호 대책을 강구하고 있다. 이런 기업의 경계 보안 활동이 강화됨에 따라 공격자가 외부에서 내부로 직접적인 해킹을 통해 진입하는 방식의 공격은 예전보다 줄어들고 있다. 하지만

Received(10. 18. 2016), Modified(12. 06. 2016), Accepted(12. 06. 2016)
† 주저자, crav3r@gmail.com
‡ 교신저자, sangjin@korea.ac.kr(Corresponding author)

여전히 APT(Advanced Persistent Threat)유형의 침해사고, 내부자에 의한 정보 유출 그리고 최근 기승을 부리는 랜섬웨어 감염 등의 사고는 근절되지 않고 있다. 그 이유는 여전히 외부에서 내부로의 연결 통로가 존재하고, 내부자의 인터넷 접근이 가능하기 때문이다.

이와 더불어 소프트웨어 취약점은 해 마다 5000개 이상 발견되고 있으며, 2015년에는 총 54개의 제로데이(zero-day) 취약점이 발견되었다[1]. 이는 일주일에 1개씩의 제로데이 취약점이 등장한 셈이다. 이러한 환경에서는 알려진 악성을 탐지하는 방식의 보안 제품을 이용하고, 최신 업데이트가 적용된 소프트웨어를 사용하며, 주의를 기울여 위험 사이트들을 방문하지 않더라도 인터넷을 사용하는 한 악성코드에 감염될 가능성은 여전히 존재한다.

따라서 외부로부터 유입되는 악성 파일을 탐지하고 차단에 집중하는 경계 보안 전략만으로는 조직의 내부 자산을 보호하는데 충분하지 않다. 내부 시스템을 지속적으로 모니터링을 수행하는 것과 함께 사고는 미연에 완벽히 방어 할 수 없다는 것을 인정하고 사고 후의 대응 방안을 마련하는 것이 필요할 것이다.

최근의 공격자들은 공격 대상에게 전자 우편을 통해 악성코드를 전달하거나, 특정 웹사이트에 악성코드를 심어 웹 사이트 방문자가 악성코드에 감염되도록 하는 방법을 주로 사용하고 있다. 얼마 전 발표된 LightCyber의 Cyber Weapon 2016 보고서[2]에 따르면, 취약점을 이용한 악성코드는 주로 최초 진입 시에만 사용되며, 공격자가 접근 권한을 획득한 이후에는 이미 시스템에 존재하는 시스템 관리 도구나 보안 점검용 도구 등을 활용하는 것으로 나타났다. 외부에서 악성코드가 유입되는 시점에 실시간으로 위협을 탐지하고 방어하는데 실패했다면, 침입을 인지하기까지는 상당한 시간이 소요될 가능성이 높다. FireEye의 Mtrend 2016 보고서에 따르면, 2015년도 침해 사고 분석 결과 사고 발생 후 사고를 발견하는데 걸리는 평균 시간은 126일 걸리는 것으로 나타났다[3].

내부자에 의한 위협도 이와 마찬가지이다. 악의적인 내부자는 자신이 속해 있는 조직의 시스템 및 네트워크에 접근이 가능한 인가된 사용자이다. 이들은 정당한 권한과 정상적인 방법을 이용하여 정보를 외부로 유출하며, 악성코드 등을 사용하지도 않으므로 보안 제품에 탐지 될 가능성이 매우 낮다. 망 분리, DRM, DLP, 매체제어, 물리 보안 등을 통해 외부 유출이 철저히 차단되어 있지 않은 환경이라면 정보의 유출을 막기란 결코 쉽지 않으며, 유출된 사실을 인지하는 것도 어렵다.

결국, 두 가지 유형 모두 사고 발생 후 사건을 인지하기 까지는 최소 수 개월 이상의 시간이 소요된다. 사건 발생 후 시간이 지남에 따라 시스템에 남아있는 관련 흔적들은 새로운 정보로 덮어 써지거나 소멸되므로, 점점 시간이 흐를수록 사건을 정확히 파악할 수 있는 가능성이 줄게 된다. 따라서 사건 분석을 위한 정보를 기록하고 이를 가능한 오래 보존하는 방법에 대한 고민이 필요하다.

본 논문에서는 포렌식 준비도 제고를 위해 파일 접근이력을 확보하는 방안에 대해 다루고자한다. 컴퓨터 시스템의 동작은 디스크에 저장된 파일을 메모리에 로드 한 후 프로세서가 이를 처리하고, 처리된 데이터를 다시 파일에 기록하는 행위가 반복되는 과정이다. 따라서 파일의 처리 이력 기록은 시스템에서 발생한 사건의 흐름을 파악하는데 많은 도움을 줄 수 있다. 하지만 윈도우 시스템의 기본 상태의 아티팩트에서 얻을 수 있는 파일 처리 정보는 주로 최종 접근 정보만이 기록되어 과거의 이력을 파악하기에는 다소 부족하다.

본 논문에서는 이를 보완하기 위한 방법으로 윈도우 시스템에서 제공되는 SACL 감사 기능을 활용하여 파일 접근 이력을 효과적으로 남기는 방안에 대해 제안하고자 한다.

II. 관련 연구

2.1 윈도우 이벤트 로그

윈도우에는 운영체제에서 발생된 사건을 기록하기 위한 로그 시스템이 제공되어 시스템 관리, 장애 처리 그리고 보안 목적 등에 활용할 수 있다. 또한 감사 정책 설정을 통해 기업의 보안 정책 및 시스템의 성격에 맞게 설정을 수행할 수 있으며 윈도우에서 제공되는 API를 이용하여 이벤트로그를 기록할 수도 있어, 백신, 데이터베이스, 모니터링 도구 등 다양한 응용프로그램들이 로그를 남기기도 한다.

2.1.1 윈도우 보안 감사 기본 정책

윈도우 이벤트로그에서 기록하는 여러 이벤트로그 중 보안감사 이벤트로그는 감사 정책 설정이 가능하

다. 보안 감사는 9개의 카테고리로 이루어져 있으나, 기본적으로 모두 활성화 되어 있는 것은 아니다.

클라이언트 버전(Windows XP, Vista, 7)과 서버 버전 (Windows 2003, 2008)에 따라 기본 설정이 다르게 구성되어 있다. 전체 53개 감사 항목 중 클라이언트는 4개 카테고리(계정관리, 로그온/로그오프, 정책변경, 시스템)의 12개 항목, 서버는 6개 카테고리(계정로그온, 계정관리, DS액세스, 로그온/로그오프, 정책변경, 시스템)의 17개 항목이 기본 설정되어있다[4].

2.1.2 윈도우 보안 감사 정책 기준

윈도우 시스템의 로그 설정 강화를 위해 미국 정부 기관(NIST, NSA)에서는 윈도우 보안 감사 가이드를 만들어 제공하고 있다[5][6]. 그러나 Table 1.를 보면 파일 접근 이력을 기록할 수 있는 Audit Object Access에 대한 성공 감사 설정은 추천되지 않고 있음을 알 수 있다. 최근 Microsoft Technet에 올라온 "Audit Policy Recommendations"[7]도 Audit Object Access 는 추천 대상에서 제외되었다.

아직 국내에는 감사 설정에 대한 상세한 가이드는 없으며, 국제 정보 보호 인증 제도인 ISO/IEC 27001:2013[8]의 12.4 항목에 해당하는 로깅과 관련된 부분이 ISMS[9] 11.6.2 "로그기록 및 보존" 심사 항목에 유사하게 포함되어 있는 정도이다.

Table 1. Audit Settings [5][6]

Audit Setting	NIST Enterprise	NIST SSLF	NSA
Audit account logon events	Success	Success, Failure	Success, Failure
Audit account management	Success	Success, Failure	Success, Failure
Audit directory service access	No Auditing	No Auditing	No Auditing
Audit logon events	Success	Success, Failure	Success, Failure
Audit object access	No Auditing	Failure	Failure
Audit policy change	Success	Success	Success, Failure
Audit privilege use	No Auditing	Failure	Failure
Audit process tracking	No Auditing	No Auditing	No Auditing
Audit system events	Success	Success	Success Failure

2.1.3 이벤트로그 중 파일 처리 관련

윈도우 시스템의 기본 설정 상태에서는 System.evtx 로그에서 서비스 실행, 종료 이력을 얻을 수 있고, Application.evtx 로그에서는 이벤트 로그에 기록하도록 제작된 프로그램이 남긴 정보를 통해 해당 프로그램이 실행 중이었음을 일부 추정할 수 있다. 그 외에 Microsoft Office가 동작 중 발생되는 오류나 다이얼로그 이벤트 등의 Alert이 발생된 경우 OAlerts.evtx 로그에 기록을 남기므로 이를 통해 엑세스(Microsoft Access), 파워포인트(Microsoft PowerPoint), 엑셀(Microsoft Excel), 워드(Microsoft Word) 등의 문서 파일 열람 및 수정 시도를 일부 파악할 수 있다. 이외에는 파일의 읽기, 생성, 수정, 실행, 삭제 등의 전체 행위를 파악할 수 있는 정보는 이벤트 로그에 거의 남지 않는다.

하지만 Security.evtx 로그에는 파일 처리의 전 과정을 기록할 수 있는 "개체 액세스 - 파일 시스템 감사"[10](object access-audit file system)와, 프로세스의 생성을 기록할 수 있는 "세부 추적 - 프로세스 만들기 감사"(detailed tracking-audit process creation)[11]가 있다. 이 두 가지 감사 모두 기본 설정은 비활성이다. 이 둘의 차이점은 Table 2.과 같다.

Table 2. Comparison between "Audit File System" and "Audit Process Creation"

	Description
Audit File System - Execute	- EventID is 4663 - need to set SACL on target files - not only exe fils but also module(dll) files - Parent process's name and PID
Audit Process Creation	- EventID is 4688 - TokenEvelevationType - don' need to set SACL - Parent process's PID

2.2 SACL (System Access Control List)

윈도우에서는 DACL(Discretionary Access Control List)을 이용하여 계정 별로 개체에 접근하는 것을 통제하며, SACL(System Access

Control List)을 이용해 개체에 대한 접근을 감사할 수 있다[12].

SACL을 이용하여 파일 접근을 감사하기 위해서는 두 단계의 설정이 필요하다.

1 단계 : 감사 정책 활성화
2 단계 : 감사 대상 개체에 SACL 설정

1단계는 Local Group Policy Editor 에서 Advanced Audit Policy - System Audit Policies - Local Group Policy - Object Access - audit File System을 Success 감사로 설정할 수 있다.

2단계는 탐색기에서 파일 및 폴더 개체의 속성에서 감사를 설정하거나, PowerShell 스크립트 등을 이용하여 설정할 수 있다.

위의 두 단계 설정이 완료되면, 개체에 접근 시도가 발생 시 접근을 유발한 주체 (사용자 계정, 프로세스)와 접근 유형, 개체 경로 및 이름 등의 정보가 윈도우 보안 이벤트로그(security.evtx)에 기록된다.

윈도우에서 파일에 대한 접근제어 및 감사는 13가지의 ACE (Access Control Entry)로 구성되어 있다. 파일 객체의 ACE 들을 읽기, 쓰기, 실행, 삭제의 유형으로 분류해 보면 Table 3.과 같다.

각 ACE는 내부적으로 4byte 크기의 Access Mask 값으로 표현되며 중첩 표현이 가능하다. 이벤트로그에 기록될 때는 AccessMask는 AND 연산된 0xH 형태의 16진수 값으로 기록되며, AccessList는 %%DDDD 형태가 나열되어 기록된다. AccessMask별 의미는 Table 4.와 같다.

Table 3. ACE types

Read type	- List folder/read data - Read attributes - Read Extended attributes - Read permissions
Write type	- Create files/write data - Create folders/append data - Write attributes - Write extended attributes - Change permissions - Take ownership
Execute type	- Traverse folder/execute file
Delete type	- Delete subfolders and files - Delete files

Table 4. Access Mask and Access List of File Access log

Access Mask	Access List	Description
0x1	%%4416	ReadData
0x2	%%4417	WriteData or AddFile
0x4	%%4418	AppendData or AddSubdirectory or CreatePipeInstance
0x8	%%4419	ReadEA
0x20	%%4421	Folder Traverse / Execute
0x80	%%4423	ReadAttributes
0x100	%%4424	WriteAttributes
0x10000	%%1537	Delete
0x20000		READ_CONTROL(read security descriptor)
0x40000		WRITE_DAC
0x80000		WRITE_OWNER

2.3 SACL 관련 Event ID

SACL 설정과 관련된 이벤트들은 Table 5.와 같다. 4670, 4985, 4719, 4907은 감사정책 설정 및 SACL 설정 시 기록되는 이벤트이며, 실질적으로 파일 접근과 관련된 이벤트는 4659, 4660, 4663, 4664이다. 4659와 4660은 개체 삭제와 관련되어 있으나, 삭제 이벤트는 4663에 중복으로 저장되므로 분석 시에는 4663 이벤트만을 대상으로 수행해도 무

Table 5. Event ID related to SACL

Event ID	Description
4659	A handle to an object was requested with intent to delete.
4660	An object was deleted.
4663	An attempt was made to access an object.
4664	An attempt was made to create a hard link.
4670	Permissions on an object were changed.
4985	The state of a transaction has changed.
4719	System audit policy was changed.
4907	Auditing settings on object were changed.

방하다.

2.4 SACL을 이용한 파일 감사의 장단점

파일 시스템 감사는 파일의 접근 이력을 기록한 것이므로, 파일이 삭제되어 존재하지 않더라도 접근 이력을 얻을 수 있으며, 해당 파일에 접근한 사용자와 프로세스 정보를 얻을 수 있다. 또한 SACL을 이용한 파일 감사는 OS의 기본 기능이므로, 추가 소프트웨어 설치가 필요하지 않다.

장점

- 행위의 주체, 대상, 행위내용, 행위시점을 알 수 있다.
- 감사 대상 및 감사 항목 선택이 가능하다.
- 동일한 시각에 발생된 이벤트라도 발생 순서를 구분할 수 있다.
- 상대적으로 흔적의 변조가 어려워 증거로써의 신뢰도가 더 높다.
- 로그의 용량 설정이 가능하고, 로컬 및 원격 백업이 가능하다.
- 운영체제에서 제공하는 로그 시스템을 사용하므로 안정적으로 해석이 가능하다.

단점

- 성능 저하 및 로그 폭증 가능성이 있어 적절한 설정 필요하다.
- 감사 대상 폴더 및 파일에 대해 SACL 설정을 관리해야 한다.

III. 제안 방법

3.1 제안 내용

윈도우에서 로그 유형이 아닌 아티팩트들은 대부분 최종 상태 값을 가지고 있어 침해 사고나 내부 정보 유출 등의 사건 분석 시 사건의 과정을 파악하고 입증에 활용하기가 힘든 측면이 있다. 또한 이마저도 시간이 흐를수록 점차 휘발되어 사건 발생 후 수 개월 이상 지난 분석 시점에는 획득이 어려운 상황도 자주 발생된다. 따라서 이를 보완하기 위해 파일 접근 이력을 충분히 남기는 것이 포렌식 준비도를 향상시킬 수 있는 방법 중 하나가 될 수 있을 것이다.

이를 위해 윈도우의 SACL을 이용할 것을 제안하고자 한다. 하지만 지금까지 SACL은 포렌식 준비도 향상 목적을 위해 적극적으로 사용되지 않아 왔다. 심지어 OS 제조사인 Microsoft사나 미국 정부기관 가이드에도 포함되지 않았다. 그 이유는 시스템에서의 파일 접근은 매우 빈번하게 발생하여 상시 로그로서 SACL을 활용할 경우 대량의 이벤트가 유발되고, 시스템에 부하를 유발할 수 있기 때문이다.

따라서 SACL을 실제로 활용하기 위해서는 감사 가능한 항목 중 사고 분석에 필요한 부분만을 선별하여 감사하는 것이 필요하다. SACL의 경우 일괄적으로 기록되는 다른 흔적들과 달리 개별 설정이 가능하므로, 상황에 따라 보다 유연한 적용이 가능하다. 또한 최근 SSD, 고용량 메모리 사용 등 시스템의 성능 향상도 SACL 적용에 긍정적으로 작용할 것이다.

본 논문에서는 파일의 접근 이력 기록을 위한 SACL 설정으로 13가지의 감사 항목 중 Execute(0x20), AppendData(0x4), Delete (0x10000)를 설정할 것을 제안한다. 또한 6개월 이상 기록을 보관하기 위해 Security.evtx의 로그 용량을 20GB 이상 확보할 것을 제안한다.

IV. 실 험

제안된 방법의 효과를 검증하기 위해 부하실험, 효과성 실험 그리고 기업 환경에서 발생될 수 있는 침해 시나리오를 가정하여 사건 분석에 활용할 수 있는지를 확인하였다.

4.1 부하 실험

4.1.1 Full ACEs 감사 (on IDLE PC)

SACL 설정이 발생시키는 로그의 양을 확인하기 위하여 Windows 7 시스템을 대상으로 전체 드라이브의 모든 파일 및 폴더 개체에 대해 Full ACEs(13개)를 활성화 하고, 사용자 개입이 없는 IDLE 상태로 운영한 결과 1시간 동안 총 230,599개의 4663 이벤트로그가 발생했다. 용량으로 따지면 SACL 로그 1개당 700 bytes 정도의 크기를 가지므로, 총 154MB(161,419,300 bytes = 230,599 * 700bytes)의 로그가 기록된 것이다.

Table 6.에서 ACE별 이벤트 발생 비율을 보면 Read 유형에 해당하는 0x1 (ReadData), 0x80 (ReadAttribute), 0x20000(ReadControl)이 전

Table 6. Number and Ratio of logs depending on AccessMask

AccessMask	Count	Ratio
0x1	112807	48.92%
0x100	100	0.04%
0x10000	177	0.08%
0x2	39	0.02%
0x20	2772	1.20%
0x20000	32365	14.04%
0x20008	3435	1.49%
0x21	15	0.01%
0x3	98	0.04%
0x4	15	0.01%
0x40000	10	0.00%
0x6	250	0.11%
0x80	78511	34.05%
0xC0000	5	0.00%

체 로그의 97%를 차지한다. 즉 시스템에서 파일에 접근하는 대부분의 행위는 읽기임을 알 수 있다.

또한 Full ACEs 감사 설정 상태에서 시스템을 재부팅하는 경우 약 10,000개(7MB)의 로그가 추가로 발생된다.

따라서 Full ACEs 감사 설정은 컴퓨터를 IDLE 상태로 켜둔 채 하루에 1번씩 재부팅 한다고 가정하면 하루에 약 3.6GB 정도의 로그(3703MB = 154MB * 24 hours + 7MB)가 발생하게 되므로 현실적으로 Full ACE 감사를 활성화하여 사용하기는 어렵다. 이 점이 SACL이 현실적으로 활용되기 어려운 주된 이유일 것이다. 그렇기 때문에 로그의 대부분을 차지하는 Read는 감사 대상에서 제외하는 것을 고려해야할 필요가 있다.

반면 포렌식 분석에 의미 있는 정보인 파일의 생성, 실행, 삭제와 관련된 0x4(AppendData), 0x20(Execute), 0x10000(Delete) 로그는 시간당 2964개의 로그(2MB)로 전체로그의 약 1.29%에 해당하므로, 감사 로그의 양을 상당히 감소시킬 수 있어 SACL을 이용한 파일 감사가 상시 감사에 활용될 수 있는 가능성을 보였다. 위와 같이 계산해 보면 하루에 약 55MB 정도의 로그(55MB = 2MB * 24 hours + 7MB)가 기록된다.

4.1.2 파일 수정, 실행, 삭제 감사 (on Active PC)

이번에는 IDLE 시스템이 아닌, 24시간 동작 중인 실제 업무용 시스템에서 ACE 중 0x4 (AppendData), 0x20(Execute), 0x10000 (Delete)를 활성화하여 1개월간 테스트한 결과, 4663 이벤트는 일평균 162,259개(108MB)가 발생되는 것으로 측정되었다. 따라서 Security.evtx의 크기를 20GB로 설정 시 약 6개월간의 파일 접근 로그를 보관할 수 있을 것이다.

Fig 1.과 이 프로세스를 기준으로 보면, 가장 많은 로그를 발생시키는 것은 안티바이러스 소프트웨어로써 이는 사용자의 행위와 관계없이 지속적으로 많은 로그를 발생시킨다. 두 번째로는 브라우저로 사용자 행위와 연관성이 높으면서 로그를 많이 유발시키는 것으로 나타났다. 그 외에는 크게 두드러지는 유형은 없었다. SACL은 개체 단위로 감사를 수행하도록 되어있어, 특정 프로세스를 감사에서 제외하거나, 포함시키는 등의 설정은 불가하다.

Fig 2.는 개체의 위치를 기준으로 분석된 결과이다. 사용자 폴더, 안티바이러스, 윈도우 폴더에서 파

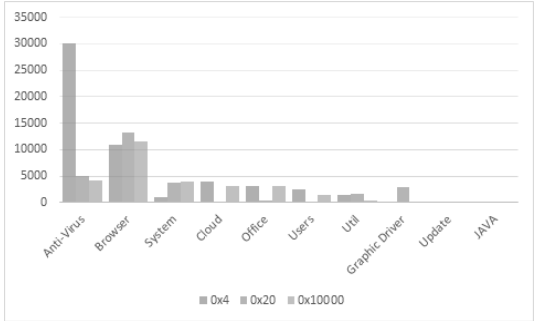

Fig. 1. Number of logs depending on process type.

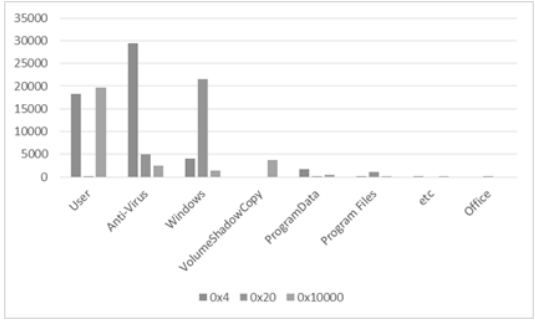

Fig. 2. Number of logs depending on location

일 처리 이벤트가 많이 발생했다. 사용자 폴더에서는 파일을 생성하고 삭제하는 로그가 비슷한 비율로 발생되었고, 안티바이러스 관련 경로에서는 파일 쓰기 로그가 많았으며, 윈도우 폴더에서는 실행 관련 로그가 많이 발생되었다. 따라서 이벤트 발생량을 더 줄여야 할 필요가 있을 때는 폴더별 설정을 세분화 하는 것이 방법이 될 수 있다.

4.2 효과성 실험

SACL의 효과를 확인하기 위하여 시스템에 발생될 수 있는 행위를 침입 킬 체인 단계(Intrusion Kill Chain)[13]로 구분하여 발생 가능한 행위 별로 기존 로그와 SACL 로그가 해당 행위를 식별할 수 있는지 확인하였다.

Fig 3.의 1~3번째 단계 까지는 공격 대상 시스템 외부에서 진행되는 과정이고, 4번째 Exploitation 단계부터는 피해 시스템 내에서 수행되는 과정이다. 따라서 Exploitation 이후부터 공격의 흔적들은 시스템에서 발견될 수 있다.

Table 7.은 공격자 또는 정보유출 행위자가 악성

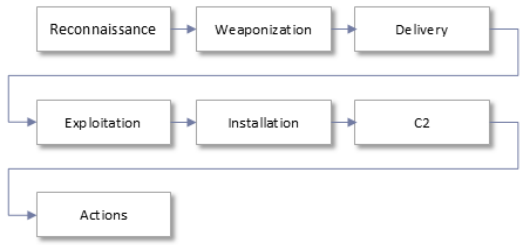

Fig. 3. Intrusion Kill Chain [13]

행위를 수행하고 관련 파일은 삭제하였으며, 이로부터 4개월이 지난 후에 아티팩트 획득이 가능한지를 확인 한 결과이며, 기본 설정을 사용하는 Windows 7 시스템을 대상으로 실험하였다.

공격과 관련하여 시스템 내부에서 발생 가능한 행위를 20가지로 구성하였다. 이 중 4개월 후 얻을 수 있는 아티팩트는 6개 행위에 그쳤다. 관련 행위 유형은 파일의 다운로드, 실행, 폴더접근 등이었다. 이 중에서도 과정 이력 정보까지 얻을 수 있는 경우는 폴더 접근 정보를 얻을 수 있는 shellbag이 유일했다.

하지만 SACL를 이용하여 "쓰기", "실행", "삭제"

Table 7. Effectiveness of SACL against suspicious behavior on Intrusion Kill chain

	Phase	Suspicious behavior on host	Related artifacts	Long-term	History	SACL
outside - host	Reconnaissance	N/A	N/A	N/A	N/A	N/A
	Weaponization	N/A	N/A	N/A	N/A	N/A
	Delivery	watering-hole	N/A	N/A	N/A	N/A
		send spear-phishing email	N/A	N/A	N/A	N/A
inside - host	Exploitation	visit malicious webpage	Browser	No	Yes	No
		execute exploit codes	Memory	No	No	Yes
		read malicious email	Browser, MailClient	No	Yes	No
		execute attached files	LNK, Prefetch	Yes	No	Yes
		unpack attached files	$MFT, NTFS Log	No	No	Yes
	Installation	drop or Download malwares	Browser Cache	Yes	No	Yes
		deploy malwares(copy,move)	$MFT, NTFS Log	No	Yes	Yes
		delete malwares	$MFT, NTFS Log	No	Yes	Yes
		execute malwares	Prefetch, Userassist	Yes	No	Yes
	C2	connect to C2	DNSCache	No	No	No
	Actions on Objectives (similar to internal user's behaviors)	open folders	Shellbag, Jumplist	Yes	Yes	Yes
		open files(documents)	Jumplist,MRU,LNK	Yes	No	Yes
		create Files	$MFT, NTFS Log	No	Yes	Yes
		copy Files	$MFT, NTFS Log	No	Yes	Yes
		edit Files	$MFT, NTFS Log	No	Yes	Yes
		execute Files	Prefetch, Userassist	Yes	No	Yes
		delete Files	$MFT, NTFS Log	No	Yes	Yes
		pack Files	$MFT, NTFS Log	No	Yes	Yes
		send email with attached file	MailClient	No	Yes	No
		visit web pages	Browser	No	Yes	No

설정을 적용했다면 20개 행위 중 15개의 행위에 대한 정보를 얻을 수 있는 것으로 파악되어, 악성 행위 관련 흔적 획득률이 30%에서 75%로 향상됨을 알 수 있다.

SACL 로그로 파악 가능한 행위 중 기존 아티팩트를 보강해주는 효과가 있는 경우가 40%, 기존 아티팩트가 남지 않았던 행위에 대한 정보를 제공해 주는 경우가 60% 인 것으로 파악되었다.

SACL로그는 Fig 4에서 볼 수 있는 바와 같이 누가(User, Process) 어떤 객체에(Target) 어떤 행동(read,write,delete,execute)을 했는지를 알 수 있다. 또한 파일 실행과 관련해서는 DLL파일의 실행 정보도 남게 되므로 다른 유형의 아티팩트와 비교했을 때 동일한 정보라 하더라도 좀 더 사건 분석에 효과적이다.

하지만 제안한 방식을 이용하더라도 웹사이트를 방문하거나, 메일을 읽고 발송하는 행위, C2와의 접속 등 네트워크와 관련된 행위에 대해서는 흔적이 남지 않았다. 특히 최신 브라우저들은 브라우저 사용 흔적(History, Cache, Cookie)을 데이터베이스 형태로 남기거나, 랜덤파일명을 사용하므로, 파일 접근 이력을 확인만으로는 행위의 식별이 불가했다.

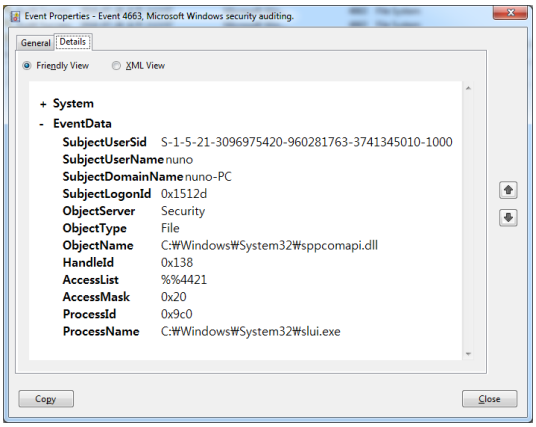

Fig. 4. a example SACL eventlog

4.3 가상 시나리오 실험

4.3.1 취약점을 이용한 악성코드감염

4.3.1.1 시나리오

피해자는 웹 서핑을 하던 중 악성코드가 삽입된 웹사이트를 방문하게 되었으며, 브라우저 취약점을 이용한 Drive-by Download 공격에 의해 악성코드가 피해자 컴퓨터로 다운로드 되고 실행되었다. 유입된 악성 파일은 RAT(Remote Administration Tool) 유형의 악성 프로그램으로, 실행 후 C&C(Command and Control) 서버로 네트워크가 연결되었으며, 공격자는 악성프로그램을 이용해 시스템을 살펴보고, 일부 문서 파일을 수집하여 압축 후 유출하였다. 시스템에는 여전히 악성 코드가 동작 중이었다. 침해 사실은 수주가 지난 후 인지하게 되었으며, 피해자는 그 동안 매일 시스템을 재부팅했으며, 시스템 정리 도구를 사용하였다.

4.3.1.2 분석 결과

SACL 설정을 하지 않은 경우, 악성코드 파일과 레지스트리 Run키에 등록한 흔적만 발견되었다. 이마저도 악성코드가 삭제되지 않고 남아서 동작 중이라고 가정했기 때문에 획득 가능한 흔적이었다. 만약 공격자가 공격 수행 후 악성코드를 삭제했다면, 분석 시점에는 얻을 수 있는 정보가 거의 전무했을 것이다.

Table 8. Scenario 1

Phase	Activity flow	without SACL	SACL Log
Delivery	visit web site		
	redirect to malicious site		
Exploitation	exploit		O
Installation	download malware1		O
	execute malware1		O
	register malware2 in run key	O	
	drop other malware2	O	O
	execute malware2		O
C2	connect to C&C		
Actions on Objectives	pack document files		O
	transfer packed documents		
	delete packed documents		O
	delete malware1		O

SACL 설정을 한 경우, Exploit 코드의 실행, 악성코드 설치, 정보 수집, 수집 파일 및 악성코드 삭제 등의 과정을 파악할 수 있었다. 하지만, 레지스트리 Run키 등록, C2접속, 네트워크를 통한 파일전송 등은 확인할 수 없었다.

4.3.2 이메일을 통한 랜섬웨어 감염

4.3.2.1 시나리오

공격자는 피해자에게 랜섬웨어를 첨부하여 스피어피싱 메일을 발송하였으며, 피해자는 의심 없이 Outlook을 이용하여 이메일을 읽고 첨부파일을 다운로드한 후 압축을 해제하고, 실행하였다. 랜섬웨어는 곧바로 실행되어 시스템에 있는 모든 문서 파일을 암호화 했으며, 모든 폴더마다 암호화 사실을 알리기 위한 문서 파일을 생성했다. 랜섬웨어는 사용자가 VSC(Volume Shadow Copy)를 이용하여 파일 복원을 하지 못하도록 하기 위해 기존에 저장되어 있던 VSC를 삭제하고 랜섬웨어 파일 자신도 삭제했다.

4.3.2.2 분석 결과

이번 시나리오에서는 SACL을 설정한 것과 그렇지 않은 것의 차이가 크지 않았다.

이유는 랜섬웨어를 이용한 공격은 공격의 목표가 시스템을 장악하고 장시간에 걸쳐 정보를 몰래 수집하는 APT 유형의 공격과는 목적이 다르기 때문이다.

Table 9. Scenario 2

Phase	Activity flow	without SACL	SACL Log
Exploit	read email	O	
Installation	save attached file	O	O
	unpack file		O
	execute attached file	O	O
Actions on Objectives	encrypt files by ransomeware		O
	create files to notify damage	O	O
	execute wmic (for delete VSC)	O	O
	delete VSC		
	delete malware by itself		O

랜섬웨어 실행 후 사용자가 피해 사실을 쉽게 인지할 가능성이 높아 비교적 단시일 내에 분석에 착수할 수 있어 아티팩트들이 소멸되기 전에 확보가 가능하다.

SACL을 설정한 상태에서는 Outlook을 이용한 메일의 읽기와 VSC를 지운 것은 확인할 수 없었으나 첨부된 악성파일의 다운로드, 압축해제, 악성코드 실행, 문서 파일 암호화, 협박용 문서 생성, VSC삭제를 위한 wmic 실행, 자가 삭제 등의 거의 전 과정을 확인할 수 있었다.

4.3.3 내부 직원의 정보 유출

4.3.3.1 시나리오

퇴사를 앞둔 회사원 A씨는 사내 네트워크에서 다른 사용자들의 공유 폴더를 발견하고, 그곳에 저장되어 있던 다수의 문서들을 자신의 시스템으로 복사하였으며, 일부 엑셀 파일은 직접 열어 내용을 확인하였다. 자신의 시스템에 설치된 네이버 클라우드 네트워크 드라이브로 파일들을 복사하여 외부로 반출하였으며, 로컬 시스템에서는 파일들을 삭제했다. 한 달 후 퇴사 프로세스에 따라 시스템 감사를 실시하였다.

4.3.3.2 분석 결과

내부자 유출 행위와 관련해서는 SACL을 설정하지 않은 경우라도 폴더의 접근이나, 문서파일 오픈 등은 Shellbag, Jumplist, MRU, lnk 등을 통해 해당 문서파일의 존재 및 접근 여부는 비교적 쉽게 확인이 가능했다. 또한 클라우드 프로그램을 이용한 유출 시에는 클라우드 프로그램이 남기는 로그를 통해 유출 파일 목록을 획득할 수 있었다. 하지만 내부

Table 10. Scenario 3

Activity flow	without SACL	SACL Log
Access shared folders on other computers	O	
copy the files to local folder (deleted)		O
open a few excel files	O	O
move files to naver cloud network drive	O	
delete files		O

자의 유출 행위 이후에 수행된 파일 삭제 행위와, 과거의 접근 이력 등에 대해서는 알 수 없었다.

SACL을 설정한 경우 파일이 삭제 목록을 모두 확인할 수 있었으며, Jumplist, MRU, lnk 등의 파일 접근 아티팩트와 달리 SACL로그는 파일을 처리한 프로세스와 시간이 누적하여 기록되어 있어, 장기간의 SACL 로그를 분석하여 과거에서 사용했던 파일인지 퇴사를 앞두고 갑자기 생성된 것인지 등 사용자의 행위 흐름을 파악할 수 있어 의도성을 입증하는데 좀 더 효과적임을 확인하였다.

SACL 로그를 주기적으로 모니터링 한다면, 사용자별 취급 문서 목록 파악이 가능하여 조직 내의 문서 흐름을 파악하는 용도로도 활용할 할 수 있을 것이다.

4.4 SACL 활용 실 사례

A사에서는 내부적으로 사용하는 Windows Server 2012 서버 3대에서 일 주일에 한 두 차례씩 특정 폴더 및 파일이 비정기적으로 삭제되는 현상이 발생되었다. 해당 업체에서는 이상 증상을 인지하고 1개월간 시스템을 조사하였으나 결국 원인을 찾지 못하고 분석을 의뢰하였다.

이에 해당 서버 3대에 대해 증거들을 수집하고 분석하였으나, 외부로부터의 침입 흔적은 없었으며 특별히 주목할 말한 점은 발견되지 않았다. 3대 모두 감사 설정이 강화되어 있었으나, Security Eventlog 로그의 크기가 기본 크기(20MB)로 설정되어 있어, 약 2시간가량의 로그만 남아 있었으며, 남아있는 로그의 대부분도 네트워크 연결 및 차단과 관련한 로그가 대부분을 차지하고 있었다.

파일 시스템 감사 설정 활성화 및 SACL 설정을 (Append, Execute, Delete) 한 후, Security.evtx 로그의 크기 제한을 20GB로 설정하였다. 증상의 재발 여부를 하루 단위로 확인하였으며, 8일이 지나자 이상 증상이 재발되었다.

이벤트로그를 분석한 결과, 비정기적으로 삭제되던 파일에 접근하는 프로세스 목록을 확보할 수 있었다. 웹 서비스용으로 사용되는 tomcat7.exe 프로세스가 cmd.exe를 실행하고, cmd.exe가 해당 파일들을 삭제했음을 확인할 수 있었다. 추가로 분석한 결과 서버에서 데이터 처리 시 사용된 임시폴더를 삭제하는 코드의 버그로 인해 드라이브 전체를 삭제하는 코드가(“cmd.exe /c rmdir /S /Q \”) 실행된 것으로 확인되었다.

본 사건은 이상 증상의 원인이 장기간 파악되지 않은 채 시스템을 재설치 하는 것으로 대응될 수 있었던 실 사례로, SACL을 통한 감사 방법이 실제 사건 조사에 효과적으로 활용될 수 있음을 확인할 수 있었다.

V. 결론 및 추후 연구

5.1. 결론

본 논문에서는 윈도우의 SACL 기능을 활용하여 파일의 접근 이력을 로그로 남기는 방안을 제시하였다. 파일 시스템 감사 설정 시 대량의 로그가 발생되는 원인이 파일의 읽기 유형의 감사에 있음을 밝히고, 파일의 수정, 실행, 삭제로 감사를 제한한다면 로그의 발생량을 현실적으로 수용 가능한 수준으로 줄일 수 있음을 확인하였다. 또한 이렇게 설정한 로그가 침해 단계 별로 흔적을 확인하고, 기존 아티팩트보다 사건의 흐름을 파악하는데 훨씬 효과가 있음을 확인하였다.

SACL 설정을 한다고 해서 모든 흔적이 남는 것은 아니다. 또한 분석의 속도가 향상되는 것도 아니다. 다만 파일 감사 로그와 기존 아티팩트의 교차분석을 통해 이전에는 추정으로 넘어가야했던 부분들에 대한 흔적간의 연결이 좀 더 강화될 수 있는 효과를 얻을 수 있을 것이다.

5.2. 추후 연구

감사 항목을 제한하여 SACL 로그의 발생량을 줄인다 하더라도 SACL은 여전히 적은 용량의 로그가 아니다. 사고 분석 시 집중되는 데이터 처리 부하를 줄이기 위해서는 주기적으로 SACL 로그를 분석하여 요약 정보를 남기는 방안에 대한 연구가 필요할 것이다. 또한 시스템의 포렌식 준비도 향상을 위해서는 파일 시스템에 대한 로그 이외에도 네트워크, 웹 브라우저 등 다른 아티팩트들의 수명도 6개월 이상 유지할 수 있어야 한다. 추후 연구로 윈도우 시스템에서 사건 해결에 필요한 주요 아티팩트들이 최소 6개월 이상 고르게 유지할 수 있는 방안에 대한 연구를 진행할 예정이다.

References

[1] Symantec, "Internet Security Threat Report 2016," https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf, vol. 21, Symantec, Apr. 2016,

[2] LIGHTCYBER, "CYBER WEAPONS 2016 REPORT," http://lightcyber.com/wp-cyber-weapons-report-lp/, LIGHTCYBER, Jun. 2016

[3] FireEye, "M-TRENDS 2016," https://www2.fireeye.com/WEB-M-Trends-2016-KO.html, FireEye, Feb. 2016

[4] Microsoft, "Audit Policy," https://technet.microsoft.com/en-us/library/cc766468(v=ws.10).aspx, 2016

[5] Murugiah Souppaya, Karen Kent and Paul. M. Johnson, "Guidance for Securing Microsoft Windows XP Systems for IT Professionals: A NIST Security Configuration Checklist," NIST Special Publication 800.68, vol. 68, Oct, 2005

[6] R. BICKEL, M. Cook and J. Haney, "Guide to Securing Microsoft Windows XP," National Security Agency, pp. 1-129, Oct. 2002

[7] Bill Mathers, "Audit Policy Recommendations," https://technet.microsoft.com/en-us/windows-server-docs/identity/ad-ds/plan/security-best-practices/audit-policy-recommendations, Microsoft, 2016

[8] ISO, "Information technology - Security techniques - Information security management systems - Requirements," ISO/IEC 27001:2013, Oct, 2013

[9] KISA, "Detailed check items of ISMS certification standard," http://isms.kisa.or.kr/kor/notice/dataView.jsp?p_No=48&b_No=48&d_No=114&cgubun=&cPage=1&searchType=ALL&searchKeyword=isms, KISA, May. 2013

[10] Microsoft, "Audit File System," https://technet.microsoft.com/ko-kr/library/dd772661(v=ws.10).aspx, Microsoft, Jun. 2009

[11] Microsoft, "Audit Process Creation," https://technet.microsoft.com/ko-kr/library/dd941613(v=ws.10).aspx, Jun. 2009

[12] Microsoft, "Access Control Lists," https://msdn.microsoft.com/en-us/library/windows/desktop/aa374872(v=vs.85).aspx, Microsoft, 2016

[13] Eric. M. Hutchins, Michael J. Cloppert and Rohan M. Amin, "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," Leading Issues in Information Warfare & Security Research, pp. 80-94, Mar. 2011

〈저자소개〉

이 명 수 (Myeong-Su Lee) 정회원
2012년 2월: 평생교육진흥원 컴퓨터공학 학사
2012년 2월~현재: 고려대학교 정보보호대학원 정보보호학과 석사과정
2011년 8월~현재: 안랩 책임연구원
〈관심분야〉 디지털포렌식, 침해사고분석, 데이터분석, 데이터시각화

이 상 진 (Sangjin Lee) 종신회원
1987년 2월: 고려대학교 수학과 졸업
1989년 2월: 고려대학교 수학과 석사
1994년 8월: 고려대학교 수학과 박사
1989년 10월~1999년 2월: ETRI선임 연구원
1999년 3월~2001년 8월: 고려대학교 자연과학대학 조교수
2001년 9월~현재: 고려대학교 정보보호대학원 교수
2008년 3월~현재: 고려대학교 디지털포렌식연구센터 센터장
2012년 7월~2013년 12월: 디지털포렌식연구회 회장
2013년 2월~현재: 고려대학교 정보보호대학원 부원장
2013년 11월~현재: 한국디지털포렌식학회 회장
〈관심분야〉 디지털 포렌식, 심층 암호, 해쉬 함수