Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.26, NO.5, Oct. 2016 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2016.26.5.1131

레지스트리 접근권한 변조에 관한 포렌식 분석 연구

김 한 기,[1†] 김 도 원,[1] 김 종 성[1,2‡]
[1]국민대학교 금융정보보안학과, [2]국민대학교 수학과

Study on Forensic Analysis with Access Control Modification for Registry

Hangi Kim,[1†] Do-Won Kim,[1] Jongsung Kim[1,2‡]
[1]Dept. of Financial Information Security, Kookmin University,
[2]Dept. of mathematics, Kookmin University

요 약

레지스트리 하이브 파일 구조에서 sk(security key) 셀은 레지스트리 키에 대한 접근제한 기능을 제공한다. 따라서 sk 셀에 대한 악의적인 변조가 이루어진 하이브 파일이 사용된다면 공격자는 레지스트리의 비밀 정보를 알아내거나 이벤트 로그 감사정책, 휴지통, 프리패치 생성여부와 같은 보안과 관련된 설정을 조작할 수 있다. 본 논문은 하이브 파일을 셀 단위로 변조하여 레지스트리 키의 접근제한 속성을 조작할 수 있는 유효한 방법을 제시한다. 그리고 이러한 조작으로 인해 일어날 수 있는 보안 측면의 문제점과 의도적으로 변조된 하이브 파일에 남아 있을 수 있는 흔적에 대해 논의한다.

ABSTRACT

In the Hive file format, the sk(Security Key) cell provides access control to registry key. An attacker can figure out secret information on registry or change the security set-up if she could apply modified hive files on system. This paper presents various methods to change access control of registry key by modifying or replacing cell on hive file. We also discuss threats by access control modification and signs of attacks analysis by modified hive files.

Keywords: Forensic, Registry, Hive file, Security key, Access control, Modification

I. 서 론

레지스트리는 윈도우 운영체제가 시스템 운영에 필요한 정보를 저장하는데 사용하는 데이터베이스이다. 부팅과정부터 윈도우 계정 로그인, 서비스 실행 등 거의 모든 사용자행위에 관여하며 포렌식 관점에서 매우 유용한 윈도우 아티팩트인 휴지통, 프리패치, 이벤트 로그 등의 설정도 저장된다. 그러므로 레지스트리 정보는 시스템 활동의 타임라인을 수립하거나 사용자 행동을 분석해야 하는 컴퓨터 포렌식 수사에 있어 매우 유용하다.

레지스트리 정보는 하이브 파일이라는 물리적인 파일에 존재하며 셀이라는 단위로 저장된다. 사용자는 윈도우 운영체제가 제공하는 레지스트리 편집기(이하 regedit)로 레지스트리 정보를 열람할 수 있다. regedit에서 폴더와 같이 보이는 것을 키, 내부에 존재하는 정보들은 값이라고 하며 각각은 하이브 파일내에 특정 셀에 대응된다. 그러나 sk 셀이 제공하는 접근제한 기능으로 인해 계정에 따라 볼 수 없는 정보도 존재한다. 예를 들어 HKLM\SAM\SAM이나 HKLM\SECURITY와 같은 키는 시스템 계정으로만 접근 가능하도록 설정되어 있어 regedit에서 열람할 수 없다. 또한 프리패치 파일 생성 설정을 저장하는 PrefetchParameters 키의 Enable

Received(05. 30. 2016), Modified(1st: 09. 01. 2016, 2nd: 09. 01. 2016), Accepted(09. 09. 2016)
† 주저자, tiontta@kookmin.ac.kr
‡ 교신저자, jskim@kookmin.ac.kr(Correspinding autor)

Prefetcher 값은 일반사용자 계정으로 수정이 불가하다.

일반 사용자가 레지스트리의 접근제한을 무시하고 키에 접근, 수정할 수 있다면 큰 보안 문제가 발생한다. 예를 들어 SECURITY 하이브에서 관리하는 감사정책 수정이 가능해져 특정 이벤트 로그가 생겨나지 않도록 하는 등의 감사정책을 설정할 수 있다. SAM 하이브에서 관리하는 계정정보와 사용자 프로필정보가 노출될 수 있다. 또한 프리패치에 대한 설정을 저장하는 레지스트리 키를 수정하여 프리패치 파일의 생성을 막을 수 있다. 따라서 레지스트리의 접근제한 설정은 보안과 직결되며 포렌식 관점에서 중요하게 다루어져야만 한다.

본 논문에서는 레지스트리 정보에 대한 접근제한 설정을 조작하기 위해 하이브 파일을 위·변조 하였다. 그 결과, nk 셀을 변조하는 방법으로 단일키에 대한 접근제한 속성을 강화 혹은 약화시킬 수 있었다. 또한 sk 셀을 변조하는 방법으로 해당 sk 셀을 참조하는 모든 키에 대한 접근제한 설정을 조작할 수 있었다. 공격자가 수정 권한이 없는 레지스트리 정보를 조작해 시스템에 적용시킨다면 프리패치나 이벤트 로그 등의 포렌식 관점에서 중요한 요소들의 설정을 바꿀 수 있다. 계정정보와 프로필정보와 같은 레지스트리에 대한 접근권한을 약화시켜 사용자 정보를 탈취하는 것도 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 하이브 파일 구조와 키 삭제 시 하이브 파일 변화에 대해 설명한다. 3장에서는 하이브 파일내의 sk 셀 혹은 nk 셀을 변조하여 레지스트리 정보에 대한 접근제한 설정을 바꾸는 방법을 제시한다. 또한 운영체제가 하이브 파일을 인식할 수 있도록 각각의 변조 법에 대한 유효한 적용방안을 강구하였다. 4장에서는 실험결과를 활용할 수 있는 포렌식 기술을 제시하고 대응방법에 대해 논의한다. 5장에서 결론 및 향후 연구를 제시한다.

II. 레지스트리 구조

Windows 95/98 운영체제부터 레지스트리 데이터베이스 체제가 적용되었고 Windows XP/Vista/7부터는 이전보다 더 많은 레지스트리 데이터를 다룬다. 레지스트리 정보는 모두 하이브 파일로 관리되며 실시간으로 업데이트된다. 예를 들어 regedit에서 볼 수 있는 키 중 HKU의 서브키는 %USERPROFILE%에, HKLM의 서브키는 %SYSTEMROOT%System32\config에 하이브 파일의 내용으로 존재한다. 이와 같이 regedit에서 보이는 모든 레지스트리 정보는 각각 하이브 파일에 저장된다. 그리고 하이브 파일에는 존재하지만 regedit에서는 확인할 수 없는 내용이 있는데, 이런 현상은 해당 셀이 sk 셀에 의해 접근제한 설정이 걸려있기 때문에 일어난다. 본 장에서는 레지스트리 정보를 저장하는 하이브 파일의 구조를 설명하고 키 삭제 시 하이브 파일의 변화에 대해 서술한다.

2.1 하이브 파일 구조

하이브 파일은 레지스트리 정보를 저장하는 물리적인 파일로, 윈도우는 시스템 변화에 따라 이 파일을 실시간으로 업데이트한다. regedit으로 확인할 수 있는 모든 정보는 물리적으로 하이브 파일에 존재하며 그 각각의 의미는 [1]에서 확인 가능하다.

2.1.1 전체 구조

레지스트리 하이브 파일은 Fig. 1.과 같이 블록(block), 빈(bin), 셀(cell)이라는 구성요소를 가진다[2,3]. 블록은 4,096바이트이므로 하이브 파일의 크기는 항상 4,096바이트의 배수이다. 하이브 파일의 첫 번째 블록은 베이스 블록(Base block)이라고 하며, 해당 파일의 전체적인 정보를 담고 있다. 빈은 연속된 블록을 차지할 수 있는 논리적인 데이터 단위로, 여러 개의 셀을 저장하는 칸막이라고 볼 수 있다. 셀은 레지스트리 정보를 가지는 가장 작은 단위로, 그 사이즈는 항상 8바이트의 배수이다. 베이스 블록은 regf, 빈은 hbin, 각각의 셀은 nk, sk, lf, lh, ri, li, vk등의 시그니처를 헤더에 가진다.

레지스트리 정보는 하이브 파일에 존재하는 셀들의 트리구조이다. 포인터로 이루어진 트리를 nk 셀이 가진 정보를 중심으로 표현하면 Fig. 2.와 같다.

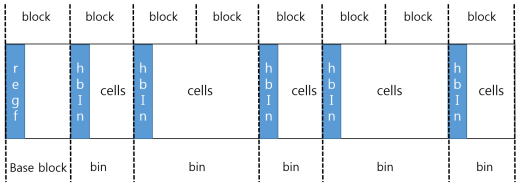


Fig. 1. Registry hive file

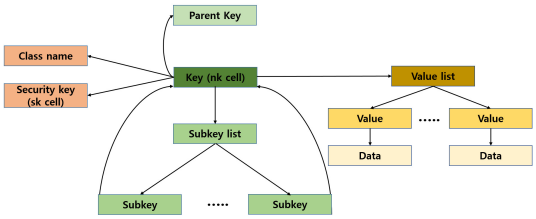


Fig. 2. Registry tree from nk cell

2.1.2 nk 셀

nk 셀은 각각의 키에 대응되는 셀로, 각기 다른 정보를 가진 셀들의 위치를 이용해 종합적인 한 개의 레지스트리 키를 구성한다. Fig. 3.은 nk 셀이 가지는 정보와 그 구조를 나타내고 있다. nk 셀의 구조상 Security key offset은 해당 셀이 참조할 sk 셀의 위치를 표기한다. 이를 통해 키에 접근, 수정할 수 있는 계정에 대한 권한설정을 한다.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	Cell Size				n	k	Flag		Timestamp							
10					Parent Key offset				Num of subkey				Num of subkey			
20	Subkey list offset				Subkey list offset				Num of values				Value list offset			
30	**Security key offset**				Classname offset				Max name length of subkeys				Max classname length of subkeys			
40	Max name length of values				Max value data size								Keyname len		Classname len	
50	Key name															
...																

Fig. 3. nk cell structure

2.1.3 sk 셀

sk 셀은 접근권한에 대한 설정 내용을 Windows security descriptor 구조에 담고 있다(Fig. 4.). SID[1], DACL[2], SACL[3]등의 정보를 이용하여 접근제한 설정을 구성하고 있다. 자세한 구조는 마이크로소프트사가 제공하는 SECURITY_DESCRIPT OR structure를 참고하도록 한다[4].

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	s	k			Previous SK offset				Next SK offset				Reference count			
10	Security descriptor Size				Windows security descriptor											

Fig. 4. sk cell structure

1) SID: Security Identifier
2) DACL: Discretionary Access Control List
3) SACL: System Access Control List

2.2 키 삭제 시 하이브 파일 변화

운영체제는 하이브 파일을 인식할 때 파일의 유효성을 검사하고, 정상적인 하이브 파일이 아니라면 받아들이지 않는다. 특히, 컴퓨터 부팅 시에 중요 하이브 파일이 제대로 인식 되지 않으면 부팅에 실패하게 된다. 따라서 하이브 파일을 유효하게 인식하도록 변조해야만 하고, 이를 위해서는 regedit에서 키를 삭제할 때 하이브 파일내에 생기는 변화를 알아야 한다[5-7].

하이브 파일에서, 사용하는 셀과 삭제된 셀을 구분하는 가장 큰 단서는 셀의 크기부분이다. 모든 셀은 구조상에 자신이 할당하고 있는 바이트 크기를 가지고 있는데, 이를 2의 보수[4]를 취해 음수 표기한다. 예를 들어 어떤 셀의 크기가 0x20바이트라면, 2의 보수를 취한 0xE0가 셀의 크기로 표기된다. 그리고 셀이 삭제되면 2의 보수가 아닌 양수로 바뀐다.

nk 셀은 Parent key offset, Value list offset, Class name offset, sk cell offset, Subkey list offset등 여러 가지 정보를 담고 있다. regedit에서 키를 하나 삭제하면 하이브 파일에서는 그에 해당하는 nk 셀이 삭제되는데, 이때 Subkey List offset과 sk cell offset은 0xFFFF FFFF로 덮어씌워진다. 따라서 삭제된 nk 셀이 어떤 sk 셀을 참조하고 있었는지는 알 수 없다. 자신을 참조하는 nk 셀이 삭제되면 sk 셀에서는 reference count가 1 줄어든다. Fig. 5.는 삭제된 nk 셀이 하이브 파일의 비할당 영역에서 가지고 있는 정보를 트리로 표현한 것이다.

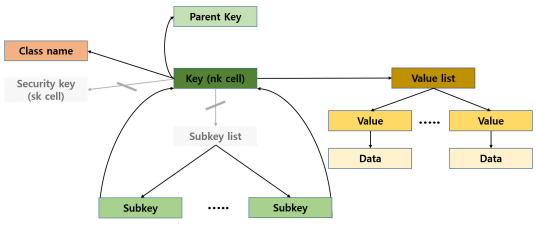


Fig. 5. Remaining data in deleted nk cell

4) 2의 보수: 어떤 수의 2진 표기에서 0, 1을 반전시키고 1을 더한 것이다. 컴퓨터에서 음수 표기를 하는데 많이 사용된다.

III. 레지스트리에 대한 접근권한 위·변조 기술

3.1 실험방법

본 절에서는 하이브 파일을 변조하고 테스트하기 위해서 필요한 환경과 방법을 제시한다.

3.1.1 하이브 파일 추출

하이브 파일은 부팅 이후에 항상 활성화 되어있으므로 일반적으로는 복사, 수정이 불가하다. 하지만 관리자 권한을 이용해 Rega, Encase, FTK 등의 툴을 사용하면 활성화된 시스템의 하이브 파일을 추출할 수 있다. 본 논문에서는 Rega를 사용하거나 부팅디스크로 부팅한 뒤 원하는 하이브 파일을 복사하였다.

regedit에서 '내보내기' 기능으로 특정 레지스트리 키를 하이브 파일로 만들어주는데, 이것은 해당 키에 대한 정보로 하이브 파일을 생성하는 것이다. 따라서 실제로 시스템이 사용하고 있는 하이브 파일과는 다른 파일이다. 이 기능을 사용해 regedit에서 수정한 레지스트리의 내용이 하이브 파일에 어떻게 적용되는지 빠르게 확인할 수 있었다.

일단 하이브 파일을 추출하고 나면 접근제어 설정과 상관없이 하이브에 저장된 레지스트리 키 트리를 모두 열어 확인할 수 있다. 본 논문에서는 하이브 파일의 레지스트리 트리 정보를 확인하는 툴로 Registry Viewer, OS Forensics을 사용하였다.

3.1.2 수정된 하이브 파일 테스트

본 논문에서는 하이브 파일이 운영체제에서 적용되는 것을 확인하기 위해 regedit에서 하이브 파일을 '가져오기' 하였다. 이 방법은 하이브 파일의 유효성과 수정에 대한 변화를 바로 확인할 수 있어 효율적이다. 하지만 실제 시스템에서 사용하는 레지스트리 정보가 바뀌어 운영체제에 예상치 못한 문제가 발생할 수 있다.

하이브 파일 자체를 교체했을 때 부팅에 끼치는 영향을 알기 위해서는 부팅디스크를 사용해 부팅한 뒤 하이브 파일을 교체한다. 그 후 원래대로 부팅하여 하이브 파일이 어떻게 적용되었는지 확인한다. 중요 하이브 파일이 유효하지 않다면 부팅이 되지 않는다.

3.2 nk 셀 수정

각각의 키에 대한 접근제한 속성은 키에 해당하는 nk 셀이 참조하는 sk 셀의 설정에 따라 결정된다. 때문에 nk 셀의 sk offset을 하이브 파일내의 다른 sk 셀의 위치로 바꾼다면 접근제한 설정이 바뀔 것을 기대할 수 있다.

아래 그림은 SAM 하이브 파일에서 확인할 수 있는 두 개의 nk 셀이다. 위쪽은 루트 키에 해당하는 nk 셀이며 아래쪽은 루트 키 아래의 SAM 키에 대항하는 nk 셀이다. Fig. 6에서 두 개의 음영 부분은 각각의 nk 셀에서 sk offset을 의미하는 부분으로, 파일의 해당하는 곳에 sk 셀이 위치한다.

아래의 두 실험은 SAM 하이브 파일에 대한 실험이다. 먼저 3.2.1에서는 nk 셀 헤더의 sk offset를 0이나 임의의 오프셋으로 대체할 때 하이브 파일을 정상 인식하는지 실험하였다. 3.2.2에서는 다른 sk 셀로 참조가 변경되었을 때 장애가 일어나지 않는지에 대해서 실험하였다.

3.2.1 nk 셀의 sk offset을 0으로 변조

SAM 하이브 파일의 루트 키 아래의 SAM 키는 시스템 계정만 접근이 가능하도록 설정되어있다(Fig. 6.의 아래쪽 키). 하지만 그에 해당하는 nk의 sk offset을 0으로 변경(Fig. 7.의 음영부분)하

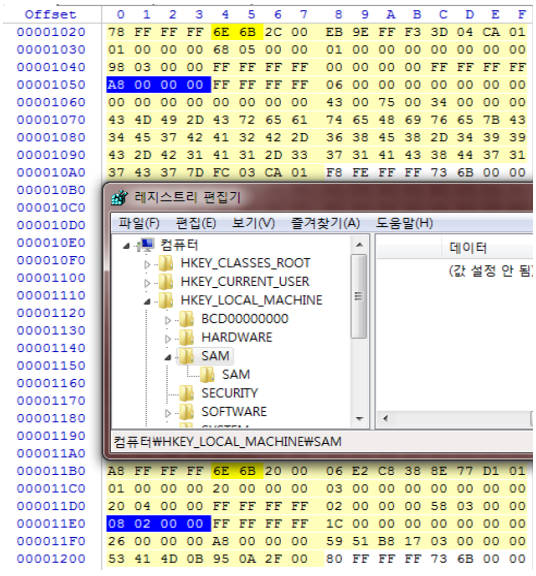


Fig. 6 . SAM Hive file

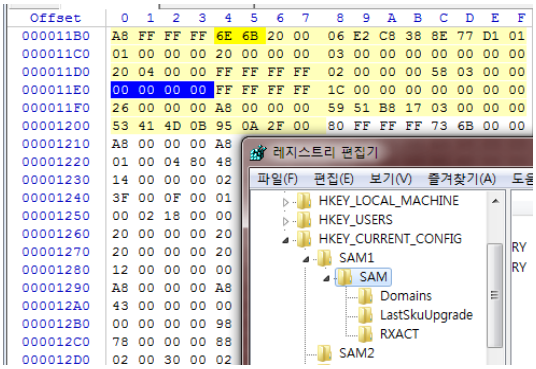


Fig. 7. Modifying sk offset to 0 (SAM1)

고 난 뒤 레지스트리 편집기에서 변조된 하이브 파일을 불러내 확인해보면(Fig. 7.의 SAM1 키) 아래의 키 목록을 확인할 수 있다.

이를 통해 sk offset에 의미 없는 값이 들어갈 때는 운영체제가 하이브 파일의 문제를 인식하지 못하며, sk의 옵션이 적용되지 않고 접근을 허가하는 것을 알 수 있다. 이를 이용하면 어떤 접근권한이 설정되어 있는 키든 접근 가능하게 하이브 파일을 변조할 수 있다.

위 실험에서 실제로 sk 셀을 참조하는 nk 셀의 개수는 달라지지만 sk 셀 구조상에 존재하는 reference count 정보는 수정하지 않았다. 그러므로 reference count가 하이브 파일 유효성과 무관한 것을 알 수 있다.

3.2.2 nk 셀의 sk offset을 다른 sk 셀의 offset으로 변경

SAM 하이브 파일의 루트 키는 접근제한이 비교적 약하게 설정되어 있어 regedit에서 열람이 가능하다(Fig. 6.의 위쪽 키). 하지만 루트 키에 해당하는 nk의 sk offset을 강한 접근제한 설정을 가진 sk 셀의 offset으로 변경하자 계정에 대한 접근제한이 걸려 열람할 수 없게 된다(Fig. 8.의 SAM2 키). 따라서 하이브 파일내에 강력한 접근제한 기능을 가진 sk 셀이 존재한다면 nk 셀의 sk offset(Fig. 8.의 음영부분)을 해당 sk 셀의 offset으로 바꾸는 것으로 키에 대한 접근을 막을 수 있다.

3.2.3 외부 sk 셀로 참조 변경

위에서 소개한 방법들은 하이브 파일내에 이미 존재하는 sk 셀을 이용한다. 하지만 특정 하이브 파일

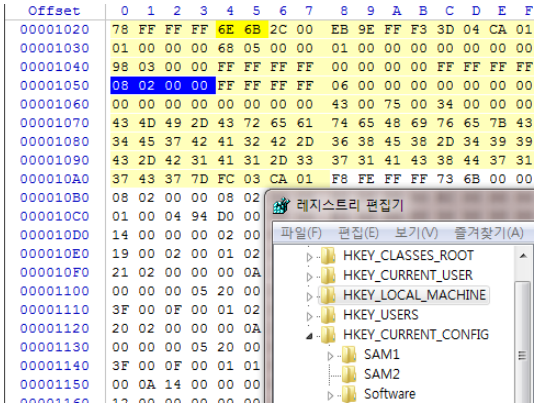


Fig. 8. Modifying sk offset to another sk cell's offset (SAM2)

에서 원하는 접근제한 설정을 가지는 sk 셀을 찾는 것은 번거로울뿐더러 존재하지 않는 경우도 있다. 아래 실험에서는 sk 셀을 만들거나 다른 하이브 파일에서 가져와 특정 키에 대한 접근제한을 조작할 수 있음을 보인다.

하이브 파일구조에서 빈은 자신에게 할당된 셀 이후에 남는 부분의 크기를 마지막 셀 뒤에 표시한다. 하이브 파일 마지막 부분에 셀을 추가한 뒤 이 부분을 적절히 수정하면 새로운 하이브 파일을 구성할 수 있다. Fig. 9.는 실험용 TEST 하이브 파일의 마지막 부분으로, 음영은 빈의 남은 공간을 표시하고 있다.

SAM이나 SECURITY 하이브 내에는 접근제한 설정을 강하게 가진 sk 셀이 존재한다. Fig. 6.에서

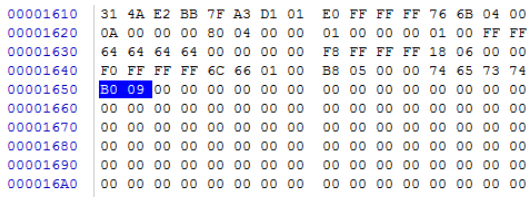


Fig. 9. End of Hive File

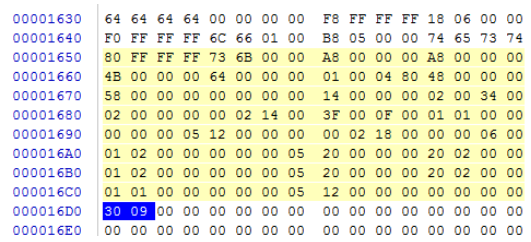


Fig. 10. Adding sk cell on End of Hive File

이러한 sk 셀을 복사하여 위의 TEST 하이브 파일 끝부분에 추가하였다(Fig. 10.의 음영부분).

위 그림은 접근제한 설정을 강하게 가진 sk 셀을 복사하여 SYSTEM 하이브 파일에 적절히 적용한 상태이다. 이 sk 셀의 offset을 특정 nk 셀의 sk offset에 할당한 결과는 Fig. 11.과 같다.

변조 전의 하이브 파일아래에는 여러 개의 test 키를 확인할 수 있다(Fig. 11.의 TEST HIVE). 반면에 추가된 sk 셀을 루트 키가 참조하도록 변조한 뒤 regedit에 하이브 파일을 등록해 보면 루트 키 아래의 test 키들을 열람할 수 없다(Fig. 11.의 TEST HIVE add KEY). 이는 새로운 셀을 만들거나 다른 하이브 파일의 셀을 복사하여 원하는 하이브 파일에 적용시킬 수 있다는 것을 의미한다.

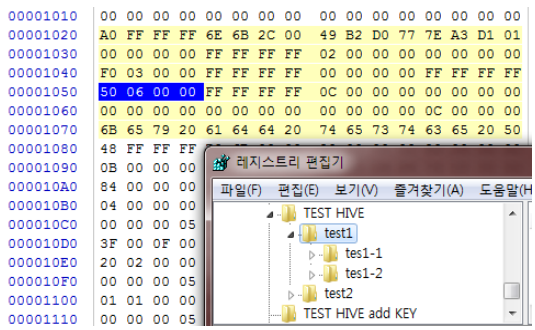


Fig. 11. Root Key Referencing new sk cell

3.3 sk 셀 수정

nk 셀을 수정하는 방법으로는 한 개 키에 대한 접근제한 설정을 바꿀 수 있었다. 하지만 sk 셀 자체를 수정한다면 해당 sk 셀을 참조하던 모든 nk 셀의 설정을 단번에 바꿀 수 있다는 장점이 있다.

3.3.1과 3.3.2에서는 SAM 하이브 파일의 강력한 접근제한 설정을 가진 sk 셀(Fig. 12.)을 이용해 실험을 수행한다.

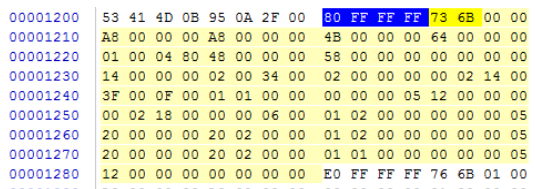


Fig. 12. sk cell in SAM

3.3.1 sk 셀 삭제

하이브 파일내의 특정 sk 셀을 삭제하여, 해당 sk 셀을 참조하는 모든 키의 접근제한을 없앤다. regedit에서 키를 삭제할 때 생기는 하이브 파일의 변화에 대해 2.2절에서 논의하였다. 하이브 파일의 셀 삭제 알고리즘에 따라 sk 셀 크기를 나타내는 상위 4바이트에 2의 보수를 취해주면 해당 셀을 삭제된 것으로 인식한다. 예를 들어 Fig. 12.의 음영부분에 2의 보수를 취해 Fig. 13.의 음영부분과 같이 고친다.

이 방법으로 하이브 파일의 sk 셀을 모두 삭제된 셀로 처리하면 해당 하이브 파일내의 모든 키에 대한 접근제한이 풀리게 된다. Fig. 6.에서 SAM키의 서브키를 확인할 수 없는데 비해, sk 셀을 삭제 한 뒤에는 Fig. 13.과 같이 서브키를 확인할 수 있다. 이를 이용하면 일반계정으로 접근할 수 없던 레지스트리의 숨겨진 정보를 참조하는데 사용할 수 있다.

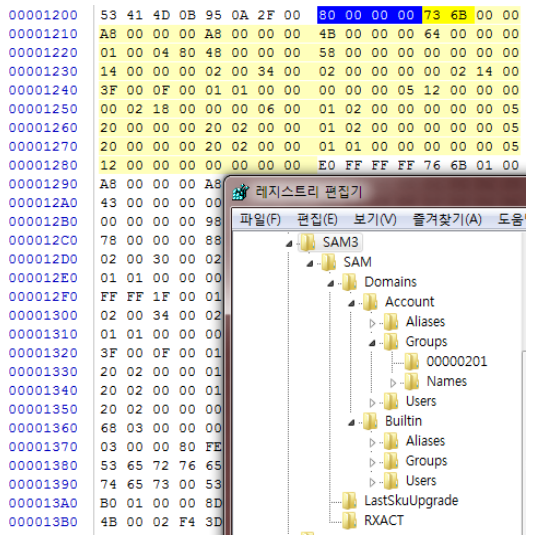


Fig. 13. SAM hive when sk cell is deleted(SAM3)

3.3.2 sk 셀 대체

하이브 파일의 sk 셀을 다른 셀로 치환하는 방법으로, 특정 sk 셀을 참조하는 모든 키의 접근통제 수준을 제어할 수 있다. 원하는 접근제한 설정을 가지는 sk 셀을 선정하여 하이브 파일의 sk 셀을 앞에서부터 덮어씌운다. 이때, 덮어씌우고 남는 부분은

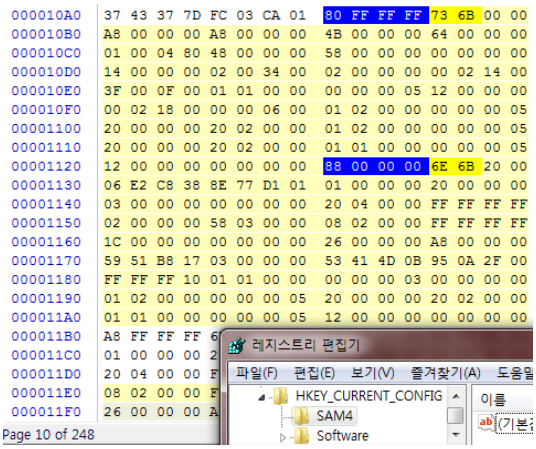


Fig. 14. Replacing sk cell

삭제된 셀로 보이게 해야만 하이브 파일이 정상적으로 인식된다.

SAM 하이브의 루트 키는 기본적으로 접근제한이 걸려있지 않다. 이 루트 키에 해당하는 nk 셀이 참조하던 sk 셀을 접근제한 설정의 sk 셀로 대체한다(Fig. 14.). 이때, 셀의 남는 크기를 적당한 nk 셀로 메꾸어준다. 실험에서는 0x88바이트가 남았다(Fig. 14.의 음영부분).

이 방법은 원래 존재하던 sk 셀의 크기가 대체하고자 하는 sk 셀 크기보다 커야만 적용가능하다. 그렇지 않은 경우 대체할 sk 셀에 뒤따르는 셀의 영역을 침범하기 때문이다. 접근통제 설정을 강하게 가진 sk 셀은 비교적 크기가 작기 때문에 이 방법이 적용가능하다. 만약 원래 존재하던 sk 셀의 크기가 적용하고자 하는 sk 셀 크기보다 작다면, 3.2.3에서 소개한 방법을 이용해야만 하이브 파일내의 정보 손상 없이 접근제한 설정을 바꿀 수 있다.

IV. 활용 방안

악의적인 사용자는 레지스트리 키를 지워 악성 프로그램 정보나 자신의 행동 기록을 지울 수 있지만, 삭제된 키와 값에 대한 정보를 복구하는 기술은 이미 발표된바 있다[5-7]. 공격자는 이런 복구기술에 대한 안티 포렌식 기술로 하이브 파일의 비할당 영역의 정보를 지우는 방법을 사용할 수 있다. 그러나 현재까지 연구된 포렌식 기술로는 접근권한을 가지지 않은 레지스트리 정보를 수정하거나 사용할 수 없다.

프리패치나 이벤트 로그, 휴지통과 같은 포렌식 관점에서 중요한 운영체제 아티팩트는 레지스트리에 그 설정을 저장한다. 이런 민감한 설정들은 접근제한으로 인해 일반 사용자 권한으로는 수정이 불가하다. 강한 접근제한이 걸려있는 레지스트리 정보를 일반사용자가 수정할 수 있는 방법으로는 하이브 파일내의 해당하는 값을 직접 변조하여 적용시키는 것이 있다. 하지만 하이브 파일을 구조적으로 분석하고 정확한 위치의 정보를 유효하게 수정하는 것은 현실적으로 어려운 점이 많다.

실질적인 공격시나리오는 부팅디스크로 부팅 후 하이브 파일을 추출, 조작하여 시스템에 적용시키는 것이다. 부팅디스크를 사용하는 공격자는 데이터 탈취, 시스템 조작이 모두 가능한 상태라고 볼 수 있기 때문에, 위의 공격 가정은 비교적 강하다. 이는 공격 시점에 데이터를 탈취하고 악성 행위를 일으키는 공격자에게는 본 논문에서의 공격이 의미가 크지 않음을 뜻한다. 하지만 일반 사용자 권한으로 레지스트리 정보를 모두 수정, 활용할 수 있다는 점에서 본 논문의 공격은 치명적일 수 있다. 공격자가 하이브 파일을 조작해 적용하면 일반 사용자 계정으로도 시스템의 감사정책을 바꾸거나 프리패치, 이벤트로그 생성을 막을 수 있다. 따라서 공격 이후 컴퓨터 사용 시에 포렌식 관점에서 중요한 증거를 남기지 않고 행동할 수 있다. 또한 특정 레지스트리에 대한 접근제한을 없앤다면 악성코드가 SECURITY 하이브의 사용자정보와 같이 민감한 정보를 사용할 수 있게 된다. 반대로 어떤 키를 참조할 수 없도록 접근제한을 상승시킨다면 이후 정당한 사용자가 사용하는 프로그램이 기능장애를 일으킬 수도 있다. 사용자는 이런 조작을 당하더라도 시스템이 정상작동 하므로 공격당한 사실을 알아채기 힘들다는 점에서 본 논문의 공격은 의미를 더한다.

공격자가 레지스트리 접근권한을 변경하는 의도는 크게 두 가지로 볼 수 있다. 첫 번째로는 특정 계정의 키에 대한 접근을 제한하여 시스템 장애를 일으키는 것이다. 이 경우엔 키의 sk 셀 참조를 특정 sk 셀로 바꿔야한다. 두 번째는 키에 대한 접근제한을 완화시켜 레지스트리 정보를 누구든 참조할 수 있게 하는 것이다. 이 경우엔 굳이 키의 sk 셀 참조를 정확히 바꿔줄 필요 없이 의미 없는 값(0)을 sk 셀로 참조하도록 한다. 이러한 방법들을 이용해 중요한 키에 대한 접근제한을 약화시켜 원하는 정보를 참조하거나 수정할 수 있다. 또한 키에 대한 접근제한을 강화시켜 시스템이 레지스트리를 참조하지 못하도록 방

Table 1. Experiment result and its application

Group	Experiment	Result	Application
Modifying nk cell	Modifying sk offset to 0	It removes access control of particular key.	One can change system set-up or get information of a key without proper authority.
	Modifying sk offset to that of another sk cell in the same hive file	It replaces access control of particular key with another sk cell in the same hive file.	One can change system set-up or get information by weakening access control of a key. Or one can make system problem by strengthening access control of a key.
	Modifying sk offset to that of another sk cell in a different hive file	It replaces access control of particular key with a new sk cell set-up.	
Modifying sk cell	Removing sk cell	It removes access control of all key referencing the sk cell.	One can change system set-up or get information from all key referencing the sk cell.
	Replacing sk cell	It replaces access control of all key referencing the sk cell with another sk cell set-up .	One can change system set-up or get information of a key without proper authority. Or one can make system problem by strengthening access ontrol of keys.

해할 수 있다. 조금 더 나아간다면 공격자는 원하는 레지스트리 키를 만들어 등록한 후 특정 sk 셀을 사용해 해당 레지스트리 정보를 숨길 수도 있다. 본 연구의 실험 내용, 결과 및 활용방안은 Table 1.에 정리하였다.

레지스트리 접근제한 설정에 조작이 이루어졌는지를 확인하기 위해서는 하이브 파일을 분석해야 한다. nk 셀이 참조하고 있는 sk 셀 offset이 의미 없는 값이라면 sk 셀 참조위치를 조작했을 가능성이 있다. 또한 sk 셀 구조의 reference count가 실제와 맞지 않는다면 그 또한 조작의 흔적이라고 볼 수 있다. 보통 시스템 운영에 중요한 하이브 파일은 내용이 서로 비슷하기 때문에 다른 시스템에서 추출한 하이브 파일과 비교하는 방법도 유용하다.

V. 결 론

본 논문은 하이브 파일을 변조하여 레지스트리 키에 대한 접근권한을 조작하는 방법을 제시하였다. 이를 통해 키에 대한 접근제한을 강화 또는 약화시킬 수 있다. 공격자는 이벤트 로그나 프리패치, 휴지통 등 포렌식 관점에서 중요한 프로그램이나 윈도우 아티팩트의 설정을 3장에서의 실험내용을 이용해 조작할 수 있다. 프리패치나 이벤트 로그가 정상적으로 작동하는지 매번 확인하는 사용자는 드물기 때문에, 이렇게 이루어진 공격은 사고가 일어난 뒤 포렌식 수사관이 윈도우 아티팩트 분석을 시행하기 전까지는 알아채기 힘들다. 따라서 포렌식 수사관은 이와 같이 조작된 하이브 파일을 구별해 낼 수 있어야 하며, 사용자는 시스템에 중요한 설정정보를 수시로 확인하여 피해를 막아야만 한다.

상용화된 하이브 파일 뷰어는 대부분 sk 셀의 정보는 사용하지 않고, 키와 값에 대한 정보만을 트리형식으로 나열한다. sk 셀을 참조하여 각각의 레지스트리 정보에 대한 접근권한 설정을 분석해 보여주는 툴이 개발된다면 하이브 파일 분석에 용이할 것이다.

하이브 파일 구조에는 해시값이나 CRC와 같이 무결성을 제공하는 장치가 부족하기 때문에 변조가 비교적 쉽다. 따라서 레지스트리 정보에 대한 접근권한 조작을 막기 위해서는 하이브 파일의 무결성을 보장

할 수 있도록 하이브 파일 구조 업데이트가 이루어져야 한다.

References

〔1〕 Derrick J. Farmer, "A Windows registry quick reference: for the everyday examiner", Forensic Focus, Oct. 2007.

〔2〕 Peter Morris, "The internal structure of the windows registry", Ph.D. Thesis, Cranfield University, Feb. 2009.

〔3〕 Timothy D. Morgan, "The Windows NT registry file format version 0.4", June. 2009.

〔4〕 Microsoft, Access Control Litsts, https://msdn.microsoft.com/ko-kr/library/windows/desktop/aa374872(v=v s.85).aspx

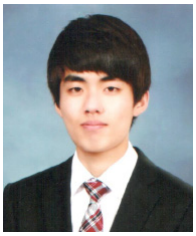
〔5〕 Timothy D. Morgan, "Recovering deleted data from the Windows registry", DFRWS, S33-S41, Sep. 2008.

〔6〕 Jolanta Thomassen, "Forensic analysis of unallocated space in windows registry hive files", Ph.D. Thesis, The University of Liverpool, Apr. 2008.

〔7〕 Zhenhua Tang, "Carving the Windows registry files based on the internal structure", ICISE, 2009 1st International Conference on. IEEE, Dec. 2009.

〈저자소개〉

김 한 기 (Hangi Kim) 학생회원
2016년 2월: 국민대학교 수학과 졸업
2016년 3월~현재: 국민대학교 금융정보보안학과 석사과정
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식



김 도 원 (Dowon Kim) 학생회원
2015년 2월: 국민대학교 수학과 졸업
2015년 3월~현재: 국민대학교 금융정보보안학과 석사과정
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식

김 종 성 (Jongsung Kim) 종신회원
2000년 8월/2002년 8월: 고려대학교 수학 학사/이학석사
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 정보보호 공학박사
2007년 2월: 고려대학교 정보보호대학원 공학박사
2007년 3월~2009년 8월: 고려대학교 정보보호기술연구센터 연구교수
2009년 9월~2013년 2월: 경남대학교 e-비즈니스학과 조교수
2013년 3월~현재: 국민대학교 수학과 부교수
2014년 3월~현재: 국민대학교 일반대학원 금융정보보안학과 부교수
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식