**정보보안기사(최종수정일 8/25)**

**4장 시스템 보안**

**12절 보안 운영체제**

보안 운영체제란 기존 운영체제 내에 시스템 보안 기능을 통합한 보안 커널을 추가로 이식한 운영체제이다. 기술적 핵심은 정보영역의 분리, 역할기반 영역분리, 최소권한 유지, 커널 레벨의 강제적 접근통제 등을 들 수 있다.

**보안 운영체제의 주요 제공 기능**

**보호대상:** 메모리, 보조기억장치 상의 파일 또는 데이터 집합, 메모리상에서 실행 중인 프로그램, 파일들의 디렉터리, 하드웨어 장치

**보호방법:** 물리적 분리(사용자별 하드웨어 분리), 시간적 분리(프로세스가 시간 당 하나만), 논리적 분리(프로세스에 논리적 구역 제한), 암호적 분리(내부 사용 정부를 외부에서 알 수 없도록).

나열된 방법들은 복잡도가 증가하는 순이다. 분리방법을 조합해 이용할 수도 있다.

**파일시스템 보호**

파일의 지나친 보호는 자원낭비, 지나친 공용은 보안성 상실을 야기한다. 다양한 파일 보호기법으로 균형을 찾아야 한다.

1. **파일의 이름 명명(Naming):** 다른 사용자의 파일 이름을 알 수 있는 기법이 없고 추측도 어렵다는 가정에 기초한다.
2. **패스워드:** 사용자마다 패스워드를 부여하는 방법.
3. **암호화:** 파일 내용을 암호화 해 누구나 공유할 수 있지만 인가된 사용자만 내용을 알 수 있게 한다.

**보안 운영체제와 보안커널**

**보안 운영체제의 보안 기능:**

1. **사용자 식별 및 인증**
2. **임의적(DAC, Discretionary Access Control-신분 기반 정책)/강제적(MAC, Mandatory Access Control-규칙기반정책) 접근 통제**
3. **객체 재사용(Object Reuse) 보호:** 재사용 가능한 저장매체에서 한 번 저장된 데이터가 삭제되지 않고 존재하는 경우가 많다. 이를 통한 비밀 데이터 유출(객체 재사용)을 막기 위해 재할당되는 모든 기억장치 공간을 깨끗이 지우는 것이 객체 재사용 보호이다.
4. **완전한 조정:** 임의적/강제적 접근통제가 효과적이기 위해서는 모든 접근을 통제해야 한다. 따라서 보안 운영체제는 모든 접근을 통제하는 완전한 통제를 수행해야 한다.
5. **신뢰 경로(Trusted Path):** 패스워드 설정, 접근 허용 변경 등과 같은 보안 작업 수행 시에 사용자의 안전한 통신 경로를 제공할 수 있어야 한다.
6. **감사 기록(Audit Logging) 및 기록 축소:** 모든 보안 관련 사안은 감사 기록부(Audit Log)에 반드시 기록되어야 하고 감사 기록부는 명백히 보호되어야 한다.

**보안 커널(Security Kernel):** 신뢰 컴퓨팅 기반TCB(Trusted Computing Base) 내에 있는 하드웨어, 소프트웨어, 펌웨어로 구성된다. 참조 모니터 개념을 구현하고 집행. 보안 커널은 주체와 객체 사이의 모든 접근과 기능을 중재한다. 보안 커널 자체가 TCB의 핵심이다.

참조 모니터 개념을 수행하는 프로세스를 위해 분리가 제공되고 프로세스는 변조 방지 기능이 있어야 한다. 모든 접근에 대해 실행되어야 하고 우회가 불가능해야 한다. 따라서 보안 커널은 완전하고 오류가 없어야 한다. 또한 완전하고 포괄적인 방식으로 충분히 시험되고 확인되어야 한다. 보안 커널의 도식은 반원형과 원형(허니웰의 멀틱스)이 있다.

**신뢰 컴퓨팅 기반(TCB, Trusted Computing Base):** 하나의 컴퓨터 시스템(H/W, F/W, S/W) 내의 모든 보호 메커니즘의 총체이다. 시스템과의 조화를 통해 보안 정책을 적용할 책임을 갖는다. 신뢰된 경로는 사용자, 프로그램, 커널과의 통신채널을 말한다. TCB는 어떤 경우에도 이 채널이 훼손되지 않도록 보호 기능을 제공한다. **보안경계(Security Perimeter)**는 시스템의 구성요소 중 신뢰/비신뢰 구성요소의 경계이다. 물리적 실체가 아니라 신뢰/비신뢰 구성요소 사이에 선을 긋기 위해 운영체제에서 사용하는 개념적 선이다. TCB는 안전을 위한 건축설계 기준과 같다고 생각하자.

**참조모니터(Reference Monitor):** 주체의 객체에 대한 접근통제를 담당하는 추상 머신이다. 다음 세 가지 규칙을 가진다.

* 반드시 부정조작이 없어야 한다 – Must be tamperproof
* 항상 무시되지 않고 호출되어야 한다 – Always invoked
* 모든 동작을 항상 분석과 테스트를 통해 확인할 수 있어야 한다 – Must be Verifiable

주체가 참조 모니터를 거치지 않고 객체에 직접 접근하는 경우가 참조 모니터를 위반하는 경우이다. 참조 모니터는 보안커널 데이터베이스(SKDB)를 참조해 객체에 대한 접근허가 여부를 결정한다.

**신뢰 플랫폼 모듈(TPM)**

Trusted Platform Module이란 신뢰 컴퓨팅 그룹 TCG라는 산업체 컨소시엄에서 표준화된 개념이다. TC에서 핵심이 되는 하드웨어 모듈을 말한다.

**TPM의 특징:** 신뢰 컴퓨팅의 가장 하위에 위치한다. 훼손 방지(Tamper Protected)가 필수이기에 하드웨어칩으로 구현하는 게 일반적이지만 소프트웨어로 구현하기도 한다. 칩으로 구현하면 소프트웨어 공격과 물리적 도난에도 정보의 탈취가 어렵다는 장점이 있다. TPM에는 신뢰 관련 연산이 있다. 1. 암호화 키의 생성/저장 2. 패스워드 저장 3. 무결성(플랫폼/프로그램)의 검증을 위한 측정값의 저장 4. 디지털 인증서 관련 신뢰 연산 등이 있다.

**TPM의 기본 서비스:** 인증된 부트 서비스(단계별 부트, 적재될 때마다 승인된 버전임을 보장), 인증 서비스(사설키를 사용해 설정 정보에 서명함으로써 디지털 인증), 암호화 서비스

**TPM 함수:** 다음과 같은 구성요소로 이루어진다. I/O, 암호화 보조프로세서, 키 생성(RSA 공유/사설키 쌍과 대칭키), HMAC 엔진(인증 프로토콜에서 쓰는 알고리즘), 무작위 숫자 발생기(RNG, Random Number Generator), SHA-1 엔진, 전원 탐지, Opt-in(고객 의도대로 TPM을 활성/비활성화할 수 있는 안전한 메커니즘), 실행 엔진(I/O포트에서 받은 TPM명령어 실행), 비휘발성 메모리, 휘발성 메모리

**13절. 클라이언트 보안**

**악성 소프트웨어(악성 코드)**

**바이러스:** 자기복제 O, 파일/부트섹터 등 감염 대상 필요, 사용자가 감염된 파일 옮김, 시스템 및 파일 손상을 목적으로 함

**트로이목마:** 자기복제 X, 정상 프로그램으로 위장 or 정상 프로그램에 코드로 삽입, 사용자가 실행시킴, PC 성능 저하나 통제권 획득을 목적으로 함

**웜:** 자기복제 능력이 매우 강함, 독자적으로 존재하며 네트워크를 통해 스스로 전파한다. 네트워크 성능 저하를 목적으로 한다.

**악성 소프트웨어 분류:**

1. **독립형과 기생형:** 호스트를 필요로 하는지 아닌지 여부로 구분. 전자는 기생형. 독립적으로 존재할 수 없다. 바이러스, 논리폭탄, 백도어 등이 해당된다. 후자는 자체적으로 구동 가능한 프로그램으로 운영체제가 스케줄 해 구동할 수 있다. 웜과 좀비가 해당
2. **자기 복제 여부:** 바이러스성 – 웜, 바이러스 / 비-바이러스성 – 트로이목마, 백도어

**바이러스:** 타 프로그램을 변형시켜 감염시키는 프로그램. 자신을 변형시켜 실행 가능하기도 하다. 변형이란 원래 프로그램에 루틴을 주입해 바이러스 프로그램의 복제를 만들어내는 걸 말한다. 바이러스는 자신을 완벽히 복제할 수 있는 비법을 가진 명령 코드를 가진다. 또 전형적으로 프로그램이나 실행 콘텐츠의 전송자 속에 내장된다. 세대별로 분류하면 다음과 같다.

1. **제1세대 원시형 바이러스(Primitive Virus):** 단순하고 분석이 쉬운 바이러스. 코드의 변형이나 변화 없이 고정된 크기를 가지며 기억장소에 상주해 부트영역이나 파일을 감염시킨다.
2. **제2세대 암호화 바이러스(Encryption Virus):** 백신 프로그램이 진단할 수 없도록 바이러스 프로그램의 일부 또는 대부분을 암호화시켜 저장한다. 실행 시작점 부분에 위치하는 암호해독 데이터는 항상 일정하다.
3. **제3세대 은폐형 바이러스(Stealth Virus):** 기억장소에 존재, 감염된 파일의 길이가 증가하지 않은 거처럼 보이게 한다. 백신 프로그램이 감염된 부분을 읽으려 하면 감염되기 전의 내용을 보여줘 바이러스가 없는 것처럼 속인다. 이렇게 스스로 모습을 바꾸는 바이러스는 다형성 바이러스라고 한다.
4. **제4세대 갑옷형 바이러스(Armor Virus):** 백신 프로그램으로부터 숨기보다 여러 단계의 암호화와 다양한 기법을 동원해 바이러스 분석을 어렵게 하고 백신 개발을 지연시킨다.
5. **제5세대 매크로 바이러스:** MS 오피스, Visio, AutoCAD 등 Visual Basic Script를 지원하는 다양한 프로그램에서 활동해 현재 등장하는 바이러스 중 가장 높은 비중을 차지한다.

**매크로 바이러스:** 매크로 바이러스는 플랫폼과 무관하며 쉽게 퍼지고(이메일) 실행 파일(COM, EXE)을 다룰 때 주의를 덜 하기 때문에 피해가 크다. 특히 어도비 사의 PDF에 자바스크립트 같은 다양한 스크립트를 포함하는 형태로 많이 유포된다.

**바이러스 방지책:**

1. **안티 바이러스:** 탐지, 식별, 제거를 통해 바이러스 예방 방지
2. **안티 바이러스 필터링 방법:** 대표적으로 virus signature와 behavioral virus scanning이 있다.

* **signature scanning:** 특정 바이러스만이 가진 시그내쳐를 찾아낸다. 이는 실행 가능한 코드(excutable code) 상에서의 이진 스트링, 스크립트에서는 명령어의 나열로 파악할 수 있다. 대부분의 안티 바이러스 프로그램들이 채택하는 방법이다.
* **behavioral virus scanning:** 바이러스가 수행 중에 어떤 행동을 보이는지 추적하는 방법. 시그니처 스캐닝에서 대처가 불가능했던 새 바이러스와 웜에 대한 대처 능력을 기를 수 있다.

1. **바이러스 예방:** 신뢰성 있는 업체에서 구매한 상업용 소프트웨어만 사용해야 하며 정기 바이러스 스캐닝, 백신 업데이트 등이 필요하다. 이때 Windows Script Host, ActiveX, VBScript, JavaScript는 비활성화한다.

**웜:** 자가 복제해 네트워크 연결을 통해 컴퓨터에서 컴퓨터로 복제본을 전송한다. 이메일 자체가 시스템에서 시스템으로 자신을 확산시키기 때문에 웜의 성격을 가지고 있다. 다른 시스템에 직접적인 영향을 미치지 않아 트로이목마와 구분되며 다른 프로그램에 기생하지 않는다는 점에서 컴퓨터 바이러스와 구분된다.

**웜 확산 모델:** 확산은 3단계로 이뤄진다. 감염된 호스트가 각각 두 개의 호스트를 감염시킨 뒤 다시 감염된 호스트들은 지수적으로 증가한다. 이후 고점을 지나면 바이러스가 서서히 퍼지며 종료 단계에 접어든다.

**웜의 실행:** 버퍼 오버플로우, 포맷 스트링, SQL 인젝션, PHP 인젝션 등 공격 가능한 시스템 취약점을 이용해 시스템의 접근 권한을 확보하고 자신을 실행시킨다. 접근 권한을 확보하면 웜을 백도어로 설치하거나 DDoS 공격에 사용할 수 있는 봇이나 좀비, 키로거 프로그램을 다운로드해 실행한다.

**웜 대응책:** 바이러스 대응책과 유사하다. 안티 바이러스와 네트워크 모니터링을 통해 기본적인 방어를 할 수 있다.

* **네트워크 기반 웜 방어:** 웜 모니터링 소프트웨어이다. 상호 연결된 LAN의 집합체로 구성된 사이트의 엔터프라이즈 네트워크에 두 가지 유형의 모니터링 소프트웨어가 필요하다.

1. **진입 모니터(Ingress Monitors):** 엔터프라이즈 네트워크와 인터넷 사이 경계에 위치한다. 경계 라우터나 외부 침입차단시스템이나 독립된 수동 모니터의 진입 필터링 소프트웨어의 일부이다.
2. **진출 모니터(Egress Monitors):** 나가는 트래픽에 대한 스캐닝 흔적이나 의심스러운 행동을 모니터링해 출처를 잡아내도록 설계되었다.

**트로이목마(Trojan Horse):** 정상 프로그램으로 가장하거나 정상 코드에 삽입된 형태로 다운/실행된다. 이후 권한을 획득해 해당 시스템을 통제하는 것을 목적으로 한다. 데이터 파괴를 목적으로 하기도 한다. 대표적으로 Back Orifice는 1998년 9월 CDC(Cult of the Dead Cow)에서 발표한 Win 95/98/2000 관리자용 도구이며 백도어로 이용하기 적합하도록 개발되었다.

**트로이목마 대응책:** 신뢰할 수 없는 파일을 설치/실행하지 않는다. 취약점 패치를 항상 최신 상태로 유지한다. 안티 – 트로이 목마 도구도 있으니 활용하면 된다.

**스파이웨어:** 민감한 정보를 수집해 주기적으로 원격지의 서버로 보내는 프로그램이다. 신원 도용, 스패밍, 사기 등과 같은 악의적 활동에 사용. 브라우저 사용 정보를 수집해 표적 광고를 보내는데 활용되기도 한다.

**기타 알아둬야할 악성 소프트웨어:** 논리폭탄, 모바일 코드, Exploit, Downloaders, Auto-rooter, Kit, 스패머 프로그램, 플러더, Rookit, 공격 킷, 크라임웨어, 브라우저 하이재커, 다이얼러, 조크, Hoax

**인터넷 활용 보안**

**브라우징 보안:** 사이트에 따라 서로 다른 보안 요구사항이 필요할 때가 있다. 브라우저는 이를 관리하기 위해 다양한 보안영역을 설정하고 사용자가 정의할 수 있게 해야 한다.

**설정 가능한 영역:** 인터넷, 로컬 인트라넷, 신뢰할 수 있는 사이트, 제한된 영역

**쿠키:** 1994년 넷스케이프에서 처음 사용. 사용자가 웹 사이트에 방문할 때 생기는 4KB 이하의 파일이다. 사용자와 웹 사이트를 연결해주는 정보가 담겨 서버가 클라이언트의 신원을 알 수 있게 한다.

**쿠키를 이용한 통신:**

1. 사용자가 웹사이트 방문시 웹사이트는 사용자의 컴퓨터에 쿠키를 만든다. 브라우저를 확인하는 ID 번호를 쿠키에 넣어 사용자의 컴퓨터에 저장하는 식이다.
2. 사용자가 해당 서버를 다시 방문하면 쿠키를 서버로 다시 전송하는 단계이다. 사이트는 신원이 확인된 사용자의 정보를 이용해 쾌적한 서비스를 제공할 수 있다.

**쿠키의 일반적인 용도:**

1. **사이트 개인화:** 아이디, 비밀번호, 성향을 파악할 수 있다.
2. **장바구니 시스템**
3. **웹 사이트 이용방식 추적**
4. **타겟 마케팅**

**쿠키의 구조:** 4개의 속성, 하나의 데이터로 구성된다. 서버는 응답에 Set-Cookie 헤더를 포함시켜 쿠키를 설정한다. 기본 구조는 다음과 같고 값을 추가할 경우 세미콜론을 구분자로 추가한다.

**Set-Cookie: name=value; expires=[Date]; domain=[Domain]; path=[Path]; [secure]**

* **유효기간:** 쿠키 데이터는 브라우저를 끄기 전까지 사용할 수 있지만 기간이 설정되어 있으면 브라우저를 재시작한 뒤에도 적용할 수 있다. 유효 기간이 지난 쿠키는 소멸된다. 파일이 지워지지 않더라도 이를 브라우저에서 읽을 수 없다.
* **패스:** 한 번 쿠키 데이터를 생성하면 쿠키 데이터를 생성한 웹 페이지만 해당 데이터를 읽을 수 있다. 하지만 Path 항목을 지정해주면 해당 Path에서는 그 쿠키 데이터를 공유할 수 있다.
* **도메인:** 패스 속성을 확장한 것이다. 패스가 하나의 사이트에서 쿠키를 읽고 쓰는 권한을 설정하는 것이면 도메인은 도메인 단위에서 쿠키 데이터를 읽고 쓰는 권한을 설정하게 된다.

**HTTP 쿠키 관련 보안 속성:**

* **httpOnly 속성:** Set-Cookie 응답 헤더에 설정하는 속성으로 클라이언트에서 스크립트를 통해 해당 쿠키에 접근하는 것을 차단해주는 속성. 세션 쿠키(세션ID 보유)를 탈취하는 XSS(Cross Site Script)공격에 대응하기 위해 사용한다.
* **secure 속성:** Set-Cookie 응답 헤더에 설정하는 속성으로 클라이언트에서 HTTPS 통신일 경우에만 쿠키를 전송하고 HTTP일 경우 전송하지 않는다. 전송구간 암호화를 통해 평문 전송을 막기 위함이다. 기밀성 보장 목적

**쿠키에 대한 오해:**

* **바이러스 전파:** 쿠키는 텍스트 파일로 실행되지 않는다. 4KB 이하의 파일로 바이러스를 전파할 수도 없다.
* **사용자 컴퓨터에 피해 입히기:** 읽기/쓰기를 할 수 없는 텍스트이기에 불가능하다.
* **다른 웹사이트에서 읽기:** 이름, 값, 유효기간, 도메인, 경로 등으로 이루어져 저장된 도메인 이름을 가지고 있는 사이트에서만 유효하다.

**쿠키 보안 취약점:**

1. **XSS(Cross-Site Scripting) 공격:** 서버사이드 스크립트가 아닌 자바스크립트나 비주얼베이직 스크립트 등이 클라이언트의 컴퓨터에서 실행된다는 점을 이용한 공격이다. document.cookie라는 명령어는 사이트에서 쿠키값을 활용해 다양한 서비스를 제공할 수 있도록 하지만 공격자들은 쿠키값을 유출하기 위해 사용한다.
2. **스니핑 공격을 이용:** 쿠키값을 암호화하지 않으면 스니핑을 통해 쿠키를 탈취할 수 있다.
3. **공용 PC에서 쿠키값 유출:** 쿠키는 하드에 저장되어 접근이 열려있기 때문에 같은 컴퓨터 이용시 유출될 수 있다.

**세션:** 개별 클라이언트 상태정보를 서버에 저장하는 기술이다. 개별 클라이언트를 식별하기 위해 세션 아이드를 부여하고 세션 아이디를 세션 쿠키를 통해 주고받는다. HTTP 세션 하이재킹은 공격자가 사용자의 세션 ID를 탈취해 사용자인 것처럼 위장 접근하는 공격이다.

**안전한 세션 ID관리:** 안전한 서버에서 생성 후 사용해야 하며 최소 128비트 길이로 생성되어야 한다. 또 난수 알고리즘을 적용해 예측 불가능한 값이어야 한다. URL Rewrite 기능 사용시 아이디가 URL에 노출될 수 있어 사용하지 않도록 설계하고 장시간 접속시 세션 ID 노출 위험이 점점 커지므로 주기적으로 세션 ID를 재할당하도록 설계한다.

**14절. 윈도우 서버 보안**

**윈도우 시스템의 구조**

윈도우 운영체제는 5개의 링 기반이다. 하드웨어 – HAL(Hardware Abstraction Layer) – 마이크로 커널 – 각종 관리자 – 응용 프로그램 순으로 구분한다. 관리자 계층 까지를 커널 모드로 구분하고 마지막 응용 프로그램 계층을 사용자 모드로 구분한다. 드라이버 개발자와 하드웨어 개발자는 윈도우에서 제시한 표준인 HAL만 따르면 된다. HAL은 하드웨어와 소프트웨어 간의 통신을 번역한다.

**윈도우의 특징:** GUI 환경, Plug & Play, 단축 아이콘(Short cut)/바로가기, 멀티태스킹, OLE(Object Linking Embedding), 네트워크 기능 향상, 다중 모니터 지원, 정보의 전송 통합

**윈도우 파일 시스템:**

1. **FAT16:** DOS와 Windows 95에서 지원. 2GB를 한 파티션으로 설정 가능. 테이블의 기본 크기는 16비트. 하드디스크에 파일을 담는 최소 단위인 클러스터가 32KB이다. 전체 가능 용량은 2GB(2^16 \* 32 \* 1024 = 2^31 = 2 \* 1024 \* 1024 \*1024)이다.
2. **FAT32:** Windows 95 OSR(OEM Service Release)2에서 처음 도입. 총 32비트 개의 클러스터를 가질 수 있다. 현재도 종종 사용되며 호환성이 좋아 다른 운영체제에 정보를 옮길 때 유용하다. 하지만 접근제어를 설정할 수 없어 보안과는 거리가 멀다.
3. **NTFS:** Windows NT부터 적용되기 시작 NTFS(New Technology File System)으로 접근 제어가 적용되기 시작했다. 파일시스템의 결함 관리(HotFixing)에 있어 FAT나 FAT32보다 훨씬 좋은 성능을 보이고 보안적으로도 더 훌륭하다. 개별 폴더와 파일에 사용 권한 설정, 폴더 암호화(복사나 이동시에도 암호화 유지), 감사 기능(Auditing – 언제 누가 해당 파일을 열었는지, 혹은 접근에 실패했는지) 등의 기능 제공. NTFS는 구조적으로 다섯 부분으로 나뉜다. MBR(Master Boot Record), VBR(Volume Boot Record), MFT(Master File Table), 시스템 파일, 파일 영역이 그것이다.

* **MBR:** 파티션 생성 시 물리적 디스크의 첫 섹터에 위치하는 512바이트 영역. 부트 코드, 파티션 테이블, 시그니처로 구성되어 있다. 운영체제가 부팅될 때 BIOS에 의해 POST(Power On Self Test)과정을 마친 후 MBR의 부트 코드를 호출하고 부트 코드는 부팅 가능 파티션을 찾아 해당 파티션의 VBR의 부트 코드를 호출해 운영체제가 기동한다.
* **VBR:** 부팅을 위한 기계어 코드와 볼륨 및 클러스터의 크기, MFT의 시작주소 등 설정 정보를 담고 있다. 파티션의 가장 마지막 섹터에 VBR백업본을 가지고 있다.
* **MFT:** 볼륨/파일시스템에 존재하는 모든 파일과 디렉터리에 관한 정보를 담고있는 테이블. 가장 중요한 영역. MFT 엔트리의 집합으로 구성되어 있다. 각 엔트리는 하나의 파일 또는 디렉터리에 대한 정보를 담고 있는 자료구조이다. 파일의 위치, 시간 정보, 크기, 파일 이름 등 다양한 속성 정보로 구성
* **시스템 파일:** MFT를 비롯해 파일 복사나 삭제 과정 중 시스템이 오류를 일으킬 때 이를 복구하기 위한 로그 파일, 디스크 볼륨 등 디스크 자체에 대한 정보를 담는 파일을 저장
* **파일 영역:** 각 파일에 대한 실제 데이터 저장

1. **디스크 파괴형 악성코드의 특징:** MBR 또는 VBR을 훼손해 컴퓨터가 정상 부팅을 못 하게 만든다. MBR의 모든 값을 특정 문자나 랜덤값으로 덮어씌워 운영체제 부팅을 할 수 없게 하거나 VBR 영역을 훼손해 파일시스템 복구가 불가능하도록 만든다. MBR 손상의 경우 부트 코드를 정상적인 디스크에서 복사하고 각각의 파티션 정보를 수집해 파티션 테이블을 완성함으로써 복구가 가능하다. VBR 손상 시에는 파티션 가장 마지막 섹터에 있는 VBR 백업본을 덮어써서 복구가 가능하다.
2. **FAT16과 FAT32 파티션을 NTFS로 변환:** 하드디스크 최적화에는 NTFS 파일 시스템이 거의 유일한 선택지이다. FAT보다 성능이 좋다. 큰 파티션에 파일이 많이 있을 때 더욱 그렇다. NTFS 시스템 성능을 최대로 끌어내려면 파티션을 NTFS로 포맷하고 파일을 저장해야 하지만 여의치 않은 경우 CONVERT 유틸리티가 있어 FAT16,32를 NTFS로 변환할 수 있다.

**윈도우 부팅 순서:** [XP, 윈도우 서버 2000/2003의 부팅 순서]와 [비스타, 7, 2008의 부팅 순서]로 구분

1. **XP, 윈도우 서버 2000/2003의 부팅 순서:**
2. **POST(Power On Self Test) 실행:** 하드웨어가 스스로 시스템에 문제가 없는지 검사
3. **기본 부팅 관련 설정사항 로드:** BIOS는 CMOS(Complementary Metal-Oxide Semiconductor)에 설정된 시스템 설정사항 및 부팅과 관련된 여러 정보를 읽어 시스템에 적용한다.
4. **MBR(Master Boot Record) 로드:** 저장 매체의 첫 번째 섹터(LBA 0)에 위치하는 512바이트의 영역. 부팅 매체에 대한 기본 파일시스템 정보 포함
5. **NTLDR(NT Loader) 실행:** 하드디스크의 부팅 파티션에 있는 프로그램, 윈도우 서버 2000이 부팅될 수 있도록 간단한 파일시스템 실행, boot.ini 파일의 내용을 읽어 가능한 옵션을 보여준다.
6. **NTDETECT.com 실행:** NTLDR에서 제어권을 위임받아 시스템에 설치. CPU, 버스, 비디오 보드, 키보드와 마우스, 각종 포트… 등의 하드웨어를 검사한다.
7. **ntoskrnl.exe(NT OS Kernel) 실행:** ntoskrnl은 HAL.DLL(Hardware Abstraction Layer)을 로드한다.
8. **비스타, 7, 2008의 부팅 순서:**
9. **1~3단계 -> POST 실행/기본 부팅 관련 설정사항, MBR 로드:** XP, 2000/2003과 동일하다.
10. **4단계 -**> 윈도우 부트 서브 시스템(Window Boot Manager) 실행: 윈도우7은 MBR에서 NTLDR이 실행되지 않고 윈도우 부트 서브시스템이 실행된다. 서브시스템에서는 bootmgr.exe가 실행, 부트 설정 데이터(BCD, Boot Configuration Data)를 읽어 실행 가능한 운영체제 목록을 보여준다.
11. **5단계 -**> 윈도우 OS 로더(Winload.exe) 실행: 윈도우 OS 로더인 Winload는 NTDETECT와 같이 각종 장치 드라이브를 로드하고 ntoskrnl.exe를 실행

**윈도우 보안**

**윈도우 설치 시 주의사항:** 파티션이란 하나의 하드디스크를 논리적으로 분할해놓은 영역이다. 필요에 따라 RAID로 분할한다.

**윈도우 계정, 권한 인증:** 제어판 – 컴퓨터 관리 – 로컬 사용자 및 그룹에서 설정할 수 있다.

디폴트 계정으로 Administrator(root, Supervisor), SYSTEM(관리자보다 상위, 원격접속 불가, 사용자가 이 계정을 사용해 시스템에 로그인 불가), Guest가 있다.

**SID(Security Identifier):** 윈도우의 각 사용자와 그룹에 부여되는 고유 식별번호이다. 유닉스의 UID, RUID, EUID와 유사하게 윈도우에도 계정을 코드로 표현한 것이 SID이다. 윈도우의 whoami /user명령을 이용해 로그인 한 사용자의 SID를 볼 수 있다.

**윈도우의 권한 상승:** 작업 관리자에서 일반 권한의 사용자가 관리자나 시스템 권한의 프로세스를 빼앗는 것을 말한다. 상위 권한으로 수행 중인 프로그램의 프로세스에 다른 작업 끼워넣기로 권한을 빼앗을 수 있다. 정상: A -> A’ -> A’’, 권한 상승 A -> B’ -> 상위 권한 프로세스 획득(일반 사용자)

**윈도우 인증과 패스워드:** LSA(Local Security Authority), SAM(Security Account Manager), SRM(Security Reference Monitor)은 윈도우 인증의 가장 중요한 구성요소이다. SAM은 윈도우에서 패스워드를 암호화 해 보관하는 파일 이름과 동일하다. LSA는 로그인을 검증하고 시스템 자원과 파일에 대한 접근권한을 검사한다. 또 이름과 SID를 매칭해 SRM이 생성한 감사 로그를 기록하기도 한다. LSA는 NT보안의 핵심이며 보안 서브시스템이라고도 한다.

**SAM**은 사용자/그룹 계정 정보에 대한 DB를 관리한다. 사용자의 로그인 입력 정보다 SAM DB 정보를 비교해 인증 여부를 결정한다. 윈도우 SAM 파일은 %systemroot%/system32/config/sam에 위치한다. SAM이 계정과 패스워드 일치 여부를 확인해 SRM에 알리면 SRM은 사용자에게 SID를 부여한다. SRM은 SID에 기반해 파일이나 디렉터리에 대한 접근 허용 여부를 결정하고 이에 대한 감사 메시지를 생성한다.

**네트워크 드라이브의 이해:** 대상 컴퓨터의 드라이브를 내 컴퓨터에서 내 드라이브처럼 쓸 수 있는 기능이다. 바탕화면의 내 컴퓨터를 우클릭 해 네트워크 드라이브 설정을 클릭하면 이용할 수 있다. 명령 프롬프트에서 net user 드라이브명: ₩₩ip₩설정대상드라이브$를 실행하고 계정 및 패스워드를 입력하면 설정 가능하다.

**파일과 폴더의 보안권한 설정:** 윈도우에서는 폴더에 6가지 권한(파일 권한 + 폴더 내용 보기), 파일에 다섯 가지 권한을 부여할 수 있다.

**다섯 가지 권한:** 모든 권한, 수정, 읽기 및 실행, 읽기, 쓰기

**권한 할당하기:** GUI를 통해 접근제어리스트를 관리한다. 대상을 우클릭 후 속성 – 보안 – 편집을 클릭하면 개체의 사용권한 대화상자가 나타난다.

**디렉터리 및 파일에 접근권한 설정:** 다음 규칙을 따른다.

1. NTFS 접근 권한은 누적된다.
2. 파일에 대한 접근 권한이 디렉터리에 대한 접근 권한에 우선한다.
3. 허용보다 거부가 우선한다.

**공유 자료 관리:** 공유 리소스 설정 시 공유 이름 끝에 $기호를 붙이면 공유 자료를 숨길 수 있다. 이렇게 하면 원격 컴퓨터를 열 때 리소스의 목록에 나타나지 않는다. 숨겨진 공유를 보려면 명령 프롬프트에서 net share 명령을 사용한다.

**윈도우 공유 폴더:**

1. **C$, D$:** 하드 드라이브 관리 목적 공유 폴더. 드라이브가 많으면 개수만큼 표시된다. 인가받지 않은 사용자가 접근하기 쉬워서 취약하다.
2. **ADMIN$:** 윈도우 설치 폴더에 접근하는 관리목적 공유 폴더. 윈도우 버전마다 다르다.
3. **IPC$(Inter Process Communication):** 프로세스 간 통신. 네트워크 등에서 프로세스 간 통신을 위해 사용하는 통로이다. 동시 수행될 개별 프로그램을 생성/다룰 수 있게 하는 인터페이스. IPC$를 제거하면 네트워크에 문제가 발생할 수도 있으므로 레지스트리 값을 수정해 익명 사용자의 네트워크 접근이 불가능하도록 설정하는 게 좋다.

**암호 기능 사용**

**폴더 및 파일 암호화(EFS):** Encrypting File System은 윈도우 운영체제에서 개별 파일 ~ 특정 폴더 안의 모든 내용을 암호화할 수 있다. 파일이나 폴더의 속성을 변경해 사용할 수 있다.

**볼륨 암호화(BitLocker):** BitLocker는 윈도우 운영체제에서 제공하는 볼륨 단위의 데이터 암호화 기능이다. 볼륨은 파티션 드라이브(논리적 드라이브)를 의미한다. 시스템 파티션을 제외한 모든 파일을 암호화할 수 있다. BitLocker는 볼륨 단위의 암호화, EFS는 사용자 단위의 데이터 암호화를 제공한다.

**레지스트리 개요:** 레지스트리는 윈도우 시스템이 운영되는데 필요한 모든 정보를 담고 있다. 소프트웨어, 환경설정, 임시 저장값까지 거의 모든 정보를 담고 있어 공격자의 흔적을 찾기에도 유용하다.

**레지스트리 편집기:** 레지스트리 편집기를 통해 CRUD, 시작 – regedit or regedt32 입력, 처음 보이는 키 다섯 개는 특수키로 핸들이라고 한다. HKEY로 시작한다. 이들 키를 레지스트리의 루트키라고도 한다. 윈도우 부팅 시 하이브 파일에서 값을 읽어 들여 레지스트리를 구성한다. 하이브 파일에서 직접 읽어 구성하는 키를 마스터 키라고 하고 마스터 키로부터 값을 가져와 재구성하는 키를 Derived 키라고 한다. 마스터키 -> HKLM(HKEY\_LOCAL\_MACHINE), HKU(HKEY\_USERS)

Derived Key -> HKCU(HKEY\_CURRENT\_USER), HKCC(HKEY\_CURRENT\_CONFIG), HKCR(HKEY\_CLASSES\_ROOT)

**레지스트리 루트키:**

1. **HKEY\_CLASSES\_ROOT(HKCR):** 시스템에 등록된 파일 확장자와 해당 파일을 열 때 사용할 애플리케이션 맵핑 정보, CCM(Component Object Model) 오브젝트 등록 정보를 저장
2. **HKEY\_CURRENT\_USER(HKCU):** 시스템에 로그인하고 있는 사용자와 관련된 시스템 정보를 저장. HKEY\_CURRENT\_USER키에서 설정한 내용이 HKEY\_USERS보다 우선권을 갖게된다. 만약 HKEY\_CURRENT\_USER 값이 변경되면 HKEY\_USERS키의 보안 식별자에 해당되는 키 내용도 바뀐다.
3. **HKEY\_LOCAL\_MACHINE(HKLM)**: 설치된 하드웨어와 하드웨어를 구동시키는 드라이버, 설정 사항에 관한 정보를 가지고 있다. 루트키 중에서 가장 다양한 하이브로 구성되어 있다.

**주요 서브키로 다음과 같은 키들이 있다**.

* **HKLM₩HARDWARE:** 메모리에 휘발성 정보로만 존재한다. 부팅시 감지된 하드웨어와 해당 장치의 드라이버 맵핑 정보 보관. 장치 관리자를 통해 볼 수 있는 하드웨어 목록은 여기에 있는 레지스트리를 읽어 보여주는 것
* **HKLM₩SAM:** 패스워드, 소속 그룹, 도메인 정보와 같은 로컬 계정 정보와 그룹 정보가 있다. 엑티브 디렉터리에 도메인 계정과 그룹 정보를 저장한다.
* **HKLM₩SECURITY:** SAM과 마찬가지로 시스템 계정 이외의 접근이 불가. 시스템 범위의 보안 정책, 사용자 권리 할당 정보가 있다.
* **HKLM₩SOFTWARE:** 시스템 범위의 소프트웨어와 환경 설정 정보
* **HKLM₩SYSTEM:** 시스템이 부팅될 때 필요한 환경 설정 정보(로드할 디바이스 드라이버, 시작할 서비스 목록). 시스템이 성공적으로 부팅 시 정보를 저장해놓고 비정상 종료시 이 값을 바탕으로 부팅할 수 있는 옵션을 제공.

1. **HKEY\_USERS(HKU):** 모든 계정, 그룹에 관한 정보. user.dat에 저장
2. **HKEY\_CURRENT\_CONFIG(HKCC):** 시스템이 시작할 때 사용하는 하드웨어 프로파일 정보. 레지스트리에서 가장 단순. HKEY\_LOCAL\_MACHINE에 서브로 존재하는 config 내용을 담는다. 디스플레이와 프린터에 관한 설정 정보를 갖는다.

**레지스트리 보호:** 접근제한, 현재 레지스트리 상태 저장, 레지스트리 키 디스크에 복사 등의 방법이 있다.

**레지스트리 공격:** 악성코드 감염 시 시스템에 파일 생성 혹은 레지스트리 변경 등의 변화가 발생하기 때문에 이를 통해 악성코드를 탐지할 수 있다. 시스템 재부팅 시 악성 프로그램을 구동하기 위해 레지스트리를 변조하게 되는데 악성코드 구동을 위해 사용되는 레지스트리는 HKLM과 HKCU이다. 또 파일의 확장자에 따라 실행될 애플리케이션을 정하는 HKEY\_CLASSES\_ROOT의 위치값을 공격자가 임의로 조정해 특정 확장자에서 원하는 악성 코드를 실행하도록 할 수 있다. 공격자가 남긴 레지스트리 정보는 최근 사용한 문서 목록, 터미널 서비스 접속 목록, 설치된 소프트웨어 목록, 열어본 파일 목록 등을 통해 체크할 수 있다.

**윈도우 방화벽:** 네트워크 상의 웜이나 공격자로부터 PC를 보호하기 위해 사용한다. PC외부에서 들어오는 패킷과 나가는 패킷을 모두 차단하고 사용자에게 패킷의 적절성 여부를 판단한다. 윈도우의 파일 공유처럼 취약점이 잘 노출되는 서비스는 기본으로 차단하기도 한다. 제어판 – 관리도구 – 서버관리자의 구성 – 고급 보안이 설정된 Windows 방화벽의 인바운드/아웃바운드 규칙을 통해 설정 인바운드는 들어오는 패킷, 아웃바운드는 나가는 패킷에 대한 규칙이다.

**윈도우 서버 보안 설정**

**계정 관리:**

1. **Administrator 이름 바꾸기: Administrator** 계정은 이름을 기본값으로 둘 경우 로그인 실패 횟수에 제한이 없다. 관리자 이름을 바꿔 패스워드 뿐 아니라 관리자 이름 자체를 쉽게 유추할 수 없게 해야 한다.
2. **Guest 계정 상태:** 게스트 계정은 비활성화가 디폴트이지만 활성화 시 게스트 계정을 통해 시스템 정보 확인, 관리자 그룹에 추가 등 악용할 여지가 있다. 불필요한 경우 비활성화 해 두는 게 맞다.
3. **계정 잠금 임계값과 잠금기간 설정:** 로그인 실패 횟수를 제한하는 방법이다. 잠긴 계정은 일정 시간 사용할 수 없으므로 자동화된 공격의 효과를 낮출 수 있다.
4. **관리자 그룹에 최소한의 사용자 포함:** 관리자와 일반 업무를 위한 계정을 분리한다.
5. **암호와 패스워드 정책 설정:** 브루트 포스 공격에 대응하기 위해 패스워드 복잡도를 설정하고 최대 사용 시간과 최소 사용기간을 정한다.

**서비스 관리:**

1. **공유 권한 및 사용자 그룹 설정:** Everyone이 공유계정에 포함되면 익명 사용자의 접근이 가능하다. 디폴트 공유인 C$, D$, Admin$, IPC$ 등을 제외한 공유 폴더가 Everyone 그룹으로 공유가 금지되었는지 점검해 익명 사용자의 접근 여부를 확인해야 한다. 공유 디렉터리 접근 권한에서 Everyone 권한 제거 후 필요한 계정을 추가한다.
2. **하드디스크 기본 공유 제거:** 불필요한 공유 제거 – C$, D$, ADMIN$는 관리자가 네트워크상에서 시스템을 관리하기 위해 기본으로 마련된 것으로 관리상 목적으로 사용하지 않는다면 제거한다. 기본공유인 IPC$는 정상적인 방법으로 제거할 수 없으며 제거할 경우 특정 서비스가 실행되지 않을 수 있으므로 제거하지 않는 게 좋다. net share 공유이름 /delete를 통해 제거할 수 있다. 기본 공유는 공유 중지를 해도 운영체제가 재시작하면 다시 생성되니 아예 끄려면 레지스트리 수정이 필요하다.
3. **Null Session 접근 차단 설정:** Null Session을 사용해 사용자 인증을 거치지 않고 시스템 내부로 접근 가능한 취약점이 존재해 인증된 사용자만 접근할 수 있도록 설정해야 한다. 방화벽, 라우터, 네트워크 고급설정을 이용해 원천적으로 차단한다.
4. **불필요한 서비스 제거:** 시스템에 불필요한 취약한 서비스들이 디폴트로 설치되어 실행된다. 이런 서비스나 응용 프로그램은 취약점이 될 수 있으므로 제거해야 한다. 대표적으로 Alerter, Clipbook, Messenger, Simple TCP/IP Services 등은 제거하자.

**패치 관리:** 감사 정책(어떤 로그를 남길지 정의한 규칙)이 없거나 감사 수준이 너무 낮으면 문제 발생시 원인 파악이 어렵고 법적 대응을 위한 근거로 삼을 수도 없다. 하지만 감사 수준이 너무 높으면 시스템 성능에 악영향을 미치고 불필요한 항목과 중요한 정보를 구별하기 힘들어지니 적절한 수준으로 로그를 남겨야 한다.

**Windows 네트워크 서비스**

**디렉터리 데이터베이스:**

1. **네트워크 디렉터리 서비스:** 사용자가 자세한 디렉터리 경로를 몰라도 사용자에게 리소스를 제공해주는 것. LDAP(Lightweight Directory Access Protocol), NDS(Novell NetWare Directory Service), Microsoft Active Directory가 있다.
2. **Active Directory:** 네트워크상에서 개체 정보를 저장해 관리자와 사용자가 정보를 쉽게 찾아보도록 한다. 디렉터리 정보를 논리적 계층 구조로 조직한다. 서버, 볼륨, 프린터 등 공유 리소스와 네트워크 사용자, 컴퓨터 계정이 포함된다.
3. **Active Directory 서버**

**16절. 유닉스/리눅스 서버 보안**

**유닉스(UNIX)**

**텍스트, 스크린샷, 문서, 폰트이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명**

**커널:** 항상 메모리에 상주. 하드웨어 자원을 제어하면서 프로세스 스케줄링, 기억장치관리, 파일관리, 시스템호출 인터페이스, 입출력서비스 등의 기능을 제공한다. 가장 하위 수준의 하드웨어와 직접 관계하므로 커널의 일부분은 특정 하드웨어에 종속적이다. 그러나 UNIX는 어떤 운영체제보다 유연하게 동작한다. 유닉스의 구조는 하드웨어 – 커널 – 셸 – 응용 프로그램으로 되어있다.

**셸:** 일반 운영체제의 명령어 해석기에 해당한다. 최초의 셸인 본 셸, C 셸, 콘 셸, 배시 셸 등 다양한 셸이 있다.

**파일 시스템:**

1. **디렉터리 구조:** 계층 트리 구조의 디렉터리와 파일로 구성되어 있다. 최상위 디렉터리는 루트이다. root 디렉터리는 “/”로 표시되며 서브디렉터리로 분기되어 나간다.
2. **파일시스템의 구조:** 부트 블록, 슈퍼 블록, 아이노드 리스트, 데이터 블록의 네 가지 영역으로 분리된 자료구조를 갖는다. 파일시스템은 이 네 영역으로 구성된 자료구조를 제어해 유지된다.
3. **i-node:** i는 인덱스이다. 각 파일에 대한 정보를 기억하는 120바이트의 고정된 크기의 구조체이다. 일반 파일이나 디렉터리 파일의 아이 노드는 각 파일의 디스크 블록의 위치를 포함한다. 특수 파일의 아이 노드는 주변 장치를 식별할 수 있는 정보를 포함한다.
4. **슈퍼 블록:** 파일시스템의 정보를 유지하는 자료구조이다. 빠르고 효율적인 관리를 지원.

현재 사용하지 않는 디스크 블록과 아이 노드를 할당받기 위해 아이 노드 리스트와 데이터 블록을 순차적으로 하나씩 비교하는 방식은 비효율적이다. 슈퍼 블록에는 파일 시스템의 요약 정보와 함께 현재 사용되지 않는 아이 노드와 디스크 블록의 위치 정보가 있다.

**UNIX 기본 사용법**

**파일과 디렉터리 관리:** ls -l(리스팅) -a(도트 파일을 포함해 모든 디렉터리 및 파일) -R(하위 디렉터리에 있는 내용까지) -F(디렉터리’/’인지 실행 파일’\*’인지 심볼릭 링크‘@’인지) -i(아이노드)

**파일 유형:** -(일반) d(디렉터리) l(심볼릭 링크) b(블록 장치) c(문자 장치) p(파이프, 프로세스간 통신에 쓰는 특수 파일) s(소켓, 네트워크 통신에 사용하는 특수 파일)

**접근 권한:** user(파일 생성자), group(소유주가 속한 그룹), other(기타 사용자)

**권한 종류:** r-w-x, 4-2-1, 일반 파일 디폴트 권한 666, 디렉터리 디폴트 권한 777

**하드링크수: == 링크 카운터,** 동일한 i-node를 참조하는 파일의 수

**소유주와 소유그룹:** 위에서 설명, chown과 chgrp로 변경 가능

**파일 권한관리:**

1. **접근권한 변경(chmod): -R(**하위 디렉터리와 파일까지 일괄**),** 대상 -> u,g,o,a 연산자 -> +,-,=(지정), 접근권한 -> r,w,x or 4,2,1
2. **소유권 또는 그룹변경(chown/chgrp):** -R(상동), -h(심볼릭 링크 파일 자체의 소유주나 그룹 변경) root만 가능 chown/chgrp owner daemon.c 순이다.

**프로세스:** 유닉스의 프로세스의 생성과 실행은 fork()와 exec()이라는 두 시스템 콜로 처리된다. fork()는 부모 프로세스의 코드, 자료, 스택을 복사해 자식 프로세스에게 할당한다. exec()은 fork()를 통해 전달받은 복사본을 자신만의 형태로 치환해 실행 가능하게 한다. fork()는 부모와 자식 프로세스를 동시에 실행한다. 이때 부모 프로세스는 자식 프로세스의 PID를 반환하고 자식 프로세스는 0을 반환한다. 동일한 두 프로세스를 두는 이유는 순수한 상태의 부모 프로세스를 통해 상태보존을 하기 위함이다. 부모 프로세스가 직접 exec()명령을 하면 원래 상태를 잃게 된다. 자식 프로세스는 종료 시 부모 프로세스에게 자신의 종료를 알린다. 그리고 부모 프로세스가 확인하기 전까지 프로세스 테이블 목록을 차지한 좀비 프로세스가 된다. 좀비 프로세스는 코드, 자료, 스택, 세그먼트 등 어느 하나도 갖지 않지만 프로세스 테이블에 남아 일정 부분 자원을 차지한다. 과도하게 많아지면 더 이상 프로세스를 생성하지 못할 수도 있다. 또 자식 프로세스보다 부모 프로세스가 먼저 종료되면 자식 프로세스는 고아 프로세스가 되는데 UNIX의 규칙을 따르면 PID 1번인 init 프로세스가 고아 프로세스의 대리모가 된다.

**UNIX 시스템 관리**

**시스템 부팅:**

1. **런 레벨:** init프로세스는 /etc/inittab 파일에 정의된 런 레벨에 따라 etc/rc.d/rc[x].d [x]는 런 레벨, 디렉터리에 나열된 스크립트를 실행해 시스템의 운영상태를 구성한다.

|  |  |
| --- | --- |
| **런 레벨** | **시스템 운영 상태** |
| **0** | PROM(Programmable Read-Only Memory) 모드 |
| **S, s** | 시스템 싱글 유저 모드, 로컬 파일시스템 마운트 x |
| **1** | 시스템 싱글 유저 모드, 로컬 파일 시스템 마운트o |
| **2** | 멀티 유저 모드(NFS 클라이언트 모드) |
| **3** | 멀티 유저 모드(NFS 서버 모드) 디폴트 런 레벨 |
| **4** | 사용 안 함 |
| **5** | 시스템 Power Off 모드(리눅스에서는 X윈도우 환경 멀티유저 모드) |
| **6** | 시스템 리부팅 |

1. **런 레벨 단계 구성: INIT**의 상태는 런 레벨을 옮겨 다니며 각 레벨의 역할을 수행. init [x]로 해당 런 레벨로 이동할 수 있다.

**시스템 종료:** 접속 중인 사용자에게 종료를 공시해 작업을 마무리하도록 해야 하고 프로세스를 안전하게 종료해야 한다. 또 하드디스크를 갱신해 파일시스템의 무결성을 유지해야 한다. shutdown 명령은 시스템을 안전하게 종료할 때와 런 레벨을 변경할 때 사용한다. UNIX는 하드디스크의IO 효율을 높이기 위해 버퍼를 운영하기도 한다. 시스템이 비정상적으로 종료되면 버퍼에 있는 데이터가 하드디스크에 반영되지 않아 파일시스템 무결성에 문제가 생길 수 있다. 따라서 하드디스크 동기화(sync)는 시스템을 종료하는 중에 해야 할 가장 중요한 작업이다.

**파일 시스템 관리:**

1. **파일 시스템 연결(mount):** 보조기억장치를 시스템에 연결해야 작업을 처리할 수 있다. mount 명령은 보조 기억장치에 설치된 파일 시스템을 UNIX 시스템이 인식하도록 특정 디렉터리에 논리적으로 연결시키는 명령이다.

mount [-option] [device | mount\_point]

mount [-option] device mount\_point로 사용

옵션은 -a가 있다. /etc/fstab 파일에 정의된 모든 파일 시스템 마운트

mount /dev/cdrom /mnt/cdrom

/dev/cdrom 디바이스 파일 CD-ROM을 /mnt/cdrom 디렉터리에 마운트한다.

인수 지정없이 mount만 사용하면 현 시스템에 마운트 된 정보를 출력한다.

1. **파일 시스템 연결 해제(umout):** 프로세스가 해당 파일을 사용 중이거나 사용자가 파일시스템의 디렉터리에 있으면 umount 명령은 device is busy 메시지와 함께 실패한다. /etc/mtab 파일은 마운트 된 파일시스템의 정보를 관리한다. mount 명령을 실행하면 마운트 정보를 /etc/mtab 파일에 기록하고 umount하면 정보를 삭제한다.

umount [-option] [device | mount\_point]

-a: 마운트 된 모든 파일시스템을 언마운트한다. -f: 해당 파일시스템을 사용하는 프로세스를 강제로 종료하고 파일시스템을 언마운트한다.

**프로세스 스케줄 관리:**

1. **정기적 스케줄 관리(cron):** cron 데몬 프로세스는 지정 시간에 작업을 처리하기 위해 사용한다. 일괄적으로 처리할 작업, 반복적인 작업에 유용하다. cron 데몬 프로세스를 사용하려면 세 가지 구성요소가 필요하다.

* 작업 목록을 정의한 crontab파일
* crontab파일을 제어(편집)하는 crontab 명령
* crontab파일을 읽어 정의된 대로 작업을 처리하는 cron 데몬 프로세스

유닉스에서 기본으로 제공하는 프로세스이므로 crontab 명령으로 작업을 정의하는 방법만 알면 된다.

1. **crontab 파일의 구조:** 각 행은 Space나 Tab으로 구분된 6개의 필드로 이뤄진다.

|  |  |  |
| --- | --- | --- |
| 구분 | | 기술 방법 |
| 필드 | 의미 |
| 1 | 분 | 0-59 |
| 2 | 시 | 0-23 |
| 3 | 일 | 1-31 |
| 4 | 월 | 1-12 |
| 5 | 요일 | 0-6(0:일요일) |
| 6 | 작업 | 지정 시간에 실행할 작업 절대 경로로 기술, 필요한 옵션 및 인수를 함께 나열한다. |

\*기호는 각 필드의 범위에 해당하는 모든 값을 의미하고 -기호는 범위를 기술할 때 사용한다. ,기호는 값을 구분할 때 사용, /기호는 간격값을 지정할 때 사용한다.

1. **crontab 파일의 제어:** 크론탭 파일은 계정별로 생성된다. 루트는 login\_name을 명시적으로 사용해 다른 사용자의 크론탭 파일을 편집할 수 있지만 일반 사용자는 자신의 것만 편집 가능하다.

crontab -e | -l | -r [login\_name]

-e: edit -l: list -r: remove

계정 이름을 명시하지 않으면 자신의 계정이다.

리눅스에서는 crontab -u user -e와 같이 -u 옵션을 더 붙인다.

**UNIX 서버 보안**

**사용자 패스워드 관리:** /etc/passwd에 사용자 정보가 담겨있다. 과거에는 패스워드를 인코드한 텍스트를 파일의 두 번째 필드에 저장했지만 최근 UNIX 시스템에서는 /etc/shadow 파일에 암호화 해 저장한다.

1. **passwd:** 라인마다 개별 사용자에 대한 정보

**[user\_account]:[user\_password]:[user\_ID]:[group\_ID]:[comment]:[home\_directory]:[login\_shell]**

ex. root:x:0:0:Super-User:/root:/bin/sh

경로는 절대 경로로 표기한다.

추가적으로 /etc/shadow에 저장된 파일을 passwd파일 두 번째 필드에 예시처럼 x가 아닌 실제 패스워드 문자열로 드러내고 싶으면 pwconv 명령을 사용하면 된다. 해제 시에는 pwunconv한다.

1. **shadow:** /etc/shadow에 암호화된 패스워드를 저장하고 root만 읽을 수 있게 제한한다. 패스워드와 패스워드 에이징 정보가 있다. 패스워드 에이징 정보는 시간의 흐름에 따른 패스워드 관리 정책을 말한다.

**[user\_account]:**[encrypted\_password]:[last\_change]:[minlife]:[maxlife]:[warn]:[inactive]:[expires]

**접근권한 마스크(umask):** 관리자는 /etc/profile에 umask를 설정해 전체 사용자에게 획절적인 umask 값을 적용할 수 있다. umask [mask]로 쓴다 mask값은 8진수 r-w-x값이다. 보통 그룹과 other에 쓰기 제한을 두는 022를 많이 사용한다.

**권한 상승(SetUID, SetGID):** 계정이 누군가를 식별하는 UID, GID를 각각 RUID(Real), RGID라고도 한다. 유닉스는 어떤 권한을 가지고 있는가에 대한 UID, GID를 따로 갖고 있다. EUID(Effective), EGID라고 한다. real은 프로세스를 실행시킨 사용자의 UID 또는 GID를 말하고 EUID, EGID는 프로세스가 실행 중인 동안만 부여되어 자원의 접근권한을 판단한다. SetUID 비트가 설정된 파일은 모두 해킹 대상이 될 수 있어 백도어 및 오버플로우 공격에 악용된다. SetUID가 설정된 파일은 목록화 해 관리하는 작업이 필요하다.

**SetUID 비트를 이용한 bash 셸 획득**

**backdoor.c**

**#include <stdio.h>**

**main() {**

**setuid(0);**

**setgid(0);**

**system(“/bin/bash”);**

**}**

프로그램이 실행되었을 때 이미 root권한의 프로세스이기 때문에 system(“/bin/bash”)을 실행하는 프로세스 권한은 root가 된다.

**SetUID설정 시 주의사항과 SetGID:** SetUID가 설정된 프로그램을 실행하는 경우, 프로세스가 갖는 권한은 프로그램을 실행한 사용자의 권한이 아닌 프로그램 소유주의 권한이 된다. root로만 접근할 수 있는 시스템이나 파일에 일반 사용자도 접근할 필요가 있다면 SetUID를 활용할 수 있다. 불필요한 이용은 당연히 보안상 허점이 된다. 파일 종류와 권한 설정에 있어 유닉스는 16비트를 사용한다.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 파일종류(4bit) | 특수권한(3bit) | | | User(3bit) | | | Group(3bit) | | | Other(3bit) | | |
| -, d, c, b, s, l, p | 4 | 2 | 1 | 4 | 2 | 1 | 4 | 2 | 1 | 4 | 2 | 1 |
| suid | sgid | stick-bit | r | w | x | r | w | x | r | w | x |

**디렉터리 접근권한(sticky-bit):** 특별 접근권한을 부여할 수 있는 비트가 sticky-bit이다. 공유모드라고도 한다. 스티키 비트가 설정된 디렉터리는 시스템 내의 모든 사용자가 파일이나 하위 디렉터리를 생성할 수 있다. 삭제는 소유주나 root만 가능하다. su-sg-sticky 순으로 4000 2000 1000번대 8진수를 사용한다.

**네트워크 보안**

1. **슈퍼 서버[inetd 데몬]:** 유닉스 시스템에는 다양한 서버 프로그램이 실행되고 있다. inetd 데몬은 N개의 개별 서버를 하나로 통합해 클라이언트로부터 서비스 요청이 올 때마다 관련된 실행 모듈(FTP, Telnet, TFTP) 등을 실행해준다. 불필요한 서비스를 제한하려면 /etc/inetd.conf 파일의 해당 라인을 주석처리(#)하면 된다.
2. **inetd.conf:**

**telnet stream tcp6 nowait root /usr/sbin/in.telnet in.telnet**

분석하자면 차례대로 서비스, 소켓 타입, 프로토콜, 대기 설정, 로그인 이름, 서버, 인자이다. 서비스는 서비스 이름을 정의하는 것으로 /etc/services에 정의되어 있어야 한다. 소켓 타입은 서비스에 대한 소켓 유형을 설정한다. TCP일 경우 stream, UDP일 경우 dgram이라고 표현한다. 프로토콜은 /etc/protocols에 정의된 프로토콜의 종류와 번호이다. 프로토콜 번호는 임의가 아닌 표준이다. 대기 설정은 inetd가 클라이언트에 서비스를 요청받은 경우, 이후에 즉시 또 다른 요청을 처리할 것인지에 따라 nowait, wait로 구분한다. TCP는 반드시 nowait여야 한다. 로그인 이름은 데몬을 어떤 사용자의 권한으로 수행할지 명시한다. 데몬의 실행 권한 설정은 보안에서도 매우 중요한데 버퍼 오버플로우같이 해킹 공격에 취약한 데몬이라면 절대 root를 주면 안 된다. http 같은 경우 반드시 nobody로 설정해야 한다. 서버는 해당 서비스를 수행하기 위해 어떤 프로그램을 실행할지 절대 경로로 적는 공간이다. 인자에는 데몬을 실행하는데 필요한 인가 값을 넣는다. 리눅스는 최근 inetd 데몬보다 xinetd를 더 많이 이용한다.

|  |  |  |
| --- | --- | --- |
| 구분 | Xinetd 환경(리눅스) | Standalone 서비스 |
| 의미 | Xinetd슈퍼데몬에 의해 제어되는 서비스 데몬 | 독립적으로 실행되는 서비스 데몬 |
| 실행방법 | 필요할 때 xinetd에 의해 수행 | 독립적 서비스를 위해 항상 독립적으로 수행됨 |
| 데몬상주여부 | Xinetd에 의해 불린 후 서비스 완료 후 종료 | 독립적인 서비스를 위해 항상 메모리에 독립 데몬으로 상주 |
| 접근제어 | Tcpd에 의해 접근제어(/etc/hosts.allow, /etc/hosts.deny 파일로 제어) | Tcpd와 무관하지만 자체 설정파일에 의해 접근제어를 할 수 있음 |
| 기타 | /etc/xinetd.d/ 서비스명으로 된 xinetd 제어파일이 존재함 | httpd, sendmail, named 등 |

**접근 통제(TCPWrapper):** 외부에서 들어오는 클라이언트에게 접근통제 기능을 제공하는 게 TCPWrapper이다. 클라이언트의 IP를 확인해 접근을 허용한 호스트들에 대해서만 서비스를 제공한다. TCPWrapper설치 시 inetd 데몬은 TCPWrapper의 데몬인 tcpd 데몬에 연결을 넘겨준다. tcpd 데몬은 클라이언트가 접속 요구시 클라이언트의 권한을 확인하고 해당 데몬 연결을 넘겨준다. 이때 연결에 대한 로깅도 실시한다. 모든 프로토콜에 대한 접근 제어를 할 수는 없으나 inetd 데몬이 관리하지 않는 standalone데몬 같은 경우 inetd 데몬이 통제할 수 없으며 TCP외의 일부 프로토콜에 대해서만 통제가 가능하다.

**hosts.allow(접근 허용)와 hosts.deny(접근 금지):** 접근 허용과 금지는 /etc/hosts.allow와 /etc/hosts.deny 파일에 정의된 호스트 정보를 기준으로 한다. 이를 통해 외부로부터의 접근을 선택적으로 제한할 수 있다. hosts.allow -> hosts.deny 순으로 읽는다. hosts.allow와 hosts.deny두 파일에 모두 접근하고자 하는 클라이언트가 설정사항에 포함되지 않을 때는 접근을 허용하게 된다.

service\_list: client\_list [: shell\_command]

service\_list는 한 개 이상의 네트워크 서비스명 또는 예약어 ,로 구분

client\_list는 한 개 이상의 호스트명, 도메인명, IP주소, 네트워크ID, 예약어로 구분자,로 구분한다.

shell\_command 일치하는 것이 있을 때 선택적으로 tcpd가 실행하는 셸 명령

설정 내용에 오류가 없는지 확인하려면 /usr/sfw/sbin/tcpdchk를 실행해본다. 오류가 출력되지 않으면 설정 파일에 구문 오류는 없는 것이다.

**PAM(Pluggable Authentication Modules, 장착형 인증 모듈):** PAM은 리눅스 배포판에서 사용자 인증의 핵심 라이브러리이다. 애플리케이션에 대한 인증 형태, 권한, 접근 자원 등을 선택할 수 있다. 기존 응용 프로그램을 수정할 필요 없이 새 인증 서비스 모듈을 추가해 사용할 수 있는 프로그램.

**PAM을 사용한 인증 절차:** 프로그램(login, Telnet, FTP 등)은 인증이 필요할 때 PAM라이브러리를 호출한다. PAM 라이브러리가 호출되면 해당 프로그램의 PAM 설정파일을 참조해 PAM 모듈들을 실행하고 결과를 응용 프로그램에 반환한다. 응용 프로그램은 반환 결과를 통해 인증 여부를 결정한다.

**PAM 설정 파일:** type control module-path module-arguments

type: 모듈 종류

* account: 사용자의 시스템 권한을 확인
* auth: 인증기능, 패스워드 확인 담당
* password: 패스워드 설정, 확인 시 사용
* session: 사용자 인증 성공 시 세션을 맺는다.

control: 각 모듈 실행 후 성공/실패에 따라 다른 PAM 라이브러리 행동 결정

* requisite: 모듈 실행에 실패 시 바로 인증 거부, 인증 실패한 단계에서 더는 진행하지 못하게 하려면 지정한다.
* required: 모듈 실행에 실패해도 즉시 인증을 거부하지 않음. 동일 유형 모듈 실행을 완료한 시점에 인증을 거부한다.(PAM 인증 결과를 마지막에 통지) 사용자가 어느 단계에서 인증에 실패했는지 모르기 때문에 requisite보다 안정성이 높다.
* sufficient: 이전 모듈이 실패해도 이 모듈에서 인증 성공 시 PAM은 인증 승인, 실패 시 평가를 계속한다.
* optional: 모듈의 성공, 실패 응답을 상관 안 한다.(거의 사용하지 않는다.)

**리눅스**

유닉스 타입의 운영체제로 유닉스를 PC버전으로 개발한 노력의 산물이다. 1991년 리누스 토발즈의 0.01 커널(PC, 멀티태스킹)에서 출발. 이후 해커들에 의해 개발, 리처드 스톨만의 FSF의 GNU 프로젝트로서 진행 중이다.

**파일 시스템의 종류 및 특징:** ext, ext2, ext3, minix, xiafs, msdos, hpfs OS/2, isofs CD-ROM, umsdos, nfs 등이 중요 파일 시스템

**minix:** 미니 유닉스, 유닉스를 배우려는 사람들을 위해 엔드류 타넨바움이 개발한 교육용 PC-UNIX,에서 사용되던 파일시스템이다. 가장 오래되고 근본인 파일 시스템. 파일시스템은 30문자로 제한, 파일시스템마다 최대 64MB의 성능제한.

**xiafs:** 미닉스의 제한 사항인 파일 이름과 파일 시스템에 대한 제한을 보완한 수정 버전

**ext:** 리눅스 초기 시스템

**ext2(Second Extended File System, 2차 확장 파일시스템):** ext2는 캐시에 저장되어 있는 데이터를 디스크에 저장하는 도중 시스템이 다운되거나 문제가 발생하면 파일시스템이 손상되는 단점. 이를 위해 fsck(file system check)라는 파일시스템 복구기능을 제공하는데 시간이 많이 소요되며 복구하는 동안 시스템 사용이 불가능하다.

**ext3(extended file system 3, 확장된 파일 시스템 3):** 커널 2.4부터 지원. ext2의 단점을 보완해 시스템 무결성과 뛰어난 복구 기능을 보장한다. ext3에 적용된 기술은 저널링이다. 저널링은 기존 fsch에 걸리는 시간을 단축시키기 위해 데이터를 디스크에 쓰기 전에 로그를 남겨 시스템의 비정상적인 셧다운 발생 시 로그를 참조해 fsck보다 빠르고 안정적인 복구를 지원하는 기능이다. ext2와도 호환된다.

**msdos:** MS-DOS의 FAT 파일 시스템과의 호환을 지원하는 파일시스템

**hpfs OS/2:** OS/2(IBM+마이크로소프트 합작 개발 운영체제)의 파일 시스템

**isofs CD-ROM:** ISO 기준을 따르는 표준 CD-ROM파일시스템이다.

**umsdos:** MS-DOS 파일 시스템을 리눅스에서 사용하도록 확장한 파일시스템

**nfs:** 네트워크 파일 시스템

**유닉스/리눅스 서버 취약점 분석-평가**

**계정 관리:**

1. **root 계정 원격 접속 제한:** 원격 접속 시 root로 접속할 수 없도록 설정파일을 수정해야 한다.

|  |  |
| --- | --- |
| **OS** | **점검 파일 위치 및 점검 방법** |
| **SunOS** | #cat /etc/default/login  CONSOLE=/dev/console |
| **LINUX** | #cat /etc/pam.d/login  auth required /lib/security/pam\_securetty.so  #cat /etc/securetty  pts/0 ~ pts/x 관련 설정이 없음 |
| **AIX** | #cat /etc/security/user  rlogin = false |
| **HP-UX** | #cat /etc/securetty  Console |

1. **패스워드 복잡성 설정:** 계정과 유사하지 않은 8자 이상의 영문, 숫자, 특수문자 조합

**SunOS, LINUX, HP-**UX: /etc/shadow에서 패스워드 점검

**AIX:** /etc/security/passwd에서 패스워드 점검

1. **계정 잠금 임계값 설정:** 브루트 포스나 패스워드 추측 공격에 대비해 적정 암호 입력 횟수를 제한해 공격을 차단하거나 지체시킬 수 있다. 계정 잠금 임계값을 5이하로 설정한다.

**OS별 점검 파일 위치 및 점검 방법**

* **SunOS:** #cat /etc/default/login에서 RETRIES=5, 5.9이상 버전은 추가적으로 policy.conf 파일을 확인한다. #cat /etc/security/policy.conf에서 LOCK\_AFTER\_RETRIES=YES
* **LINUX:** #cat /etc/parm.d/system-auth, auth required /lib/security/pam\_tally.so deny=5 unlock\_time=120 no\_magic\_root, account required /lib/security/pam\_tally.so no\_magic\_root reset
* **AIX:** #cat /etc/security/user, loginretries=5
* **HP-**UX: #cat /tcb/files/auth/system/default, u\_maxtries#5

1. **패스워드 파일 보호:** 패스워드를 평문이 아닌 암호화 해 저장하고 /etc/shadow에서 특별 권한이 있는 사용자들만 읽을 수 있게 한다. 썬이나 리눅스는 /etc/shadow가 존재하고 /etc/passwd 두 번째 필드가 x로 표시되면 암호화 되어 있는 것이다. HP-UX에서는 /tcb 디렉터리가 존재하고 /etc/passwd에서 두 번째 필드가 x인지 확인하면 된다.
2. **root 이외의 UID 0금지:** root(UID=0)과 동일한 UID를 가진 계정이 존재할 경우 root로 접근이 가능하므로 중복시키지 않도록 한다. UID 중복 자체로 사용자 감사 추적이 어려워지는 등 문제가 발생하므로 계정 및 UID 확인이 필요하다. 썬 리눅스 HP-UX 동일하게 #cat etc/passwd내 세 번째 필드값으로 확인이 가능하다. root 이외의 UID가 0이면 다른 필드값을 부여한다.
3. **패스워드 최소 길이 설정:** ~239p