퀴즈1,2

2022년 3월에 삼성전자와 LG전자를 해킹한 해킹그룹의 영문 명칭은?

LAPSUS$

방화벽 (Firewall)

(1) 연결된 두 네트워크 사이에서 데이터 통신 트래픽을 제한하는(통과시키거나 필터링하는), 네트워크간 연결 장치(inter-network connection device)  
(2) 지역 보안 정책에 따라 네트워크들 사이의 접근을 제한하는 게이트웨이  
(3) 외부에서 내부 망으로 들어오거나 또는 내부 망에서 외부로 나가는 데이터 통신 트래픽을 제한하여, 외부 위협으로부터 네트워크 시스템 자원을 보호하는, 네트워크간 게이트웨이 (inter-network gateway)

2. 네트워크 프로토콜에 대한 다음 물음에 답하시오. 영문 명칭으로 쓰시오.

(1) 웹 페이지 전송에 사용되며, 80번 포트를 기본으로 사용하는 프로토콜은? HTTP

(2) 이메일을 전송하기 위해 사용되며, 25번 포트를 기본으로 사용하는 프로토콜은? SMTP

(3) 원격 서버로부터 이메일을 가져오기 위해 이메일 클라이언트에 의해 주로 사용되며, 110번 포트를 기본으로 사용하는 프로토콜은? POP3

(4) 원격 서버로부터 이메일을 가져오기 위해 이메일 클라이언트에 의해 주로 사용되며, 995번 포트를 기본으로 사용하는 프로토콜은? POP3

(5) 원격 서버에 있는 메일박스(mailbox)를 접근하기 위해, 지역 이메일 클라이언트에 의해 사용되며, 143번 포트를 기본으로 사용하는 프로토콜은? IMAP4

Botnet의 동작방식

봇 멀웨어에 감염되어 해커에 의해 원격으로 제어되는, 하이재킹 된 컴퓨터 및 장치들의 네트워크.

봇 네트워크는 스팸을 보내고 DDoS(분산 서비스 거부)공격을 시작하는 데 사용되며 다른 사이버 범죄자에게 임대될 수 있다. 봇넷은 P2P 아키텍처 및 기타관리채널을 사용하여 한 봇에서 다른 봇으로 명령을 전송함으로써 명령 및 제어(C&C) 서버 없이 존재할 수도 있다.

봇넷은 인터넷에 연결된 여러 많은 장치로, 각 장치는 하나 이상의 봇을 실행.

봇넷을 사용하여 DDoS(분산 서비스 거부) 공격을 수행하고, 데이터를 훔치고, 스팸을 보내고, 공격자는 장치를 접근하고 연결할 수 있다.

봇넷 소유자는 명령 및 제어(C&C) 소프트웨어를 사용하여 봇넷을 제어할 수 있다.

Cryptojacking

크립토재킹(Cryptojacking)은 범죄자가 피해자의 컴퓨터 파워를 비밀리에 사용하여 암호화폐를 생성하는 사이버 범죄 유형

* 범죄자는 ‘코인 채굴기’(coin miner)라는 프로그램을 사용하여 암호화폐를 생성하거나 ‘채굴’

크립토재킹(Cryptojacking)은 웹사이트를 통해 또는 사용자가 알지 못하는 동안, 암호화폐를 채굴하기 위해 피해자 컴퓨터를 하이재킹하는 행위

* 크립토재킹 멀웨어는 일반적으로 사용자 모르게 암호 화폐를 채굴하기 위해 컴퓨터를 감염시키는 멀웨어

SQL injection

* 불충분하게 처리된 사용자 입력을 데이터베이스 백엔드로 전달하는 웹사이트를 찾는 공격
* 코드주입(code injection)의 한 기업으로 클라이언트의 입력 값을 조작하여 서버의 데이터베이스를 공격할 수 있는 공격방식
* 데이터 기반 애플리케이션을 공격하는 데 사용되는 코드 주입 기법으로, 실행을 위해 입력 필드에 악성 SQL문을 삽입할 수 있는 기법(예: 데이터베이스 내용을 공격자에게 덤프)

Cross-Site Scripting (XSS)

무해한 웹사이트에(an otherwise benign website)에 공격자가 악성코드를 주입할 수 있게 하는 취약점

* 이러한 스크립트(악성코드)는 대상 웹사이트에서 생성된 스크립트의 권한을 획득하여, 웹사이트와 클라이언트 간에 전송되는 데이터의 기밀성과 무결성을 손상시킬 수 있다.
* 주입된 데이터가 실행 불가능하도록 데이터를 정제하지 않고, 요청이나 폼(form)으로부터의 사용자 제공데이터를 디스플레이 한다면, 해당 웹사이트는 취약하다.

일부 웹 응용 프로그램(웹 애플리케이션)에서 발견된 보안 결함으로,

* 이 보안 결함을 악용하여, 권한이 없는 공격자는 클라이언트 측 스크립트가 웹 애플리케이션의 다른 사용자에 의해 실행되도록 유도할 수 있다.

3가지 유형이 존재

Type 1 : Reflected XSS (or Non-Persistent)

Type 2 : Stored XSS (or Persistent)

Type 0 : DOM-Based XSS

Attack surface란?

* 공격대상이 되는 시스템, 시스템구성요소 또는 환경의 경계에 있는 접점들의 집합.

이 접접들의 집합을 통해, 공격자는 해당 시스템, 구성요소 또는 환경에 집입하거나, (해당 시스템, 구성요소, 환경에) 영향을 미치거나, (해당 시스템, 구성요소, 환경으로부터) 데이터를 추출할 수 있음.

* 공격자가 시스템에 접근하여 데이터를 추출할 수 있는 모든 가능한 지점 또는 공격 벡터들.

Attack surface를 줄이는(최소화하는) 방법은?

* Eliminate Complexity (복잡도를 감소시켜라)
* Visualize Your Vulnerabilities (또는 취약점을 모니터링하라): (1) Attack surface modeling,(2) Attack simulation, (3) Patch simulation
* Control your endpoints, 또는 모든 것을 신뢰하지 마라(Assume zero trust), 또는 강한 암호화 정책을 사용하라
* Segment your network
* Prioritize analytics (분석 우선순위 지정) 또는 회사 구성원들을 교육.

Code review 에 대해 설명하시오

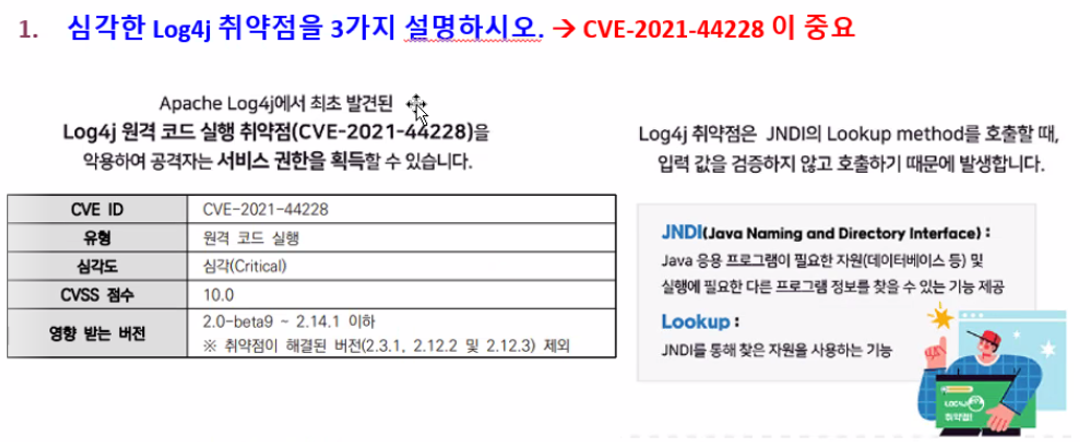
* 코드 리뷰(때로는 상호 검토라고도 함)는 한명 또는 여러 사람이 주로 소스코드의 일부를 보고 읽는 방식으로 프로그램을 확인(검토)하는 소프트웨어 품질 보증 활동으로, 구현 후 또는 구현 중간에 수행함.
* 동료 프로그래머와 함께 체계적으로 회합(회의를 소집)하여 서로의 코드에 실수가 있는지 확인하는 행위

Penetration testing (침투 테스트)

* 테스터가 개별 바이너리 구성 요소 또는 전체 응용 프로그램을 대상으로 하는 테스트 방법으로, 내부 또는 구성요소 간 취약점들이 응용 프로그램, 응용 프로그램 데이터 또는 환경 리소스를 손상시키기 위해 악용될 수 있는지 여부를 판단(결정)하는 테스팅.
* 일반적으로 특정 제약 조건에서 작업하는 평가자(assessor, 테스터)가 시스템의 보안기능을 우회하거나 무력화하려고 시도하는 테스트 방법론.
* 평가자(evaluator)가 애플리케이션, 시스템 또는 네트워크의 보안 기능을 우회하는 방법을 식별하기 위해 실제 공격을 모방하는 보안
* 테스트.

침투 테스트는 종종 실제 공격자가 사용하는 것과 동일한 도구와 기술을 사용하여 실제 공격을 실행하는 것을 포함. 대부분의 침투테스트에는 단일 취약점을 통해 달성할 수 있는 것보다 더 많은 액세스 권한을 얻는데 사용할 수 있는 단일 시스템 또는 여러 시스템에서 취약점 조합을 찾는 것이 포함됨.

1. 심각한 Log4j 취약점을 3가지 설명하시오.
2. 공격자가 이 3개의 취약점들을 악용하여 어떻게 공격할 수 있는지 설명하시오.
3. 이 3가지 취약점을 완화하거나 제거하기 위한 대응 기법을 설명하시오.



CVE-2021-44228 , CVE-2021-45046, CVE-2021-4104

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

컴퓨터의 핵심 : 데이터나 정보를 저장하고 처리하는 전자적인 기기

* DATA
  + Raw Data = 가공되지 않음
  + Unorganized = 조직화되지 않음
  + Unprocessed = 처리되지 않음
  + Chaotic or Unsorted = 혼란한 상태
  + Input to a Process = 어떤 Process의 입력

* Information
  + Useful & Relevant = 유익하고 적절한 형태로 되어있음
  + Organized = 조직화됨
  + Processed = 처리되어있음
  + Ordered or Sorted = 정렬되어있음
  + Output of a Process = 어떤 프로세스의 출력

많은 조직에서 정보와 데이터는 중요한 자산이다!!

* Information Technology : 데이터나 정보를 처리하거나 배포하기위해서 컴퓨터 시스템이나 네트워크를 개발하거나 사용하는 기술

* SW Engineer  
  컴퓨터 소프트웨어에 대해서 설계하고 개발하고 테스트, 유지보수하는 일을 함, 이러한 원칙들을 적용함  
  전체 과정에 참여 => 주로 팀단위 활동  
  HW 시스템의 구성요소를 알 필요가 있음 => ex) hdd인지 ssd인지 ...등  
  SW 개발시 필요한 도구를 만듦  
  큰 규모에 관련된 이슈를 해결할 필요가 있음 => 전체과정을 관여하기 때문에

* SW Developer  
  다양한 유형의 컴퓨터에 소프트웨어를 빌드함  
  전체 과정중에서 한 부분에서 관여함(주로 구현부분) => 주로 개별활동  
  완전한 프로그램을 작성  
  만들어진 도구를 사용(편집기 컴파일러 디버거...등)  
  제한된 규모에 관여를 함 => 한 부분에 관여하므로

* Information System  
  Hardware, Software, Networks, Data, People, Processes의 집합  
  <-> Information Technoloy : HW, SW, Networks, Data

* Data를 가공하면 정보가됨 => 회사의 절차나 정책은 정보를 어떻게 처리, 배포할것인지 관련됨 따라서 데이터와 정보가 중요함

* Security??  
  위협, 위험, 취약점이 없는 안전한 상태, 공격자가 존재하는 상황에서도 원하는 특성을 만족하는 것.
* + Interception : 중간에 가로챔(도청)
  + Interruption : 중간에 막음
  + Modification : 중간에 데이터를 변조함
  + Fabrication : 제 3자가 송신자인척 보냄 (위조) => 인증이 필요 => id/pw, 공인인증서, 생채 인증 등..
  + 기밀성 : 권한을 가진 사람만 정보 접근 => 도청 방지 => 비인가적인 읽기 막아야
  + 무결성 : 인가된 사람만 수정가능, 정보의 출발지 확인(위조 방지) => 비인가적인 쓰기 막아야 =>정보의 정확성, 안정성 보장
  + 가용성 : 인가된 사용자는 그 정보에 항상 접근할 수 있음 => DDoS 방지 => 데이터는 정상적인 사용자가 필요로할때 언제든 이용가능해야한다.

=> 공격자가 존재하는 상황에서 C.I.A를 집행함

* cia로만 충분하지 않음 => 인증필요
* 네트워크 보안도 중요 => 네트워크 프로토콜(보안 강화된) 중요, 암호알고리즘
* 인증되었어도 허가된 행동만 수행하도록 해야함 => 인가 (authorization)
* 멀티 유저 시스템에서 인가가 필요함(root, suepr user, normal user, guest)
* 부인방지  
  You did that => 하지 않았다고 부인하는것을 방지
* 책임 추적성(Accountability)  
  who, when, where, why, how, what  
  모든 transcation에 대해 기록하고 제공해야함  
  행동들이 추적간능해야한다.
* Information Security  
  인가되지 않은 접근, 기록, 수정, 변조, 파괴 등으로부터 정보를 보호  
  => 허가되지 않은 공격으로부터 정보를 보호  
  (Destruction : 일부 삭제 or 교체)
* Computer Security  
  적대적인 환경으로부터 컴퓨터 시스템을 보호하는 것  
  (계획된 사용은 허용, 불법적인 사용을 막음)  
  컴퓨터 시스템 자신과 컴퓨터 시스템이 저장하거나 접근하는 데이터를 보호하는 것.
* 컴퓨터 보안과 정보보안의 차이  
  컴퓨터보안 : 디지털 형태로 된 데이터나 정보에 초점을 두고 보호, 디지털로 된 정보, 시스템, 네트워크 디지털을 다루는 시스템이나 네트워크를 보호하는 것  
  정보보안 : 아날로그, 디지털 정보를 보호, 디지털로된 정보, 디지털로된 시스템 네트워크뿐만아니라 아날로그로 된 정보까지도 보호대상임

공격, 위협으로부터 보호

* 공통점 : 컴퓨터 시스템이나 네트워크를 보호,

* Software Security  
  악의적인 공격하에서도 SW기능이 정상적으로 동작하도록 설계,개발, 테스팅하는 개념  
  취약점을 차단하는것과 다름 => 취약점 : 보안상의 허점  
  뚫리고 난 후 사후대책을 하는 것은 아님(reacting)안전한 SW를 개발하는 모든 것  
  보안을 위해서 SW를 설계 구현 테스트하는 과정  
  Pro-Active, 능동적인 접근 방식 : 미리 예방함 => SW를 구성할때 보안을 내재화(각 개발 단계마다, 만들때부터 보안 고려)

* Security engineering과 관련있음 => 모든 과정에 관련, 시스템 설계시 보안 측면에 초점을 둔 엔지니어링의 특별한 분야  
  보안 요소 : 문제의 요소가 되는 것을 처리하는데 필요한 보안요소들 => 자연적인 재해, 악의적 공격 등

* SW에 의해서 유발되는 보안 위험을 이해하고 예방하고 보완하는 것임

* Microsoft Security Development Lifecycle  
  보안에 관련된 설계 결함, 코딩 단계의 결함을 줄이는 것 => 취약점, 해커에 의해서 악용될 수 있는 취약점을 줄이는것  
  (\* 공격자가 방어자보다 유리함, 공격자는 취약한 점 하나만 찾으면 되지만 방어는 모든 취약점을 찾아서 막아야함)=> 취약점을 제거, 제거하지 못한다면 심각도를 줄임-Secure by Default  
  공격에 문제가 없는 시스템을 만듦

* -Secure in Deployment and Communication  
  배치, 설치, 통신시 안전하게 함

* -Secure by Design  
  설계시부터 보안 고려

* 남아있는 취약점들의 심각도를 줄임

SDL은 SW 개발 주기에 보안에 관련된 점검과 대책을 추가함  
전 과정에서 보안을 고려 + 보안 교육

* Software Security != Security Software  
  Software Security => 어떤 sw를 개발/구현 할 때 모든 단계에서 보안을 내재화  
  Security Software => SW로 개발된 보안 도구, 보안제품/서비스가 sw로 개발됨

* Types of Cybersecurity attacks  
  -Phising  
  범죄자가 희생자를 믿게만들고 범죄를 수행함(주로 링크 클릭)  
  -Pharming  
  정상사이트를 위조하여 가짜 사이트를 만듦  
  -중간자 공격  
  송신자가 보낸 내용을 가로채서 변조함, 네트워크 통신을 조작하여 통신 내용을 도청하거나 조작하는 공격기법  
  -Drive by download attack  
  희생자 모르게 어떤 악의적인 프로그램이 설치되게함(동의없이 다운 or 설치)  
  -Malvertising  
  인터넷 광고로 위장하여 악성코드를 설치함(광고를 보면 악성코드가 설치됨) malware + advertising  
  -Rogue antiviurs  
  백신 프로그램을 사칭하여 결제를 유도함

정리)  
What is the difference between information security and computer  
security?

=> information security : 어떤 형태의 정보든지에 상관없이 관련된것

computer security => 디지털 형태의 정보만 관심을 가짐

What is the difference between software security and security software?

software security => 개발 단계에서부터 보안을 신경써서 소프트웨어를 개발하는것

security software=> 악성코드를 탐지하는 소프트웨어

What is malvertising?

=>광고처럼 보이는 공격

What is rogue software?

=>악성코드가 있는것 처럼 속이고 결제를 요구하는 소프트웨어(가짜백신)

* Good Security Standards follow the 90/10 rule  
  보안 문제의 10%는 기술적으로 해결가능하지만, 90%sms 사용자에게 달려있다. 보안 문제는 모든사람의 책임이라는 의미이다.
* Interruption,Interception, Modification, Fabrication Security threats 예시  
  Interruption(가로막기) => TCP flood, Ping flood, DoS DDoS  
  Interception(가로챔) => packet sniffing, key logging  
  Modification(변조) => android 리패키징 공격, 임베디드 시스템의 펌웨어 변조  
  Fabrication(위조, 가장함) => DNS Spoofing, Replay attack

암호알고리즘, 프로토콜, 접근제어는 소프트웨어로 구현이됨  
따라서 소프트웨어가 만약에 문제가 생기면 보안취약점으로 연결이되어 보안사고를 유발함

* 데이터 기밀성을 보장하는 법?
  + 암호기법  
    AES(대칭키), RSA(비대칭키)
  + 강한 접근제어사용  
    Never access(중요데이터, 관리자가 아닌경우), No read, No view
  + 중요한 데이터가 임의의 장소에서 나타나지 않게 함
* 데이터 무결성을 보장하는 법?
  + documenting system activity  
    로그로 남김
  + 강한 access control  
    no write , no append or reald only
  + 암호학적 해쉬함수  
    mac, sha-256, md5 => one-way 함수 (역으로 계산 불가) => 해쉬값으로 입력을 찾기 어려움  
    출력값의 길이는 같음 => 고정된 길이 출력  
    입력의 미세한 변경이 출력에 큰 변화를 끼침
* 가용성 보장?
  + anti-DDoS system  
    Content Delivery Networks(CDNs)  
    -> 컨텐츠를 효율적으로 전달하기 위해서 여러 노드를 가진 네트워크에 데이터를 저장하여 제공함  
    인터넷 서비스 제공자에 직접연결되어 데이터를 전송하기 때문에 컨텐츠에 대한 병목을 줄일수 있음(분산저장)  
    Scrubbing center (scrub => 불순물 제거)  
    -> 네트워크 트래픽을 분석하여 악의적인 트래픽을 제거
  + 백업 절차를 잘 만듦  
    데이터 삭제 공격을 막기 위해
* Security Threat
  + 회사나 개인의 자산을 임의로 노출시키거나 변조, 손상, 이용할 수 없게 하는 행동이나 방치해 두는 것
  + 보안 위협의 요소
  + target : 데이터, 정보, sw, hw ... (취약한 자산)
  + agent :의도를 가지고 위협유발 or 비의도적 위협유발을 하는 사람
  + event : 타겟의 취약점을 악용하는 행위 (정보 파괴, 변조, 오남용 등) => 악의적, 우연적
* 취약점  
  계산 로직의 약점, 코드의 약점, 하드웨어 내의 로직 약점 => 악용시 기밀, 무결, 가용성에 영향을 미침  
  약점이 발현되면 취약점임
* => 취약점이 제거되면 위협이 없어짐
* passive and active attack
* Passive attack => interception  
  시스템 자원에 영향을 주지 않으므로 알아차리기 힘듬  
  기밀성 깨짐

Active attack => 시스템 자원을 변경시킴 따라서 탐지하기 용이한 편  
변조, 조작 => 무결성 가용성 깨짐

* Security Threat Model  
  보안 운영을 계획하고 최적화 하는 방법 => 조직에서 위협들을 방어하기 위한 방법  
  보안 위협(취약한 자산이 타겟)을 없애기 위해서 목적을 제시하고 방어계획을 설계해야함어떤 위협이 있는지, 취약한 자산은 무엇이 있는지 그 위혐들에 대해서 어떻게 우리가 대응할 것인지 이러한 것들을 전반적으로 생각하여 최적화 시키는 것임  
  => 위협을 방어하기 위해서 사용할 수 있는 방식이나 수단
  + 위협  
    멀웨어 => 바이러스, 스파이웨어, 애드워어  
    제로데이 취약점 => 아직 패치가 나오지 않은 새로 발견된 취약점  
    DDoS 공격  
    Phising
  + 보안 위협 구성요소
    - 타겟
    - agent
    - event
* 조직이 균형잡힌 방어기법, 끊임없이 진화하느 ㄴ방어기법을 구축할 수 있게 도와줌

-해커들의 창은 기업 및 기관의 방패보다 항상 빠르게 진화하므로 피해를 최소화하기 위해서 모의훈련으로 대비를 해야한다. => 모의훈련 (공격자처럼 생각 = 예방을 잘 할 수 있고 취약점을 잘 찾을 수 있다.)

-방어보다 공격이 유리하다. => 방어는 보든 부분(서버, pc, 스마트폰, 네트워크..)을 방어해야하지만

공격은 취약한 부분 하나만 있으면 된다. (취약점 : 공격에 악용될 수 있는 발판, 약점, 결함, 실수..)

- Software Security가 중요한 이유?

Software는 종류가 다양하고,

----------------------------------------------------------------------------------

λ Systems software : OS, compiler, loader

λ Business software : Payroll, accounting

λ Scientific and engineering software

ν Computer-aided design, simulation, weather prediction, …

λ Internet software: ν B2C: business-to-customer (e.g., amazon.com)

ν Facebook, Google Chrome, …

λ PC software : Spreadsheets, word processing, games, …

λ Embedded software

ν Cars, microwave ovens, cable boxes, light switches,

ν “smart dust”, …

λ Mobile applications

----------------------------------------------------------------------------------

컴퓨터 보안의 많은 것들은  SW로 구현이 된다.

알고리즘, access control 등 대부분이 SW로 구현되기 때문에 Software Security가 중요하다.

따라서 구현할 때 쓰는 sw가 취약하면 의미가 없으므로, SW가 취약하면 보안은 무조건 깨진다.

만약, 강력한 암호 알고리즘을 도입한다해도 구현하는 SW가 문제 있다면 그 보안은 의미가 없을 것이다.

따라서 SW는 빈약한 보안의 기초가 될 수 있다.

(SW가 빈약하면 보안 자체의 기초가 흔들린다.)

\* Software 위기의 원인

- 예산 초과

- 시간 초과

- 비효율적인 SW

- 저품질의 SW

- 때때로 요구사항을 만족 못함

- 관리불가(유지보수가 힘듬 (코드가 복잡하므로))

- 원하는 결과를 내놓지 않음 or 소스코드를 넘겨주지 않음(문제 확인이 어려움)

이러한 문제를 해결하기 위해 Software Engineering이 나오게 되었다.

\* Software

Software는 실행가능한 프로그램이고, 소스코드, 라이브러리, 문서(유저 요구사항, 실제 명세서, 가이드/메뉴얼 등)이다.

그 중 핵심 기능은

데이터를 처리, 전송, 저장하는것이고

정보를 생산, 관리, 드러내는 것이다.

\* Bugs, Defects, Weaknesses, and Vulnerabilities

----------------------------------------------------------------------------------

Improper initialization

Side effects

Scoping

Operator precedence

Divide-by Zero

Infinite loop

Type confusion (illegal downcasts)

Deadlock

Integer Overflow / Underflow

Memory leak

Use-after-free

Buffer overflow = Buffer overrun

Time-of-check-to-time-of-use flaw

Format string bug

----------------------------------------------------------------------------------

위와 같이 다양한 종류가 있다.

취약점 보단 덜 위험하지만 문제를 일으킬 수 있다.

- bug와 취약점의 차이

bug : 의도치 않은 방향으로 잘못 행동하는 프로그램 내의 결함이다.

취약점 : bug 중에서도 공격자에 의해서 악용될 수 있는 것, 공격자가 접근할 수 있어야하고 공격자가 공격할 능력이 있어야하고 시스템 내에서 결함이 있어야한다.

**SW Bugs 예시)**

**-------------------------------------------------------------**

**(초기값)**

typedef unsigned int uint;

int getmin(int \*arr, uint len){

int min;

for(int i=0;i<len;i++)

min = (min < arr[i]) ? min : arr[i];

return min;

}

min에 대한 초기값이 설정되어 있지않아 정상적으로 실행되지 않을 수 있다.

inn min = 0과 같이 초기 값을 설정해주면 된다.

**(side effects)**

if(foo == 12 || (bar = 13))

baz == 12;

bar = 13은 항상 참이다.

foo와 baz의 값이 선언되어있지 않고 비교되고 있다.

**(범위)**

int a;

void calc(int b){

int a = b\*12;

if(b+24 == 96)

a = b;

}

printf("a=%d\n", a);

=> 전역변수, 지역변수 둘다  a로 선언하여 모호하다.

=> calc함수내의 a는 함수가 종료되면 없어지므로 전역변수 a의 값이 출력된다.

**(control flow)**

int x,y;

for(x=0;x<xlen;x++)

for(y=0;y<ylen;y++);

pix[y\*xlen + x] = x\*y;

두번째 for문에 ;처리하여 원하는 결과가 출력되지 않는다.

if (isbad(cert))

goto fail;

if (invalid(cert))

goto fail;

goto fail;

L10 : printf("Hello, world\n");

goto L10

goto fail;

goto fail;

을 하면 if문을 통과하고 두번째 goto fail이 항상 실행된다.

goto를 잘 못 사용하면 무한루프(L10으로 계속 이동)에 걸릴 수 있다.

**(control flow - loop)**

float x = 0.1;

while(x!=1.1){

x=x+0.1;

printf("X=%f\n", x);

}

부동 소수점은 ==으로 비교하면 안된다.

실수는 무한히 많은데 이 실수를 유한 개의 비트로 표현하기 위해서는 근삿값으로 표현해야 하기 때문이다. =>부동소수점 반올림 오차

#include <stdio.h>

#include <float.h> // float의 머신 엡실론 값 FLT\_EPSILON이 정의된 헤더 파일

#include <math.h> // float의 절댓값을 구하는 fabsf 함수를 위한 헤더 파일

int main()

{

float num1 = 0.0f;

float num2 = 0.1f;

// 0.1을 10번 더함

for (int i = 0; i < 10; i++)

{

num1 = num1 + num2;

}

// num1: 1.000000119209290

if (fabsf(num1 - 1.0f) <= FLT\_EPSILON) // 연산한 값과 비교할 값의 차이를 구하고 절댓값으로

// 만든 뒤 FLT\_EPSILON보다 작거나 같은지 판단

// 오차가 머신 엡실론 이하라면 같은 값으로 봄

printf("true\n"); // 값의 차이가 머신 엡실론보다 작거나 같으므로 true

else

printf("false\n");

return 0;

}

FLT\_EPSILON(머신 엡실론)을 사용해서 오차를 감안하여 실수를 비교해야한다.

어떤 실수를 가장 가까운 부동소수점 실수로 반올림하였을 때 상대 오차는 항상 머신 엡실론 이하이다.

머신 엡실론은 반올림 오차의 상한값이며 연산한 값과 비교할 값의 차이가 머신 엡실론보다 작거나 같다면 두 실수는 같은 값이라 할 수 있다.

double, long double을 사용한다면 머신 엡실론은 DBL\_EPSILON, LDBL\_EPSILON을 사용한다.

1. 연산한 값과 비교할 값의 차이를 구한다.

(값의 차이는 math.h 헤더 파일의 fabsf 함수를 사용하여 절댓값으로 만드므로 차이가 음수여도 상관없다.)

2. FLT\_EPSILON 보다 작거나 같은지 판단

3.연산한 값과 비교할 값의 차이가 머신 엡실론보다 작거나 같다면 두 실수는 같은 값이라 판단

**(control flow - loop)**

int k = 1

int val = 0;

while (k = 10) {

val++;

k++;

}

printf (“k = %d, val = %d \n”, k, val);

while(k=10)

k=10은 항상 참이므로 무한 루프가 된다.

**(Null pointer 역참조)**

int \*ptr = NULL;

printf(“Value of ptr: %d\n”, ptr);

int \*p = 0; //NULL 대입

\*p = 1; //NULL 포안터 역참조, 주소 0에 접근하는 포인터생성후 값 할당 시도함

/\* -------------------------------------\*/

int length;

char \*buff;

scanf (“%d”, &length);

buff = (char \*) malloc(length+1); // always Not NULL?

strcpy(buff, “Hello World! Welcome!”);

첫번째 코드는 포인터 변수 P에 NULL을 대입하고 주소가 0인 포인터에 값을 할당하려하였다.

따라서 세그멘테이션 결함이 발생한다.

if(p==NULL){

     exit(1);

}

과 같이 NULL 포인터를 체크해주어야한다.

두 번째 코드는

malloc시 시스템에 메모리가 부족하거나 메모리 할당 조건이 맞지않아 메모리 할당을 하지 못하면 null을 반환하게 된다. 따라서 buff에 null 포인터가 들어가게된다. 따라서 malloc이 null을 반환하였는지 체크하여야한다.

void Pointer(int \*ptr) {

\*ptr = \*ptr + 5;

}

main(void) {

int num = 10;

Pointer(&num);

Pointer(NULL);

}

Pointer(NULL)

null을 인자로 넘겨주면 Pointer함수에서 null에 값을 대입하려하여 null 포인터 역참조 에러가 발생한다.

**(연산자 우선순위)**

node \*find(node \*\*curr, val){

while(\*curr != NULL)

if(\*curr->val == val) return \*curr;

else

\*curr = \*curr->next;

}

(\*curr)->val로 하여야한다.

#include<stdio.h>

void main(){

int x,a,b,c,d,e,f;

a=7;b=6;c=5;d=4;e=3;f=2;

x = a&b+c\*d&&e^f==7;

printf("X = %d\n", x);

}

x = 1

연산자 우선순위는

(\*) -> (+) -> (==) -> (&) -> (^) -> (&&)

이다.

따라서

(a&(b+(c\*d)))&&(e^(f==7))가 된다.

(a&(b+(c\*d))) => 7&(6+(5\*4)) =  20&7 (10100 & 00111) => 100(2) = 4

(e^(f==7)) => 3^(2==7) => 11 xor 00 = 11(2) => 3

3&&4는 참이므로 1이된다 (3과 4는 참이므로)

따라서 x에는 1이 들어가게된다.

**(Integer Security)**

char cresult, c1, c2, c3;

c1 = 100; c2 = 90; c3= -120

cresult = c1 + c2 + c3;

printf(“%c, %d, %c, %d\n”, cresult, cresult, c3, c3);

changmin@ubuntu:~/Desktop/c$ ./d  
F, 70, �, -120

char형은 8비트이다.

c1+c2=190에서 char범위를 벗어난다.

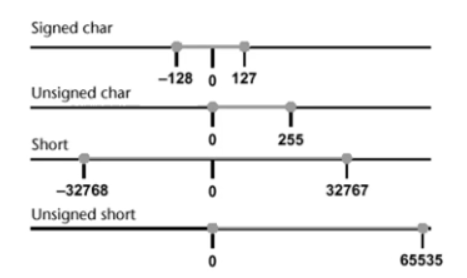
임시적인 정수 승격으로 c1, c2, c3는 int형으로 변환되고

(임시적인 정수승격 : 일반적으로 CPU가 처리하기에 가장 적합한 크기의 정수 자료형은 int형이다. 즉, int형 연산의 속도가 다른 자료형의 연산속도에 비해서 동일하거나 더 빠르다. 따라서, int보다 작은 크기의 정수형 데이터는 int형 데이터로 형 변환이 되어서 연산된다.)

계산이 끝난뒤에 값이 잘리게 된다.

따라서 c1+c2+c3는 70이된다.

char 범위는 -128~127이므로 70이 출력된다.



short s1 = 32000;

short s2 =1500;

s1 = s1 + s2;

printf("%h, %d\n", s1, s1);

changmin@ubuntu:~/Desktop/c$ ./e  
%, -32036

(%h =>short 형으로 출력,  invalid conversion specifier -Wformat-invalid-specifier 라고 되는데 %h 형식을 지원하지 않는듯 하다..... short형은 %hd를 사용해야하는 것 같다.)

32000+1500은 short범위를 벗어나므로 오버플로우가 발생한다.

**(out-of-bounds write)**

텍스트이(가) 표시된 사진

자동 생성된 설명

배열의 크기를 벗어나거나

텍스트이(가) 표시된 사진

자동 생성된 설명

memcpy시 마지막 인자가 unsigned int형으로 변환되면서 언더플로우가 발생될 수 있다.

(returnChunckSize(desBuf)의 반환값이 -1일때 -1-1은 -2가 되고 unsigned int로 변환시 언더플로우발생으로 매우 큰수가 된다.)

2017년 우크라이나 사이버 공격

◼ 2017년 6월 27일 Petya 악성코드를 사용한 일련의 강력한 사이버 공격이 시작되었습니다. 은행, 부처, 신문과 전기 회사.

⚫ ESET은 2017년 6월 28일에 모든 감염의 80%가 독일과 우크라이나에서 발생한 것으로 추정했습니다. 약 9%로 두 번째로 큰 타격을 받았습니다. ⚫ AP 통신은 전문가들이 Petya가 랜섬웨어로 가장하고 있다는 데 동의했지만, 실제로 우크라이나가 주요 목표로 최대 피해를 입히도록 설계되었습니다. ⚫ Petya는 하드 드라이브의 마스터 파일 테이블을 암호화하고 컴퓨터를 강제로 다시 시작합니다.

⚫ 그런 다음 사용자에게 메시지를 표시하여 파일이 이제 암호화되었음을 알리고 컴퓨터 암호 해독 지침을 받기 위해 3개의 지갑 중 하나에 비트코인 300달러

**Not petya 일종의 사이버무기, MEDoc(세무회계 sw)의 업데이트 기능을 통해 감염되어 확산됨.**

소프트웨어 크래킹은 기능을 제거하거나 비활성화하기 위해 소프트웨어를 변조하는 것입니다.

특히 복사 방지 기능(소프트웨어 조작에 대한 보호 포함, 일련 번호, 하드웨어 키, 날짜 확인 및 디스크 확인) 또는 잔소리와 같은 소프트웨어 성가심 스크린과 애드웨어.

⚫ 크랙(crack)은 획득 수단을 의미합니다. 예를 들어 도난당한 일련 번호 또는

크래킹 작업을 수행하는 도구입니다.

◼ 이러한 도구 중 일부를 키젠(keygen), 패치(patch) 또는 로더(loader)라고 합니다.

◼ 키젠(keygen)은 종종 기능을 제공하는 수제 제품 일련 번호 생성기입니다.

자신의 이름으로 작동하는 일련 번호를 생성합니다.

◼ 패치는 다른 컴퓨터의 기계어 코드를 수정하는 작은 컴퓨터 프로그램입니다.

프로그램.

⚫ 이것은 크래커가 릴리스에 큰 실행 파일을 포함하지 않는 이점이 있습니다.

몇 바이트가 변경됩니다.

◼ 로더는 프로그램의 시작 흐름을 수정하고 보호 기능을 제거하지 않습니다.

그러나 그것을 우회합니다.

⚫ 로더의 잘 알려진 예는 게임에서 속임수에 사용되는 트레이너입니다.

Q. How does the malware work?

"맬웨어"라는 용어는 전자 장치의 정상적인 작동을 방해하거나 조작하는 유해한 소프트웨어를 말합니다. 1   맬웨어는 개인용 컴퓨터, 스마트폰, 태블릿, 서버 및 장비까지 감염시킬 수 있습니다. 기본적으로 컴퓨팅 기능이 있는 모든 장치입니다.

동작하는 방법은 맬웨어는 일반적으로 사용자가 인터넷에서 해서는 안 되는 프로그램을 클릭 및 또는 설치하도록 속여서 컴퓨터를 감염시킵니다. 클릭 또는 설치가 발생하면 악성 코드는 사용자가 예상하거나 의도하지 않은 작업을 실행합니다.

Q. What can Alice and Bob do to protect themselves from malware?

1) 정기적으로 운영체제와 설치된 프로그램을 업데이트하고, 2) 바이러스 방지 프로그램을 설치하며, 3) 방화벽 프로그램을 설치하고, 4) NoScript 부가프로그램과 함께 파이어폭스 같은 안전한 브라우저를

Q. What can Trudy do to make malware more “effective” ?

- malware가 탐지되지 않도록 빨리 전파되도록 해야함.