컴퓨터 보안_01

실습 11주차

동국대학교 CSDC Lab.
실습조교 김선규
2023.05.17 (Wed.)

Index

1. Project#3 - 시스템 해킹 테스트 (2주 진행)

- 주 제 1 : 백도어 프로그램 개발 및 실행하기
- 주 제 2 : 레지스트리 정보 조회 및 갱신하기
- 주 제 3 : 스택기반 오버플로우 테스트하기
- 주 제 4 : SEH 기반 오버플로우 테스트하기

2. Q & A

➢ 주 제 1 : 백도어 프로그램 개발 및 실행하기

- 방화벽은 외부에서 내부 서버에 접근하는 것을 차단하고 있어서 Telnet, FTP같은 서버 접근이 필요한 서비스는 허가된 사용자들만 사용할 수 있다. 하지만 일반적으로 방화벽의 내부에서 외부로 향하는 서비스는 잘 차단하지 않는다. 즉, 방화벽의 안으로 들어가는 것은 힘들지만, 일단 침입에 성공하면 정보를 빼내기는 쉬운 일이다. 백도어는 방화벽과 같은 보안 장비를 우회해서 공격 대상이 되는 서버PC의 자원을 통제하는 기술이다. 일단 대상 서버PC에 백도어 클라이언트가 설치되면 백도어 서버는 방화벽의 외부에서 백도어 클라이언트에게 명령을 지시하고, 백도어 클라이언트는 대상 서버PC에서 이 명령을 수행하고 이 결과를 다시 방화벽 외부의 백도어 서버로 전달하게 된다.

- 백도어를 이용한 해킹에서 가장 어려운 부분은 백도어 클라이언트를 설치하는 것이다. 일반적으로 네트워크를 통해서 대상 서버PC로 백도어 클라이언트를 직접 업로드 하기는 쉽지 않기 때문에 보안이 상대적으로 취약한 웹 환경을 많이 활용한다. 가장 보편적으로 사용되는 것이 게시판 파일 업로드 기능을 활용하는 것이다. 해커는 유용한 도구나 동영상으로 가장해서 악성코드가 담긴 파일을 게시판에 올리고, 대상 서버PC 사용자는 무심코 클릭해서 해당 파일을 내려 받는다. 파일을 클릭하는 순간 사용자는 자신도 모르게 서버 PC에 백도어를 설치하게 되고 서버 PC는 원격에서 조정할 수 있는 좀비 PC가 된다. 호기심을 자극하는 문구의 e메일 또한 백도어 공격의 수단으로도 많이 사용된다.

- PC에 설치된 바이러스 백신은 대부분의 백도어를 검출할 수 있지만, 백도어의 강력한 기능을 원하는 해커들은 백신에 검출되지 않는 형태의 악성코드를 지속적으로 만들어내고 있다. 간단한 파이썬 프로그램을 통해서 백도어의 개념을 알아보고 PC에 저장된 개인정보를 검색하는 명령어를 사용해서 백도어의 위험성을 확인해보자.

주 제 1 : 백도어 프로그램 개발 및 실행하기

- 백도어는 서버와 클라이언트로 구성된다. 서버는 해커PC에서 실행되고 클라이언트는 서버PC에서 실행된다. 먼저 해커PC에서 백도어 서버가 구동되고 서버PC에 설치된 백도어 클라이언트가 실행되면서 백도어 서버에 접속한다. 그러면 백도어 서버는 백도어 클라이언트에 명령을 보낼수 있게 된다. 백도어는 개인정보 검색, 레지스트리 정보검색, 계정 비밀번호 변경 등의 다양한 치명적인 공격을 수행할 수 있다.

- 백도어 클라이언트는 서버로부터 전달받은 텍스트 형태의 명령어를 프로세스로 만들어서 실행한다. 이때 프로세스를 생성하고, 명령어를 전달하고, 실행결과를 다시 백도어 클라이언트로 돌려주는 기능을 위해서 subprocess.Popen클래스가 사용된다. 실제로 해킹 대상인 서버PC에 파이썬이 설치되어 있다고 보기 어렵다. 따라서 백도어 클라이언트를 실행하려면 파이썬 프로그램을 윈도우의 실행파일로 변환해야 한다.

- 백도어 테스트를 위해 p.268를 참고하여 testfile.txt라는 파일을 만들어 공격대상 서버PC의 C:\test 폴더에 저장하고, backdoorClient.exe파일은 C:\ 디렉토리 바로 아래에 저장한다. 즉, 가능한 시나리오는 백도어 프로그램은 e메일을 통해서 배포되었고, 서버 사용자인 A씨가 e메일을 읽다가 실수로 백도어 프로그램을 서버PC에 설치했다고 가정하자.

주 제 1 : 백도어 프로그램 개발 및 실행하기

dongguk UNIVERSITY

➢ **[과제 내용]**

- 아래의 각 문제에 대한 파이썬 코드의 수행 결과를 캡처하여 분석하고 설명하는 내용을 개인적인 견해와 함께 레포트에 모두 반영하여야 한다. (참고자료의 테스트에는 Windows11, Python 2.7.18, py2exe-0.6.9.win32-py2.7.exe가 사용되었다).

1. p.263의 예제8-1을 참고로 하여 backdoorServer.py를 해커의 PC에서 실행시켜보아라.

2. 이제 첨부자료 p.266 ~ p.267를 참고로 하여 예제8-2의 backdoorClient.py를 backdoorClient.exe로 변환하여 보아라.

3. 1번의 결과로서 해커PC에서 이미 backdoorServer.py프로그램을 실행시켜 놓았다고 가정하고 이어서 공격 대상이 되는 서버PC에서 backdoorClient.exe를 실행시켜보자. 이때 해커PC의 콘솔화면에 어떤 정보가 보이는지 결과를 확인해보자.

주 제 2 : 레지스트리 정보 조회 및 갱신하기

dongguk UNIVERSITY

➢ 주 제2 : 레지스트리 정보 조회 및 갱신하기

- 윈도우에서 파이썬을 사용하여 자동으로 사용자 계정 목록을 조회하는 프로그램을 만들어 보자(예제8-4 registryuserList.py 참고). p.271~p.274를 참고하여 레지스트리 서브 디렉토리를 지정하고, 관심 있는 정보를 추출하기 위해 약간의 프로그램 코드를 추가하면 시스템에서 사용하는 사용자 목록을 쉽게 추출할 수 있다. 레지스트리 검색을 통해 추출한 사용자 계정 정보는 시스템 해킹을 위해 유용하게 사용된다. 사전공격(Dictionary Attack)을 이용해서 사용자 비밀번호를 추출할 수도 있고 win32com 모듈에서 제공하는 adsi클래스를 사용하면 직접 비밀번호를 변경할 수 있다.

- 사용자가 인터넷 익스플로러 주소창에 입력한 URL은 레지스트리 특정 위치에 기록된다. 해커는 인터넷 사용기록을 조회해서 사용자의 생활 패턴을 유추할 수 있다. 만일 전자상거래 사이트를 자주 접속한다면 개인 정보를 탈취하기 위한 프로그램을 심어 놓을 수 있고, 개인의 성향을 파악하는 기본 자료로 활용할 수 있다. 윈도우 방화벽 관련 설정 정보 역시 레지스트리에 보관된다. 방화벽 사용/해제 정보, 방화벽 상태 알림 정보, 시작 프로그램 추가 여부, 방화벽 정책 설정 정보, 등록 애플리케이션 정보 등 다양한 정보가 저장된다. 레지스트리 값 변경을 통해 간단하게 방화벽 사용을 해제하는 예제를 만들어보자 (예제 8-5 registryFirewall.py).

- 레지스트리에 다양한 값을 입력함으로써 시스템 설정에 많은 영향을 미칠 수 있다. 보안 설정 변경을 위해 방화벽이 허용하는 서비스 목록을 임의로 등록할 수 있고, 인터넷 익스플로러나 워드프로세서와 같은 애플리케이션 설정을 프로그램을 통해 바꿀 수 있다.

주 제 2 : 레지스트리 정보 조회 및 갱신하기

dongguk UNIVERSITY

➢ **[과제 내용]**

- 아래의 각 문제에 대한 파이썬 코드의 수행 결과를 캡처하여 분석하고 설명하는 내용을 개인적인 견해와 함께 레포트에 모두 반영하여야 한다. (참고자료의 테스트에는 Windows 11, Python 2.7.18이 사용되었다).

1. 예제8-4 registryuserList.py 를 실행해 보고 그 결과를 나타내어 보아라.

2. 예제 8-5 registryFirewall.py를 실행해서 윈도우 방화벽 설정을 변경해 보자
   (이때 방화벽 관리프로그램이 레지스트리 정보를 강제적으로 읽도록 명령 해야하는데, 가장 단순한 방법은 윈도우를 다시 시작하는 것이다).

➢ 주 제3 : 스택기반 오버플로우 테스트하기

- 스택기반 버퍼 오버플로우(Stack Based Buffer Overflow)기법은 레지스터의 특징을 활용한다. 입력 값을 반복적으로 변경하면서 애플리케이션을 공격하는 퍼징(Fuzzing)을 통해 버퍼 오버플로우를 유발한다. 해당 시점의 메모리 상태를 디버거를 통해 관찰하면서 의도하는 결과를 유발하는 입력값을 찾는 방식이다. 스택기반 오버플로우 기법에서는 IA-32 CPU내의 9개의 레지스터 중에서 EIP와 ESP 레지스터를 핵심적으로 사용하는데, 첫번째 목적이 두 레지스터를 입력값으로 덮어쓰는 것이다. 얼마나 많은 양의 데이터를 애플리케이션에 입력해야 EIP와 ESP값을 조작할 수 있는지 찾아야 한다. 두번째 해야할 것은 애플리케이션 실행 흐름을 ESP레지스터로 옮길 수 있는 명령어 주소를 찾는 것이다. 마지막으로 입력값에 해킹코드를 붙여서 해킹을 실행한다(그림8-13과 그림8-14 참고).

- ESP가 가리키는 스택 영역에 해킹 코드를 삽입한다. 입력값으로 들어온 'jmp esp' 명령어의 주소를 EIP레지스터에 입력한다. 버퍼 오버플로우가 발생하는 시점의 프로그램 실행은 EIP레지스터 주소를 참조한다. 즉, 'jmp esp' 명령이 실행된다. ESP레지스터에는 해킹 코드가 들어있기 때문에 해커가 의도하는 동작이 수행되도록 할 수 있다.

- http://www.exploit-db.com/ 사이트에 가면 다양한 취약점 악용사례를 볼 수 있다. BlazeDVD Pro player 6.1 프로그램의 취약점을 이용한 http://www.exploit-db.com/exploits/26889 사례를 참조한다. 사이트에서 해킹 소스 코드(Exploit Code)와 대상 애플리케이션(Vulnerable App)을 모두 내려 받을 수 있다. BlazeDVD Pro player는 plf파일을 읽어서 실행하는 프로그램이다. a문자를 반복적으로 넣은 plf 파일을 만들어서 퍼징을 시도해본다. 먼저 a문자의 16진수 코드에 해당하는 \x41을 넣어서 파일을 만든다 (예제8-6 fuzzingBlazeDVD.py참고).

주 제 3 : 스택기반 오버플로우 테스트하기

dongguk UNIVERSITY

- 퍼징에 성공했다면 메모리 상태를 점검할 수 있는 디버거를 만들어보자 (pydbg모듈을 사용한다). 먼저 BlazeDVD Pro player를 실행해야 디버거가 정상 작동한다. 즉, BlazeDVD Pro player를 실행하고 나서 그 다음 bufferOverflowTest.py를 실행하고 blazeExpl.plf를 열어본다. 파일이 열리자마자 애플리케이션은 종료되고 디버거는 메시지를 출력하게 된다. bufferOverflowTest.py 디버거에 들어가는 프로세스의 이름은 작업 관리자를 실행해서 [프로세스] 탭을 살펴보면 확인할 수 있다.

- 퍼징을 위해서 입력한 문자는 연속된 동일 문자이다. 따라서 어느 정도 길이의 문자를 입력했을 때 EIP에 데이터가 들어가는지 정확히 알 수 없다. 일정한 규칙을 가진 문자열을 입력함으로써 데이터의 흐름을 추적해보자. 간단한 테스트를 위해 텍스트 에디터를 사용해서 패턴을 생성해보자 (p.288 예제 8-8 fuzzingBlazeDVD.py 참조). 그 다음엔 명령어를 저장할 ESP레지스터의 값을 채워보자. 같은 방법으로 테스트하는데 먼저 260바이트까지는 오버플로우를 유발하는 데이터이고 그 다음 4바이트는 EIP의 주소이다. 앞의 260바이트는 a로 채우고 뒤의 4바이트는 b로 채운다. 마지막으로 테스트 문자열을 붙여서 디버깅해 본다 (예제 8-9 fuzzingBlazeDVD.py 참고).

- 메모리에 로딩된 명령어 중에서 'jmp esp'를 찾아서 해당 주소를 가져와야 한다. 다양한 기법이 있지만 가장 간단한 방법은 findjmp.exe를 이용하는 것이다. 해당 프로그램은 인터넷 검색을 통해 쉽게 찾을 수 있다. 윈도우 창을 열어서 findjmp.exe.가 위치한 디렉토리로 이동한 후 명령을 내리면 된다 (예제 8-10 jmp esp 명령어의 주소찾기).

주 제 3 : 스택기반 오버플로우 테스트하기

dongguk UNIVERSITY

➢ [과제 내용]

- 아래의 각 문제에 대한 파이썬 코드의 수행 결과를 캡처하여 분석하고 설명하는 내용을 개인적인 견해와 함께 레포트에 모두 반영하여야 한다. (참고자료의 테스트에는 Windows XP SP1, Python 2.7.6, VirtualBox (4.3.10 r93012)가 사용되었다).

1. 예제8-6 fuzzingBlazeDVD.py를 실행시켜 blazeExpl.plf파일을 생성하고 BlazeDVD Pro player 애플리케이션을 실행시켜 이 파일을 열어보아라. 무슨 일이 발생하는가?

2. BlazeDVD Pro player 애플리케이션을 실행시키고 나서 예제 8-7 bufferOverflowTest.py 디버거를 실행시키는 순서로 진행된다. BlazeDVD Pro player가 blazeExpl.plf파일을 열자마자 애플리케이션은 종료되고 디버거는 메시지를 출력한다. 이 메시지의 내용을 분석하여 보아라.

3. 예제 8-8 fuzzingBlazeDVD.py를 가지고 blazeExpl.plf를 생성하여 2번의 방식과 동일한 디버깅을 해보자. 어떤 메시지가 출력되는가? CONTEXT DUMP부분의 EIP 레지스터에는 어떤 값이 들어가 있는지 확인해 보아라. 이 값이 무엇을 의미하는가?

4. 예제 8-9부터 예제 8-10까지의 내용이 반영된 예제 8-11을 참고하여 해킹에 필요한 문자열 입력값을 모두 완성하고, 가능하다면 VirtualBox기반의 윈도우 XP SP1에서 테스트해 보아라 (Window 7이상에서는 오버플로우 공격 방지 메커니즘이 추가되어 있어서 오버플로우 공격이 동작하지 않는다).

주 제 4 : SEH 기반 오버플로우 테스트하기

dongguk UNIVERSITY

➢ 주제 4 : SEH 기반 오버플로우 테스트하기

- SEH(Structured Exception Handler)의 개념을 알아보자. 윈도우 운영체제에서 제공하는 예외처리 메카니즘이다. SEH 는 연결 리스트(linked list)로 연결된 체인구조로 되어 있다. 예외가 발생하면 운영체제는 SEH 체인을 따라가면서 예외를 처리하는 함수를 발견하면 차례대로 실행하고 예외처리 함수가 없으면 건너뛰면서 예외를 처리한다. 마지막 체인은 Next SEH 가 0xFFFFFFFF 를 가리키게 되고, 예외처리를 커널로 넘겨준다. 개발자 수준에서 모든 예외를 처리할 수 없는 현실적인 문제를 해결하고 애플리케이션이 더욱 안정적으로 운영될 수 있도록 지원한다. Windows 7 에서는 SEH 를 활용하여 버퍼 오버플로우 공격을 차단하기 위한 다양한 기술을 개발해왔다. 이제 SEH 버퍼 오버플로우 기법에 대해서 간단히 알아보고 Windows 7 에 적용된 보안 기술을 우회해서 해킹하는 방법을 알아본다.

- 먼저 퍼징을 통해서 애플리케이션 오류를 발생시키고 디버깅을 이용해서 해킹 코드를 하나씩 만들어보자. 일반적인 절차는 스택 기반 버퍼 오버플로우와 유사하지만, EIP 를 겹쳐쓰는 대신에 SEH 를 겹쳐 써서 해킹을 시도하는 것이다. 퍼징을 통해서 어느 정도 길이의 문자열을 입력했을 때 SEH 를 겹쳐 쓰는지 찾아낸다. 디버거를 이용하여 'POP POP RET' 명령어의 주소를 찾아내서 SEH 의 위치에 해당주소를 입력한다. Next SHE 에 'short jmp' 명령어에 해당하는 16 진수 코드를 입력하면 사용자가 입력한 셸코드를 실행하는 아드레날린 실행파일이 완성된다.

주 제 4 : SEH 기반 오버플로우 테스트하기

dongguk UNIVERSITY

- 샘플코드와 테스트 대상이 되는 애플리케이션은 http://www.exploit-db.com/exploits/26525/ 사이트에서 내려받고 디버거는 bufferOverflowTest.py 를 그대로 사용한다. 다만 processName 변수에 'BlazeDVD.exe' 대신 'Play.exe'를 입력한다. 예제의 동작방식은 fuzzingBlazeDVD.py 와 유사하게 임의의 길이의 연속된 A 문자를 가진 아드레날린 실행파일을 만든다 (예제 8-12 의 fuzzingAdrenalin.py 참고).

- 일정한 규칙을 가진 문자열을 생성해서 몇 번째 값이 SHE 를 겹쳐 쓰는지 확인해보자. 문자열은 a~z 그리고 0~9 까지의 문자를 가로와 세로로 교차해서 임의로 생성할 수 있다 (예제 8-13 의fuzzingAdrenalin.py 참고). pydbg 모듈로 해당 명령어를 찾기는 쉽지 않다. 편의를 위해서 OllyDbg 디버거를 다음 사이트 (http://www.ollydbg.de/download.htm)에서 내려 받는다. zip 파일을 받아서 압축을 풀면 별도의 설치 과정 없이 디버거를 사용할 수 있다. 아드레날린 플레이어를 먼저 실행한 후 OllyDbg 를 실행한다. OllyDbg 상단 [File]메뉴에서 첨부(Attach)기능을 사용해 보자. Play.exe 파일을 찾아서 첨부한다 (그림 8-19 참고).

- 일반적으로 Windows 디렉토리 이외에 애플리케이션 영역 DLL 이 취약점이 많으므로 여기서는 AdrenalinX.dll 파일을 선택해서 POP POP RET 명령어를 검색해 본다 (그림 8-20 참고). 이제 해킹 프로그램을 완성할 수 있다. 앞부분의 2,140 바이트는 특정문자로 채우고 NextSEH 부분에는 6 바이트만큼 점프하는 16 진수 코드를 입력한다. 그리고 SEH 부분에는 POP POP RET 명령어의 시작주소(그림 8-21 명령어 찾기 참고)를 입력한다. 마지막은 계산기를 실행하는 셸코드를 붙여 넣는다

주 제 4 : SEH 기반 오버플로우 테스트하기

dongguk UNIVERSITY

➢ [과제 내용]

- 아래의 각 문제에 대한 파이썬 코드의 수행 결과를 캡처하여 분석하고 설명하는 내용을 개인적인 견해와 함께 레포트에 모두 반영하여야 한다. (참고자료의 테스트에는 Windows7 professional 32 bits, Python 2.7.6 가 사용되었다).

1. fuzzingBlazeDVD.py 와 유사하게 임의의 길이의 연속된 A 문자를 가진 아드레날린 실행파일을 만든다 (예제 8-12 의 fuzzingAdrenalin.py 참고).

2. 아드레날린 플레이어를 실행하고 나서 bufferOverflowTest.py 를 실행해서 플레이어를 디버깅할 준비를 한다. 마지막으로 플레이어를 통해서 Exploit.wvx 파일을 열게되면 오류가 발생하고 디버거는 메시지를 출력한다. 이때 출력되는 메시지를 분석하여라.

3. 예제 8-13 의 fuzzingAdrenalin.py 를 실행해서 Exploit.wvx 파일을 생성하고, 아드레날린 플레이어를 실행하면 디버거로 오류 상황을 모니터링할 수 있다. 이때 SEH unwind 부분을 분석하여라.

4. 공격실행을 위하여 예제 8-14 의 fuzzingAdrenalin.py 를 실행해서 얻은 Exploit.wvx 파일을 아드레날린 프로그램을 실행시켜서 열게 되면 어떤 결과가 나오는지 분석하여라

Q & A