바코드를 이용하는 기기에서의 보안적 취약점 탐구

박범준
선정고등학교
e-mail:bjpark1000@naver.com

Study on Security Weakness of Barcode Devices

Beom-Joon Park
Sunjung High School

요 약

마트, 식당, 도서관 등 우리 주변 많은 곳에서 바코드가 사용되고 있다. 바코드는 주로 ISBN, Code128, Code39 등의 형식이 쓰이는데 그중 Code 128은 ASCII Code를 기반으로 하기 때문에 ASCII Code 0번부터 32번까지의 제어 문자를 바코드에 담을 수 있다. 제어 문자는 본래 프린터 또는 통신 접속구 등 주변 장치에 정보를 전달하기 위하여 사용되는 문자를 뜻하지만 Windows상에서 입력될 시 전혀 다른 역할을 한다. 주로 바코드 기기에서 입력 값을 검증하지 않으므로 이를 이용해 제어 문자를 담은 바코드를 태깅해 명령 프롬포트를 열고 명령을 실행할 수 있다. 또한 대부분의 바코드 인식 프로그램이 DB를 사용하고, 보안이 다른 프로그램들에 비해 취약하다는 점에서 SQL Injection 공격 가능성을 제시한다.

1. 서론

기술이 발전함에 따라 우리는 상품관리, 학생정보관리 등 삶의 많은 부분을 바코드에 의존하고 있다. 하지만 구내식당, 출석확인같이 보안에 크게 비중을 두지 않는 경우에는 바코드 관리 프로그램이 단순한 기능만을 갖고 있거나 구버전 OS를 사용해 상대적으로 보안에 취약하다. 또한 대부분의 바코드 리더기가 컴퓨터에 키보드로 인식된다는 점을 생각해본다면 별다른 입력 장치 없이도 공격코드를 전송할 수 있어 해킹에 취약할 수 있다. 그런 기기가 내부 네트워크에 연결되어 있는 경우 해킹 당한다면 내부 네트워크에 연결되어있는 다른 컴퓨터들에게도 위협이 될 수 있다. 그래서 해킹 가능성을 알아보기 위해 바코드를 이용하여 ASCII Code의 제어 문자를 이용한 공격을 시도해보았다. 또한 대부분의 바코드 인식 프로그램이 DB를 사용하고, 보안이 다른 프로그램들에 비해 취약하다는 점에서 학교 자습실 입출입 시스템 분석을 통한 SQL Injection 공격 가능성을 제시한다.

2. 제어문자를 이용한 해킹

2.1 제어문자

제어 문자란 컴퓨터 화면에 나타나거나 프린터에 인쇄되지는 않지만 프린터 또는 통신 접속구 등 주변 장치에 정보를 전달하기 위하여 사용되는 문자를 뜻한다.[1] 따라서 제어 문자를 입력하는 것으로 컴퓨터가 특수한 행동을 하게 할 수 있다. 가장 일반적으로 사용되는 제어 문자는 ASCII Code 중 앞부분에 있는 32개를 들 수 있다. [2]

<표 1> ASCII Code 제어문자

10진수	16진수	문자	16	0x16	SLE
0	0x00	NUL	17	0x17	DC1
1	0x01	SOH	18	0x18	DC2
2	0x02	STX	19	0x19	DC3
3	0x03	ETX	20	0x20	DC4
4	0x04	EOT	21	0x21	NAK
5	0x05	ENQ	22	0x22	SYN
6	0x06	ACK	23	0x23	ETB
7	0x07	BEL	24	0x24	CAN
8	0x08	BS	25	0x25	EM
9	0x09	HT	26	0x26	SUB
10	0x10	LF	27	0x27	ESC
11	0x11	VT	28	0x28	FS
12	0x12	FF	29	0x29	GS
13	0x13	CR	30	0x30	RS
14	0x14	SO	31	0x31	US
15	0x15	SI	32	0x32	SP

2.2 해킹 시나리오

제어문자를 이용한 해킹 시나리오는 다음 단계와 같다.

① 제어 문자와 문자와의 조합을 통해 명령 프롬포트를 실행하는 바코드 생성
② 바코드를 태깅하여 명령 프롬포트 실행
③ ftp 명령어를 이용하여 외부 서버에서 백도어를 다운로드하는 바코드 생성
④ 바코드를 태깅하여 백도어 다운로드
⑤ 컴퓨터 장악 후 내부 네트워크 침투

위 시나리오는 인터넷이 연결되어 있는 상태에서의 시나리오이지만 인터넷이 연결되어 있지 않은 상태에서도 얼마든지 데이터를 삭제 또는 변조 할 수 있다.

2.3 바코드 생성

윈도우상에서 제어 문자가 입력될 경우 이들은 원래의 기능이 아닌 다른 기능을 한다. 예를 들어 17번 ETB(End of Transmission Block)은 본래 데이터를 분할 전송할 때 분할된 데이터의 끝부분에 붙여주는 제어 문자이다. 하지만 윈도우상에서 입력될 경우 키보드 상에서 ESC키를 누른 것과 같은 기능을 한다. 제어 문자 32개의 바코드를 생성하고, 태깅한 후 몇 가지 유용한 기능을 가진 제어 문자를 찾을 수 있었다.

I. 13번 CR(Carriage Return)

(그림 1) CR(Carriage Return)

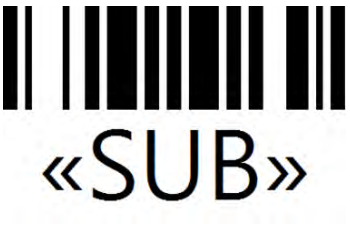
본래는 행의 첫 부분으로 커서를 옮겨주는 역할을 하는 제어 문자이다. 윈도우 상에서 입력되었을 경우에는 키보드의 'Enter'키를 누른 것과 같은 기능을 가진다.

II. 23번 ETB(End of Transmission Block)

(그림 2) ETB(End of Transmission Block)

본래는 전송상의 이유로 분할된 데이터의 끝을 나타내는 전송제어를 위한 제어 문자이다. 윈도우 상에서 입력되었을 경우에는 키보드의 'ESC'키를 누른 것과 같은 기능을 한다.

III. 26번 SUB(SUBstitute)



(그림 3) SUB(SUBstitute)

본래는 전송한 마지막 데이터의 끝 부분을 정정해주는 역할을 한다. 윈도우 상에서 입력되었을 경우에는 키보드의 'ctrl'키를 누르고 있는 것과 같은 기능을 한다.

IV. 27번 ESC(ESCape)

(그림 4) ESC(ESCape)

본래는 확장 문자로 이 문자가 나타나면 그 후의 문자는 모두 특별한 규칙으로 해석한다. 윈도우 상에서는 'SUB' 제어 문자를 취소하는 역할을 한다.

이것들을 종합하여 Windows상에서 명령 프롬포트 창을 여는 바코드를 생성할 수 있다.

- Windows XP용

(그림 5) Windows XP용 명령프롬포트 실행 바코드

- Windows 7용

(그림 6) Windows 7용 명령프롬포트 실행 바코드

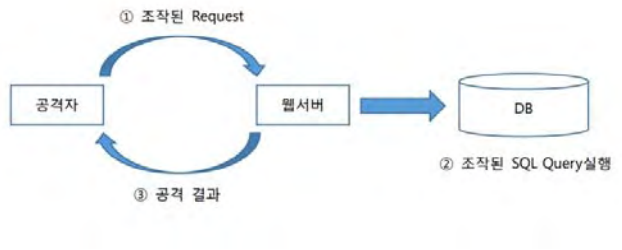
바코드 리더기가 읽을 수 있는 바코드의 길이가 제한 되어있어 바코드를 두 부분으로 나누어서 생성했다. 바코드를 차례대로 태깅하면 명령 프롬포트 창이 열리는 것을 확인할 수 있다. 그 후 백도어를 다운로드하는 바코드를

생성하여 태깅하면 컴퓨터를 장악할 수 있다.

3. SQL Injection을 이용한 해킹

3.1 SQL Injection

다른 공격에 비해 고도의 지식이 필요한 기법이 아니라 쉽게 배울 수 있고, 자동화된 툴이 나올 정도로 간편해서 최근 웹 해킹 중 가장 많은 빈도로 발생하는 공격이다. 웹 환경에서 특정 DB에 값을 입력하거나 삭제하려면 ASP, PHP, JSP등의 스크립트 코드가 사용되는데, 스크립트 코드로 이루어진 웹 어플리케이션에서 DB와 연동된 부분은 크게 로그인, 검색, 게시판으로 나눌 수 있다. 사용자로부터 입력을 받을 때 웹 어플리케이션 코드 상에서 입력 값을 검사하지 않으면 공격자는 SQL Query를 삽입하여 공격할 수 있다. 로그인으로 예를 들면 공격자는 아이디, 패스워드 대신에 특정 SQL문을 삽입하고, 그 SQL문이 그대로 데이터베이스로 전송되어 비정상적인 결과를 일으킨다. [3]

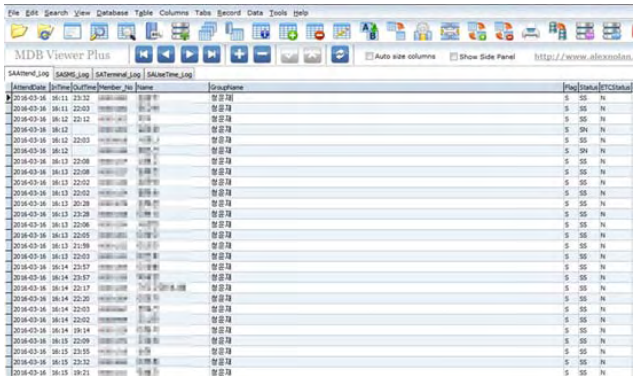


(그림 7) 일반적인 SQL Injection 공격과정

가장 흔한 공격인만큼 스크립트상이나 서버에서 입력 값을 검증하여 공격을 막는 등 대비 방법도 널리 알려져 있어 조금만 더 신경쓴다면 충분히 방어할 수 있다.

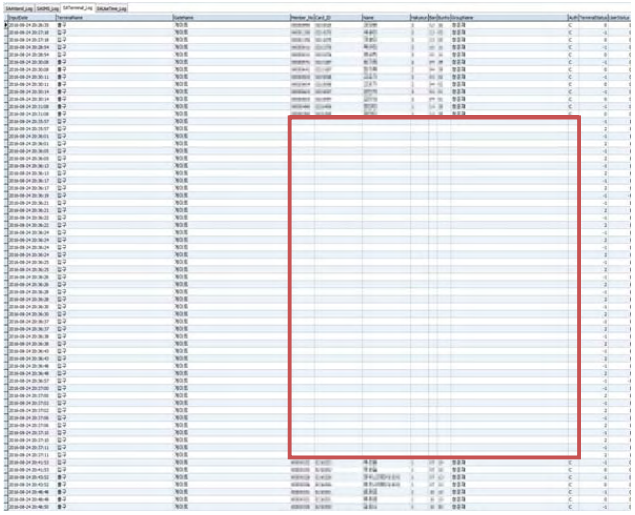
3.2 학교 입출입 시스템

조사를 통해 학교 자습실 입출입 시스템에서 MDB형식의 데이터베이스를 사용하고, 이를 관리하는 응용프로그램을 사용한다는 것을 확인할 수 있었다. 또한 데이터베이스에는 학생들의 바코드 정보가 기록되어있고 학생증 바코드를 태그할 때마다 SAAttend_Log 테이블에 입출입 로그가 기록된다.



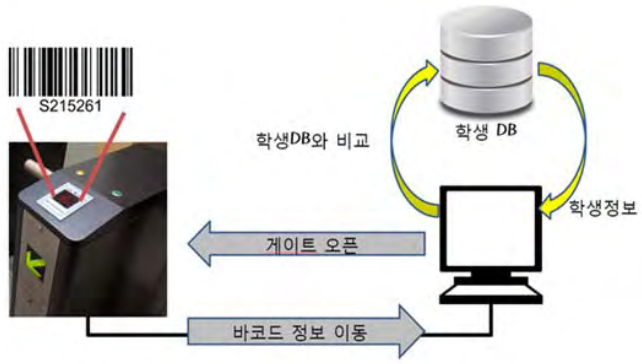
(그림 8) 학교 자습실 입출입 시스템 로그파일

학교 입출입 시스템에 비정상적인 바코드를 태깅하고 DB를 분석해본 결과 (그림 9)와 같이 로그 파일에 빈 값이 기록된다.



(그림 9) 빈 값이 입력된 로그파일

이것으로 보아 비정상적인 데이터가 입력되었을 시 학생 DB와 비교하는 과정에서 일치하는 학생 정보가 없어 빈 값이 반환되었고, 그것이 로그에 저장되었다고 추정한다. 이 빈 값이 시스템이 비정상적인 데이터를 필터링한 결과라고 생각할 수도 있지만 만약 그렇다면 학생DB와 비교하는 과정 전에 걸러져 따로 로그가 저장되거나 적어도 로그파일에 빈 값으로 저장되지는 않을 것이다. 따라서 바코드를 따로 필터링하지 않을 가능성이 크다.



(그림 10) 개략적인 학교 자습실 게이트 동작과정

(그림 10)을 통해 학교 자습실 게이트 동작과정이 웹 어플리케이션을 이용하지 않는다는 것을 제외하고는 웹 스크립트 동작 과정과 별 차이가 없는 것을 확인할 수 있다. 더군다나 바코드 기기에서 SQL Query를 담은 바코드를 따로 필터링하지 않을 가능성이 크므로 SQL Injection 공격 가능성을 제시한다.

3.3 공격 코드 생성

학교의 동의를 얻지 못해 직접 시행은 해보지 못했지만

만약 학교 자습실 게이트의 DB대조 과정이 일반적인 웹 스크립트 동작 과정과 동일하다고 가정했을 때 <표 2>의 SQL Query를 담은 바코드를 태깅하면 'S21521'에 해당하는 학생의 출입 시간이 로그파일의 SAAttend_Log테이블에 저장이 된다. 그리고 뒤의 INSERT INTO SAAttend_Log(OutTime) VALUES('22:00')로 인해 SAAttend_Log테이블의 OutTime 칼럼에 22:00이라는 값이 삽입된다(22시 퇴실 의미). 쉽게 말하자면 자습실 게이트에서 위의 정보를 담은 바코드를 태깅하면 입실과 동시에 그날 22시에 퇴실한 것처럼 기록된다.

<표 2> SQL Injection 공격코드

S21521' INSERT INTO SAAttend_Log(OutTime) VALUES('22:00');--

4. 결론

일반적으로 usb형식의 바코드 스캐너를 사용하는 경우, 컴퓨터에 바코드 스캐너가 키보드로 인식되어 키보드와 같은 기능을 한다. 또한 시리얼 포트를 사용하는 바코드 스캐너는 컴퓨터에 키보드로 인식되지는 않지만 입력 받은 값을 키보드 입력으로 처리한다. 따라서 비정상적인 바코드를 걸러내지 않고 웹 어플리케이션과 같은 동작과정을 거친다면 SQL Injection 공격의 가능성이 존재한다. 또한 code 128과 같이 아스키코드를 기반으로 하는 바코드는 여러 기능을 가진 제어 문자를 표현할 수 있어 Windows 운영체제를 사용할 경우에는 제어 문자를 담은 바코드 태깅을 통해 별다른 입력장치 없이도 컴퓨터를 조작할 수 있다. 따라서 바코드를 이용한 기기에서도 입력값을 검증하는 등의 보안적 노력이 필요하다.

참고문헌

[1] 한국정보통신기술협회 정보통신용어사전, 제어문자, http://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=039718-1

[2] 위키피디아, 미국정보교환표준부호(ASCII Code), https://ko.wikipedia.org/wiki/%EB%AF%B8%EA%B5%AD%EC%A0%95%EB%B3%B4%EA%B5%90%ED%99%98%ED%91%9C%EC%A4%80%EB%B6%80%ED%98%B8#cite_ref-1

[3] 김점구, 노시춘 "공격코드 사례분석을 기반으로 한 SQL Injection에 대한 단계적 대응모델 연구" 정보·보안 논문지 제12권 제1호(2012.03)