Network Security Question 답안

[Lab2]

1. Alice는 aliceDetails.txt를 열고 몇 가지 사항을 변경했습니다. 그녀는 변경한 사항을 저장할 수 있는지에 대해 설명하세요.
   A. 가능하다. 파일의 소유권이 있으며 접근 권한 또한 충분하기 때문이다.

2. Bob은 자신의 파일 bobDetails.txt에 접근하려고 하지만 접근할 수가 없었습니다. 접근할 수 없었던 이유를 설명하고 Bob은 이 파일에 접근하려면 어떻게 해야 하는지 설명하세요.
   A. 해당 파일의 접근 권한이 지정되어 있지 않으므로 접근하기 위해서는 R권한을 부여하면 된다. (chmod u+r bobdetails.txt)

3. Trudy가 ls –l 명령어를 사용하여 Alice의 홈 디렉토리에 있는 aliceFolder의 내용을 표시하려고 하면 파일과 디렉토리를 볼 수 있습니다. Trudy는 cd 명령어를 통해 디렉토리에 접근할 수 있는지에 대한 여부와 그 이유를 설명하세요.
   A. 접근이 불가하다. 해당 디렉토리에 대한 실행 권한이 지정되어 있지 않기 때문이다.

4. UNIX 권한 구조가 열 접근인지 행 접근인지 설명하세요.
   A. 열 접근 방식이다. 사용자 당 파일 권한을 나열한 것이 아닌 각 개별 파일에 대한 사용자 권한을 나열하기 때문이다.

5. /usr/bin/passwd는 시스템에서 비밀번호를 변경하는 데 사용되는 파일입니다. 이 파일은 다음과 같은 권한 설정이 되어있습니다.
   -rwsr-xr-x 1 root root 41292 2009-07-31 09:55 /usr/bin/passwd
   여기서 s는 무엇이며, 그 역할을 설명하세요.
   A. S는 setUID가 root로 설정되었음을 의미한다. 즉, passwd 파일은 파일의 소유자와 동일한 권한으로 실행되므로 다른 사용자가 해당 파일을 실행할 때마다 root의 권한을 가지고 파일을 실행하게 된다.

6. public_html (Alice의 홈 디렉토리 중)에는 index.html이라는 파일이 있으며 다음과 같은 권한이 설정되어 있습니다.
   -rw-r--r-- 2 alice alice 4096 2010-01-17 18:46 index.html
   Trudy는 index.html 파일의 내용을 표시하도록 요청된 웹 서버를 호스팅 할 때 그 내용이 표시되는지에 대한 여부와 그 이유를 설명하세요.
   A. 내용이 표시된다. Trudy는 그 외(other) 사용자로 읽기 권한이 없는 디렉토리의 파일 목록을 읽을 수는 없으나, 읽기 권한이 있는 파일을 사용하기 때문에 디렉토리의 내용을 읽을 수 있다.

7. Documents (Alice의 홈 디렉토리 중) 디렉토리와 관련하여 각 사용자들의 권한을 나열하세요.
   A. Alice는 디렉토리의 소유자이며, 읽기/쓰기/실행 권한을 모두 가지고 있다. Bob은 소유자는 아니지만 동일한 group에 소속되어 있으므로 읽기와 실행 권한을 가지고 있다. Trudy는 소유자도 아니고, 동일한 group에 소속되어 있지 않으므로 아무런 권한이 없다.

[Lab3]

1. [SQL Injection 취약점 찾기]에서 어떤 문자열을 입력하여 해결했는지 기술하세요. 그리고 왜 가능했는지에 대해 이유를 서술하세요.
   A. ID';# 이와 같이 쓸 경우에는 where에서 ID만 확인하고 Password는 확인하지 않고 True값을 반환함. 그 외에도 ' or '1'='1'# 이나 'or 'x'='x'#와 같이 ID를 검증하지 않고 True값을 만들어서 반환하는 것 또한 가능함.

2. [SQL Injection 취약점 찾기(심화)]에서 어떤 문자열을 입력하여 해결했는지 기술하세요. 그리고 간략하게 공격을 진행했던 과정을 서술하세요.
   A. ' or 1=1;# 을 입력할 경우 현재 등록되어 있는 e-mail을 확인할 수 있다. 원래 있던 email주소의 정보를 확인한 후에는 table명과 attribute명을 찾아야 하며, browser의 개발자 도구를 통해 유추할 수 있는 정보 중 emailaddress라는 속성명을 통해 attribute명을 검증하고자 x' AND emailaddress is null;#을 입력하여 attribute 이름이 emailaddress임을 확인한다. 마지막으로 x' AND 1=(SELECT COUNT(*) FROM users);#를 통해 table의 이름까지 확인하였으며, 이 정보들을 토대로 최종적으로 작성하는 Query는 다음과 같다.

   x'; UPDATE users SET emailaddress = '본인email' WHERE emailaddress='기존email';#

3. 1개의 테이블이 가질 수 있는 Primary Key와 Foreign Key는 총 몇 개입니까?
   A. PK는 오직 1개의 테이블에 1개만 있어야 하며, FK는 몇 개를 가져도 상관이 없다.

4. 다음은 버스 운행과 관련된 테이블입니다.

Driver-ID	Bus-ID	Route-ID	Start-Time	Stop-Time
44	67	6	2019/10/01-14:00:00	2019/10/01-22:00:00
44	62	6	2019/10/02-12:00:00	2019/10/02-16:00:00
54	63	9	2019/10/01-16:00:00	2019/10/01-21:00:00
54	63	8	2019/10/02-09:00:00	2019/10/02-17:00:00

Primary Key로 가장 적합한 필드 또는 필드 조합은 무엇인지, 왜 그런지 설명하세요.

A. 현재 주어진 Table에서는 단일 Attribute로 PK를 지정할 수 없는 상황이며, 복합된 속성들이 Unique할 수 있는 조합을 통해 PK를 지정해야 한다. 여기서 가장 이상적인 것은 Driver-ID와 Start-Time의 조합이다. 한 버스기사가 다른 시간에 똑같이 출발할 수는 없기 때문이다.

5. 다음은 어느 산악회의 테이블 구조입니다.

Climber-ID	Name	Skill Level	Age
123	Edmund	Experienced	80
214	Arnold	Beginner	25
313	Bridgett	Experienced	33
212	James	Medium	27

Primary Key가 Climber-ID라고 했을 때, 다음 각 행들을 테이블에 추가할 수 있는지에 대한 여부를 설명하세요.

Climber-ID	Name	Skill Level	Age
214	Abbot	Medium	40
	John	Experienced	19
15	Jeff	Medium	42

A. 첫 번째 레코드는 PK가 중복이므로 추가할 수 없다. 두 번째 레코드는 PK는 NULL값을 취할 수 없기 때문에 이 또한 추가할 수 없다. 마지막 레코드는 PK가 중복되지 않으므로 추가할 수 있다.

[Lab4]

1. 사용자 입력단에서 SQL을 주석처리하는 '--을 제거한다면 모든 SQL Injection을 방지할 수 있습니까? 이에 대한 답과 이유를 서술하세요.
   A. 불가능하다. 주석처리는 -- 뿐만이 아닌 #이나 /*와 같은 형식으로 코드를 주석처리 할 수 있으므로 모든 SQL Injection을 방지하는 것은 가능하지 않다.

2. 단순히 Addslashes() 또는 mysql_real_escape_string()을 사용하여 SQL Injection 방어를 하면 어떤 문제가 있습니까?
   A. 특수문자만 처리하기 때문에 특수문자를 사용하지 않고도 Injection을 하는 경우는 막을 수 없으며, 또한 인코딩 방식의 차이를 이용하여 이를 우회하는 것 또한 가능하다.

3. DB에 대한 보안적 접근을 유지하는데 있어서 UNIX 계열 OS의 접근 제어 메커니즘이 충분하지 않은 이유는 무엇입니까?
   A. 파일은 사용자의 입력을 가져와서 실행하지만, 이를 누가 입력했는지는 따지지 않음. 따라서 DB를 편집하는 권한을 가진 로컬 파일을 통해 DB에 공격을 가해도 OS 단계에서는 이를 처리할 방법이 없음.

4. 기능적인 부분에 있어서 addslashes()와 htmlentities()의 차이는 무엇입니까?
   A. addslashes()는 \ 또는 NULL값과 같은 특정한 escape sequence에 대해서만 취급하지만 htmlentities()는 문자를 HTML entity로 그대로 보여주지만 DBMS에는 영향을 미치지 않도록 문자를 변환한다.

[Lab5]

1. Becoming the Victim's Friend Task에 주어진 Skeleton Code에서 ①과 ②가 사용된 이유를 설명하세요. 왜 이 Code가 있어야 합니까?
   A. 해당 코드는 접속한 사용자의 토큰을 변수로 선언한 부분으로 토큰을 사용하지 않을 경우 사용자 인증이 불가능하다. 따라서 토큰을 사용하기 위해 다음과 같은 코드를 작성하게 된다.

2. About Me Field에서 HTML 편집만 제공할 경우에도 XSS 공격을 계속 성공시킬 수 있습니까?
   A. 악성 스크립트를 검증하는 기능이 존재하지 않는다고 한다면 가능하다.

3. Modifying the Victim's Profile Task에 주어진 Skeleton Code에서 ①은 왜 필요합니까? 해당 Code를 제거한 후 XSS 공격을 반복하여 이에 대한 HTTP Request를 Task 초반에 제시한 Tool을 통해 관찰 결과와 그에 대한 설명을 하세요.
   A. ①의 Code는 현재 접속해 있는 사용자가 Samy인지 아닌지를 검증하는 부분으로 Samy가 아니라면 아래의 Code를 실행하게 되며 Samy라면 수행하지 않는다. 해당 Code를 제거한 후 공격을 수행할 경우 Samy 본인의 프로필 또한 변경될 수 있다.

[Lab6]

1. 변조된 HTTP Request가 제대로 작동하려면 Alice의 사용자 ID(guid)가 필요합니다. Boby는 공격 전에 Alice를 확실한 목표로 정했으며 Alice의 사용자 ID를 얻는 방법을 찾을 수 있습니다. 하지만, Boby는 Alice의 Elgg 비밀번호를 모르기에 Alice의 계정에 로그인하여 정보를 얻을 수는 없습니다. 그럼 사용자 ID를 얻기 위해서 어떻게 해결해야 하는지 설명하세요.
   A. Elgg의 Member 목록에서 브라우저에 내장되어 있는 개발자 도구를 사용하거나 별도의 코드 분석 프로그램을 통해 guid를 확인할 수 있다.

2. Boby는 자신이 생성한 공격용 웹 페이지를 방문하는 사람이라면 누구든지 공격을 당하도록 하고 싶습니다. 하지만 방문하는 사람을 미리 알 수 없습니다. 불특정 다수의 방문자의 Elgg 프로필을 수정하는 공격을 한다고 했을 때, 여전히 CSRF 공격이 가능합니까?
   A. 가능하다. CSRF는 인증된 사용자에 의해 이뤄지는 공격이므로 사이트에 접속이 되어 있는 사용자는 인증이 되어있는 상태이기 때문에 공격자가 의도한 행위를 수행하게 된다.

[Lab7]

1. BOF-(1)에서 hello를 입력하여 Both strings are equal to hello가 올바르게 나왔을 경우 gets()가 call되기 전과 된 이후의 경우를 생각하여 다음 표를 완성하세요.

	str1	str2
Before gets()		
After gets()		

A.

	str1	str2
Before gets()	hello	
After gets()	hello	hello

2. BOF-(1)에서 hello를 입력하지 않고도 Both strings are equal to XXX이 나오는 경우, 왜 정상적으로 동작하는 것처럼 보이는지 설명하세요.
   A. 해당 코드는 접속한 사용자의 토큰을 변수로 선언한 부분으로 토큰을 사용하지 않을 경우 사용자 인증이 불가능하다. 따라서 토큰을 사용하기 위해 다음과 같은 코드를 작성하게 된다.

3. 다음 표는 BOF-(2)에서 변수 check_password가 있는 스택입니다. 각 주소마다 단일 바이트가 들어있고 또한 주소 역시 0x01부터 시작한다고 가정했을 때 Task에서 입력한 값이 어떻게 입력되는지 표를 완성하세요.

Address	Stores a byte for variable	Value	Address	Stores a byte for variable	Value
0x01	check_password.str1	f	0x0a		
0x02			0x0b		
0x03			0x0c		
0x04			0x0d		
0x05			0x0e		
0x06			0x0f		
0x07			0x10		
0x08			0x11		
0x09					

A.

Address	Stores a byte for variable	Value	Address	Stores a byte for variable	Value
0x01	check_password.str1	f	0x0a	check_password.str2	a
0x02	check_password.str1	u	0x0b	check_password.str2	s
0x03	check_password.str1	z	0x0c	check_password.str2	s
0x04	check_password.str1	z	0x0d	check_password.str2	w
0x05	check_password.str1	b	0x0e	check_password.str2	o
0x06	check_password.str1	a	0x0f	check_password.str2	r
0x07	check_password.str1	l	0x10	check_password.str2	d
0x08	check_password.str1	l	0x11	check_password.pass	1
0x09	check_password.str2	p			

[Lab8]

1. Stack Canary 기법은 특정한 값을 삽입하여 canary 값의 변조 여부를 확인하여 BOF가 발생했는지를 확인하는 기법입니다. 하지만 해당 기법에는 취약점이 존재합니다. 이 취약점은 무엇이며, 해당 기법의 파훼법에 대해 간략하게 설명하세요.
   A. Stack canary 기법은 직접적인 BOF에 대해서만 고려한 방어 기법으로 간접적인 접근 또는 우회를 통해 메모리에 접근하여 BOF를 발생시키는 경우에 취약함. 이는 canary의 값은 변조시키지 않으면서 함수의 return address 조작을 통해 BOF 공격을 하는 방식임.

2. ASLR은 Stack이나 Heap, Library를 임의의 메모리 주소로 배치하는 기법입니다. 이를 통해 메모리 상의 공격을 어렵게 하는 효과가 있습니다. 해당 기법을 통해 모든 BOF를 막을 수 있습니까? 이에 대한 답과 그 이유를 서술하세요.
   A. 불가능하다. ASLR 또한 확률에 의존한 기법으로 완벽한 무작위성을 지원하지 않으며 해당 기법을 우회하기 위해 많은 시간이 소요되나 memory leak 또는 offset의 계산을 통해 규칙성을 찾아내는 경우 ASLR을 우회하는 것이 가능하다.

3. C, C++에서 사용되는 함수 중 strcat(), strcpy(), gets(), scanf()와 같이 BOF에 취약한 함수들이 있습니다. 해당 함수들이 가지고 있는 취약점은 무엇이며 이를 해결하기 위한 방법에 대해서 설명하세요.
   A. 해당 함수들은 모두 문자열과 관련된 함수이며 나열된 함수들의 특징은 입력 받은 데이터의 버퍼 길이를 검증하지 않아 BOF를 유발시킬 수 있는 취약점들을 가지고 있다. 현재 다음과 함수들을 사용하는 것은 권장하지 않고 있으며, 해당 함수들을 대신하여 입력 받는 데이터의 버퍼 값 크기를 검사하는 함수인 strncpy(), strncat(), fgets()등의 함수로 대체하여 사용하며 scanf()의 경우 입력 값을 명확하게 제한하여 사용하여야 한다.

[Lab9]

1. 실제로 DoS (DDoS, DRDoS 포함) 공격에는 다양한 유형의 공격들이 존재합니다. 그 중에서도 Flooding 공격은 SYN과 RST 이외에도 다양한 플래그를 통해 공격이 가능합니다. 이와 관련된 Flooding 공격들을 조사하여 간략하게 정리하여 설명하세요. (SYN과 RST를 포함하여 최소 6개 이상)

   A.

   1. SYN Flooding : TCP가 통신을 하기 전 연결을 먼저 맺어야 하는 연결지향성을 이용한 방법으로 3-way handshaking을 이용하여 공격자가 spoofing한 IP 주소를 통해 다량의 SYN 패킷을 대상으로 전달, 대기 큐를 가득 채워 새로운 클라이언트의 연결요청을 무시하도록 하여 장애를 유발하는 공격.

   2. RST Flooding: 서버로 전달되는 클라이언트의 TCP 패킷의 Reset 값을 설정하여 클라이언트가 서버로부터 정상적인 서비스를 받지 못하도록 TCP 연결을 강제로 중단시키는 공격.

   3. ACK Flooding : 공격자가 TCP 세션이 없는 상태에서 TCP 헤더의 Flag를 ACK (0x10)으로 설정하여 무작위로 보내 수신 측에서 변조된 발신 IP로 RST 패킷과 ICMP host Unreachable 패킷을 보내도록 하여 시스템의 과부하를 초래하는 공격.

   4. TCP Connection Flooding : TCP 3-Way Handshake 과정을 과도하게 발생시켜 서비스의 과부하를 유도하는 공격. 공격을 받는 서버는 TCP 세션을 계속 연결하다 처리 자원이 고갈되면 연결을 더 이상 할 수 없어 정상적인 사용자가 더 이상 서비스에 접근할 수 없음.

   5. UDP Flooding : UDP Flooding은 많은 수의 UDP packet을 전송하여 정상적인 서비스가 불가능하도록 하는 공격. 네트워크 대역폭을 고갈시키는 것이 목적이며, 단일 호스트로는 효과를 보기 어려우므로, DDoS의 형태로 공격이 이루어짐.

   6. ICMP Flooding : 대상 시스템에 막대한 양의 ICMP echo request 패킷(ping)을 보내는 공격. 주로 ping을 이용하며, 응답을 기다리지 않고 되도록 빨리 ICMP 패킷을 보내는 옵션을 사용. 대상 시스템에 부하를 일으키기 위해서는 공격 측의 네트워크 대역폭이 대상 시스템의 네트워크 대역폭보다 더 크면 된다.

   7. Http GET Flooding : 동일한 동적 컨텐츠에 대해 HTTP GET Method를 다량으로 발생시켜 웹 서버가 해당 요청을 처리하기 위해 서버의 자원을 과도하게 사용하도록 하여 정상적인 요청을 처리하지 못하도록 만드는 공격.

   8. DNS Query Flooding : UDP 프로토콜 기반의 서비스를 제공하는 DNS에 대해 DNS 쿼리데이터를 다량으로 서버에 전송하여 DNS의 정상적인 서비스를 방해하는 공격.

[Lab10]

1. Sniffing은 Passive Sniffing과 Active Sniffing으로 나뉘어집니다. 이 둘의 차이를 설명하세요.
   A. Passive Sniffing : 단순하게 지나가는 data를 취득하여 확인하는 방법. Hub와 같이 모든 단말 장치에 단순하게 패킷을 전달하는 환경에서 사용.

      Active Sniffing : 앞의 경우와 반대로 능동적으로 data를 취득하기 위해 Switch 환경에서 특정 포트로 지나가는 데이터를 자신의 시스템을 거쳐 지나가도록 설정을 유도하여 패킷을 탈취하는 방법. 주로 MITM(Man-In-The-Middle)의 형태로 sniffing을 시도함.

2. Spoofing 공격 기법으로는 IP Spoofing, ARP Spoofing, E-mail Spoofing, DNS Spoofing이 있습니다. 각 공격 기법에 대해 간략하게 설명하세요.
   A. IP Spoofing : IP Address를 위조하는 기법. 주로 클라이언트가 사용하던 IP를 탈취하여 해당 권한을 획득하는 것을 목적으로 하며, 일반적으로 서버와 클라이언트의 연결을 중단시키고 공격자가 클라이언트로 위장해 서버에 접근하는 형태로 공격이 이뤄짐.

      ARP Spoofing : MAC Address를 위조하는 기법으로 서버와 클라이언트의 MAC Address로 위조하여 패킷이 중간의 공격자에게 오도록 LAN 통신 흐름을 왜곡하여 원하는 정보를 탈취하는 방식.

      E-mail Spoofing : E-mail의 주소를 위조하는 기법으로 주로 바이러스나 Malware를 포함한 메일이나 스팸 메일을 보내는데 사용되며 E-mail의 From 필드에 alias를 사용하여 실제 보낸 사람이 아닌 alias에서 보낸 것으로 인지하도록 하는 방식.

      DNS Spoofing : DNS는 UDP 프로토콜 기반으로 세션이 없으며, 먼저 온 패킷을 우선시하는 취약점을 이용한 것으로 Client가 DNS Server에 Query를 보내면 공격자는 DNS Server보다 먼저 위조된 DNS response를 전송하는 기법. Client는 위조된 DNS Response를 통해 원래 Web Site가 아닌 Phishing Site 또는 Malware나 Ransomware를 유포하는 Site로 접속하게 됨.