

## 6주차 1차시. 레이스 컨디션 공격에 대한 이해

### 【학습목표】

1. 레이스 컨디션 공격 및 파일 링크에 대해 설명할 수 있다.

### 학습내용1 : 레이스 컨디션 공격의 기본 아이디어

#### 1. 레이스 컨디션

- \* 한정된 자원을 동시에 사용하려는 여러 개의 프로세스가 자원 경쟁을 하는 현상
- \* 여러 개의 프로세스가 실행될 때 시분할 또는 멀티태스킹 형태로 실행
- \* 제한된 자원을 짧은 시간 서로 나누어 사용하기 때문에 사용자는 마치 동시에 실행되는 것처럼 느끼게 됨
- \* 레이스 컨디션 예제

```
void main(void) {
    int a, b, childpid;

    // parent process
    if((childpid = fork()) > 0) {
        for(a=0; a<100; a++) printf("P");
        exit(0);
    }
```

```
    // child process
    else {
        for(b=0; b<100; b++) printf("C");
        exit(0);
    }
}
```

```
$ gcc -o multiprocess multiprocess.c
$ ./multiprocess
```

#### 2. 레이스 컨디션 공격

- \* 버그를 갖고 있는 SetUID(4755)가 걸린 프로그램과 해커의 exploit(악의적인 코드)이 서로 경쟁상태(Race Condition)에 이르게 하여, SetUID 프로그램의 권한으로 다른 파일에 접근할 수 있게 하는 공격 방법
- \* 일반적으로 SetUID가 걸려있는 파일을 일반 계정으로 공격하여 관리자 권한을 획득함

### 3. 레이스 컨디션 공격 대상

- \* 관리자 권한의 프로그램에 의해 생성되고 사용되는 임시 파일
- \* 어떤 프로그램은 실행 도중에 임시 파일을 생성하여 사용
- \* 이런 프로그램에서 임시 파일을 생성한 후 그 파일에 접근하는 아주 짧은 시간 동안 끼어들 여유가 생김

### 4. 레이스 컨디션 공격의 기본

- \* 취약 프로그램이 생성하는 임시 파일의 이름 파악
- \* 생성될 임시 파일과 같은 이름의 파일 생성
- \* 생성한 파일의 심볼릭 링크 생성
- \* 생성되었을 때 심볼릭 링크를 이용해 파일 내용을 변경
- \* 파일 실행을 위한 프로세스 진행 중에 관리자 권한을 이용해 필요한 작업을 진행

### 5. 레이스 컨디션 공격 조건

- \* SetUID가 걸려 있을 것
- \* 임시 파일을 생성할 것
- \* 임시 파일 이름을 미리 알고 있을 것
- \* 임시 파일 생성 할 때 레이스 컨디션에 대한 대처를 하지 않을 것

## 학습내용2 : 파일 링크

### 1. 파일 링크 개념

- \* 파일을 잇는 끈
- 하드 링크(Hard Link)  
똑같이 복사된 파일을 생성
- 심볼릭 링크(Symbolic Link)  
원본 파일 데이터를 가리키는 링크 파일을 생성

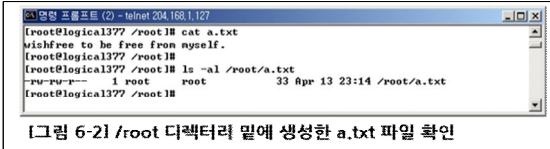
## 2. 하드 링크

\* 똑같이 복사된 파일을 생성

\* 하드 링크 생성 예시

a.txt 파일을 관리자 소유로 /root 디렉터리에 생성

파일 안에 적당한 문구 작성



```

[명령 프롬프트 (2) - telnet 204.168.1.127]
[root@logical377 /root]# cat a.txt
wishfree to be free from myself.
[root@logical377 /root]#
[root@logical377 /root]# ls -al /root/a.txt
-rw-rw-r-- 1 root root 33 Apr 13 23:14 /root/a.txt
[root@logical377 /root]#
    
```

[그림 6-2] /root 디렉터리 밑에 생성한 a.txt 파일 확인

\* 하드 링크 생성 예시

# ln /root/a.txt /wishfree/race/link.txt



```

[명령 프롬프트 (2) - telnet 204.168.1.127]
[root@logical377 race]# ln /root/a.txt /wishfree/race/link.txt
[root@logical377 race]#
[root@logical377 race]# ls -al ./link.txt
-rw-rw-r-- 2 root root 33 Apr 13 23:14 ./link.txt
[root@logical377 race]#
    
```

[그림 6-3] 링크한 파일의 링크 수 확인

하드 링크 생성 예시

링크된 link.txt 파일을 확인해보면/root/a.txt 파일과 내용이 똑같음



```

[명령 프롬프트 (2) - telnet 204.168.1.127]
[root@logical377 race]# cat link.txt
wishfree to be free from myself.
[root@logical377 race]#
    
```

[그림 6-4] a.txt 파일과 링크한 link.txt 파일 내용 확인

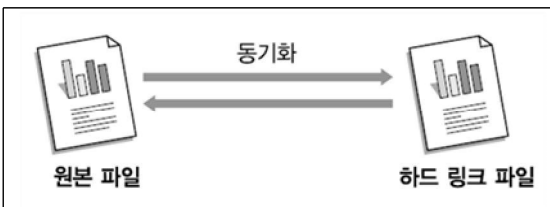
\* 하드 링크된 파일 수정하면원래 파일 /root/a.txt 파일도 똑같이 수정

\* 두 파일 중 하나를 삭제하면 파일의 내용은바뀌지 않고 링크의 숫자만 하나 줄어 듭

\* 하드 링크는 두 파일이 각각 동일한 수준의 데이터를 가지면서 로그 데이터 동기화

\* 하드 링크 제약

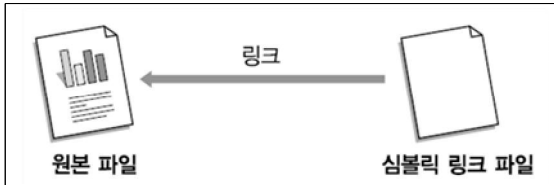
링크하고자 하는 파일이 다른 파티션에 존재하면 안됨



[그림 6-5] 하드 링크의 개념

### 3. 심볼릭 링크

- \* 레이스 컨디션 공격의 대상
  - \* 하드 링크와 달리 실제 두 파일을 생성 링크하지 않음
- 데이터가 있는 파일이 처음부터 한 개만 존재  
원본 파일 데이터 가리키는 링크 정보만을 가짐



[그림 6-5] 심볼릭 링크의 개념

## 학습내용3 : 심볼릭 링크 기능 알아보기

### 1. 주제

심볼릭 링크 기능 알아보기

### 2. 참고

- 한빛미디어
- 정보 보안 개론과 실습: 시스템 해킹과 보안
- 284페이지
- 실습 6-1. 심볼릭 링크 기능 알아보기

### 3. 심볼릭 링크 생성하기

```
# ln -s /root/a.txt /wishfree/race/symlink.txt
```

```

[명령 프롬프트 (2) - telnet 204.188.1.127]
[root@logical377 race]# ln -s /root/a.txt /wishfree/race/symlink.txt
[root@logical377 race]#
[root@logical377 race]# ls -al /wishfree/race/symlink.txt
lrwxrwxrwx 1 root root 11 Apr 13 23:35 /wishfree/race/symlink.t
xt -> /root/a.txt
[root@logical377 race]#
    
```

### 4. 생성된 심볼릭 링크의 파일 권한 확인

```
# ls -al /root/a.txt
```

```

[명령 프롬프트 (2) - telnet 204.188.1.127]
[root@logical377 race]#
[root@logical377 race]# ls -al /root/a.txt
-rw-r--r-- 1 wishfree wishfree 33 Apr 13 23:39 /root/a.txt
[root@logical377 race]#
    
```

## 5. 심볼릭 링크 파일 수정하기

symlink.txt 파일을관리자 권한으로 둔 채 내용을 편집

원본 파일은 내용 확인

# cat a.txt

```

[root@logical377 ~]# cat a.txt
wishfree to be free from myself.
fixed..!!
[root@logical377 ~]#

```

## 6. 원본 파일과 권한의 차이가 있는 심볼릭 링크 파일 수정하기

\* 원본 파일과 심볼릭 링크 파일의 권한 변경

심볼릭 링크 파일을 일반 계정 권한으로 변경

원본 파일을 관리자 계정 권한으로 변경

# chown root.root a.txt

# chownwishfree.wishfree/wishfree/race/symlink.txt

```

[root@logical377 ~]# chown root.root a.txt
[root@logical377 ~]#
[root@logical377 ~]#
[root@logical377 ~]# chown wishfree.wishfree /wishfree/race/symlink.txt
[root@logical377 ~]#

```

## 7. 심볼릭 링크된 symlink.txt 파일 내용 수정

\* symlink.txt 파일의 수정이 불가능 해짐

\* 심볼릭 링크 파일의 소유자가 일반 계정이기 때문에 관리자 root 계정 소유의 원본 파일을 변경할 수 없음

## 8. 원본 파일 삭제

\* 원본 파일을 삭제해도심볼릭 링크 파일은 영향을 받지 않음

# rm a.txt

# ls -al /wishfree/race/symlink.txt

```

[root@logical377 ~]# rm a.txt
rm: remove 'a.txt'? y
[root@logical377 ~]#
[root@logical377 ~]# ls -al /wishfree/race/symlink.txt
lrwxrwxrwx 1 wishfree wishfree 11 Apr 13 23:38 /wishfree/race/symlink.txt -> /root/a.txt
[root@logical377 ~]#

```

## 9. 심볼릭 링크 파일 내용 확인

심볼릭 링크가 가리키는 원본 파일이존재하지 않으므로 오류가 발생

# cat /wishfree/race/symlink.txt

```

[root@logical377 ~]# cat /wishfree/race/symlink.txt
cat: /wishfree/race/symlink.txt: No such file or directory
[root@logical377 ~]#

```

## 10. 동일 권한의 원본 파일 재생성하기

```
$ chmod 777 /root
$ su wishfree
$ touch a.txt
$ ls -alh /root/a.txt
```

```
명령 프롬프트 (2) - telnet 204.168.1.127
[wishfree@logical377 /root]$ touch a.txt
[wishfree@logical377 /root]$
[wishfree@logical377 /root]$ ls -alh /root/a.txt
-rw-rw-r-- 1 wishfree wishfree 17 Apr 13 23:58 /root/a.txt
[wishfree@logical377 /root]$
```

## 11. 원본 파일과 심볼릭 링크 파일 내용 확인

```
$ cat a.txt
$ cat /wishfree/race/symlink.txt
```

```
명령 프롬프트 (2) - telnet 204.168.1.127
[wishfree@logical377 /root]$ cat a.txt
Renaked a.txt
[wishfree@logical377 /root]$ cat /wishfree/race/symlink.txt
Renaked a.txt
[wishfree@logical377 /root]$
```

## 12. 심볼릭 링크 파일 수정 및 내용 확인

```
$ (printf "Fixed...%n") >> ./symlink.txt
$ cat /wishfree/race/symlink.txt
```

```
명령 프롬프트 (2) - telnet 204.168.1.127
[wishfree@logical377 race]$ (printf "Fixed.....%n") >> ./symlink.txt
[wishfree@logical377 race]$
[wishfree@logical377 race]$ cat ./symlink.txt
Renaked a.txt
Fixed.....
[wishfree@logical377 race]$
```

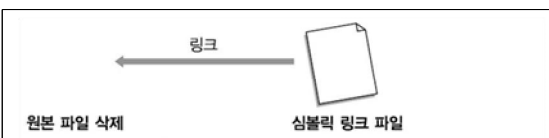
## 13. 원본 파일 내용 확인

```
$ cat /root/a.txt
```

```
명령 프롬프트 (2) - telnet 204.168.1.127
[wishfree@logical377 race]$ cat /root/a.txt
Renaked a.txt
Fixed.....
[wishfree@logical377 race]$
```

## 14. 심볼릭 링크 특징

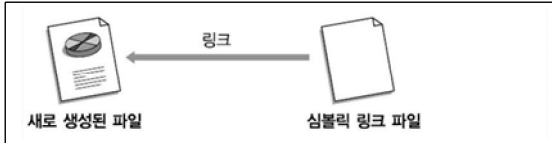
\* 원본 파일이 삭제 되었어도 심볼릭 링크는 원본 파일의 이름과 위치를 계속 기억하고 있음



[그림 6-17] 삭제된 파일에 대해

여전히 링크 정보를 가진 심볼릭 링크

\* 지운 원본 파일과 동일한 이름으로 파일을 생성하면 새로운 파일에 대한 심볼릭 링크 파일로 존재하게 됨



[그림 6-18] 새로 생성된 파일에 대해링크 정보를 가진 심볼릭 링크

## 【학습정리】

1. 레이스 컨디션이란 한정된 자원을 동시에 사용하려는 여러개의 프로세스가 자원을 위해 경쟁하는 현상을 이용한 해킹 기법이다. 하드 링크는 두 파일이 각각 동일한 수준의 데이터를 가지며, 서로 동기화 한다.