위·변조 감시 시스템

팀　　명 : 감프(감시프로그램)

지도교수 : 양환석 교수님

팀　　장 : 서규현

팀　　원 : 장정윤

조은서

설희운

류현호

2018. 11.

중부대학교 정보보호학과

목 차

1. 서론

우리나라는 지금까지 IT분야에서 선두적인 발전과 높은 성장률을 보여주고 있다. 하지만 IT분야에서 많은 발전이 이루어졌지만 그에 반해 보안 시스템에 도입률과 발전에서는 더딘 현상을 보이고 있다. 현재 대부분의 기업에서는 많은 돈을 들여 정보 보안에 큰 힘을 쓰고 있지만 학교나 중소기업과 같은 소규모 네트워크 환경을 구축하고 있는 곳에서는 정보 보안의 노력이 부족하고 보안의식이 떨어지기 때문에 다양한 유출사고나 취약한 점이 있다. 이러한 점을 보완하고자 서버 운영과정에서 발생할 바이러스 침입이나 악성코드 등에 의한 파일 위변조 행위가 일어날 경우 서버 운영에 심각한 마비를 방지하기 위해 파일 위변조 감시 시스템을 프로젝트 주제로 선정하게 되었다.

2. 관련연구

2.1 쉘 프로그래밍

쉘 프로그램(쉘 스크립트)이란 unix에서 사용하는 모든 명령어들이 각각의 기능을 발휘할 수 있도록 입력 받을 준비가 된 상태를 쉘 상태라 하고 그 상태에서 사용하는 unix명령어를 쉘 명령이라하며 하나 이상의 쉘 명령어들이 하나의 파일에 집합되어서 수록되어 있는 순서대로 처리하여 목적한 바를 수행하게끔 하는 명령어들의 집합 그 자체를 쉘 프로그램(쉘 스크립트)이라 한다.

2.2 MySQL

MySQL은 표준 데이터베이스 질의 언어인 구조화 질의 언어(SQL: Structured Query Language)를 사용하는 공개 소스의 관계형 데이터베이스 관리 시스템(RDBMS). 매우 빠르고, 유연하며, 사용하기 쉬운 특징이 있다. 다중 사용자, 다중 스레드(thread)를 지원하고, C, C++, 에펠(Eiffel), 자바, 펄, PHP, 파이선(Python) 스크립트 등을 위한 응용 프로그램 인터페이스(API)를 제공한다. 유닉스나 리눅스, 윈도우 운영 체제 등에서 사용할 수 있다. 램프(LAMP), 즉 리눅스 운영 체계와 아파치(Apache) 서버 프로그램, MySQL, PHP 스크립트 언어 구성은 상호 연동이 잘되면서도 공개 소스(오픈 소스)로 개발되는 무료 프로그램이어서 홈 페이지나 쇼핑몰 등 일반적인 웹 개발에 널리 이용되고 있다. mySQL은 공개 소스 라이선스를 따르기는 하나 상업적으로 사용할 때에는 상업용 라이선스를 구입해야 한다. 공식 사이트는 www.mysql.com 이다.

2.3 인증서

1) 인증서 개념

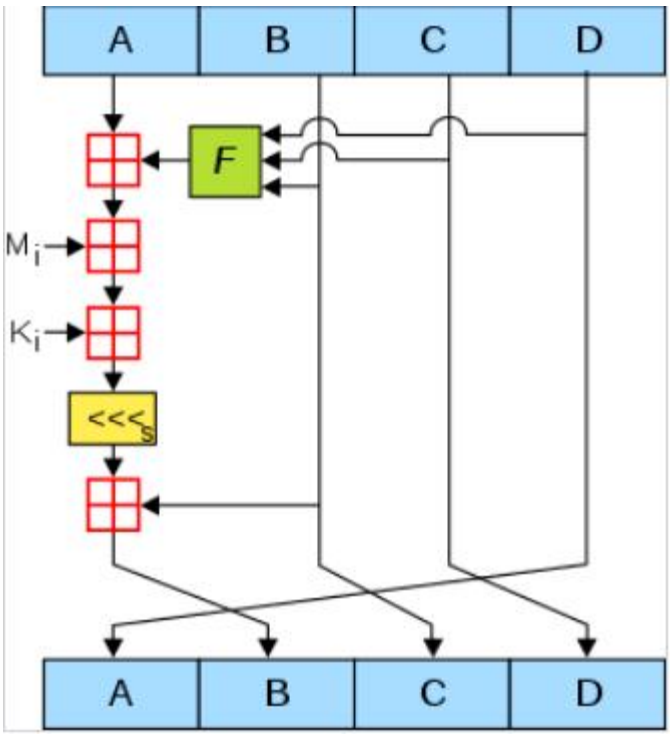
인증서는 (1) 인증(접속 당사자의 신원 확인)수단으로 사용될 수도 있고, (2) 교신 암호화에 사용될 수도 있고, (3) (클라이언트 인증서의 경우) 거래 내역을 전자서명하는 용도로도 사용될 수 있다. 그러나 거래내역 전자서명은 교신 체널의 보안이나 교신 당사자 인증과는 구분되는 개념이다.

2)MD5

MD5(Message-Digest algorithm 5)는 128비트 암호화 해시 함수이다. RFC 1321로 지정

되어 있으며, 주로 프로그램이나 파일이 원본 그대로인지를 확인하는 무결성 검사 등에 사용된다. 1991년에 로널드 라이베스트가 예전에 쓰이던 MD4를 대체하기 위해 고안했다. 메시지의 무결성(Message Integrity) 점검을 위해 이용하는 하나의 알고리즘이다.

간이 전자 우편 전송 프로토콜(SMTP: Simple Mail Transfer Protocol) 서버 소프트웨어인 센드메일(sendmail)이나 도메인 네임 서버(DNS)의 바인드(BIND) 소프트웨어 인증, 그리고 데이터베이스 등에서 널리 사용된다. 초기의 MD2가 8 비트 버전인데 비해 MD4나 MD4의 개선형인 MD5는 32 비트 컴퓨터에 최적화되어 있다.



<그림 2.5-2 MD5 알고리즘 동작 원리>

2.4 위변조

위조란 권한 없는 자가 사용할 목적으로 현존하지 아니하는 문서, 통화, 유가증권, 인장 등을 새로이 작성하거나 제조하는 것을 말하며 침입이나 공격을 목적으로 하여 데이터를 위조하는 행위. 호스트 이름과 IP 주소의 조합을 DNS 서버에 있는 것과 다른 정보로 바꿔쓰는 DNS 위조와 패킷의 경로 정보를 변경하는 RIP 위조 등 위조하는 정보에 따라 다양하다.

변조란 컴퓨터 내의 정보를 부정한 수단으로 개서(改書)하는 것. 이러한 범죄의 예로는 사원이 인사 시스템에 부정한 방법으로 접속하여 급여 금액을 마음대로 상향 조정하거나, 학생이 자신의 성적 기록을 변조하는 행위 등이 있다. 또한 통신망의 중계 서버에 침입하여, 타인의 이메일이나 데이터를 통신 도중에 갈취해서 변조하는 예도 있다.

2.5 C#

C#이란 C++의 컴퓨팅 파워와 비주얼 베이직의 프로그래밍 편이성을 결합하기 위 한 목적

으로 마이크로소프트사에서 개발한 새로운 객체 지향 프로그래밍 언 어. C++에 기반을 두고 Java와 비슷한 특색을 가지고 있으며, 마이크로소프트의 .NET 플랫폼에서 쓰인다. 프로그래머가 각 단계별로 별도의 코드를 작성하지 않고서도 프로그래밍 객체에 접근할 수 있는 확장성 마크업 언어(XML)와 단순 객체 접근 프로토콜(SOAP)의 사용을 통해 프로그래밍을 단순화한 것이다.

2.6 파이썬

Python이란 다양한 플랫폼에서 사용이 가능한 객체 지향 기반 인터프리터 방식의 고급 프로그래밍 언어 중 하나이다. 문법이 배우기 쉽고 결과를 바로 확인할 수 있어서 프로그래밍 초보자에게 추천되는 언어이다.

1991년에 귀도 반 로섬(Guido van Rossum)이 발표하였고, 좋아하는 코미디 프로인 에서 파이선이라는 이름을 가져온 것으로 알려져 있다. 공동체 기반의 개방적 개발 모델을 가지고 있으며, 파이선 소프트웨어 재단(Python Software Foundation)에서 관리한다.

문법 구조가 간단, 명확하고, 많은 시스템 호출과 풍부한 라이브러리(모듈), 다양한 윈도우 시스템용 인터페이스 등을 포함하고 있어 생산성이 높은 강력한 언어이기도 하다. 반면 실행 속도가 상대적으로 느리다는 단점이 있다. 다른 프로그래밍 언어와의 호환성이 높아, 빠른 속도가 필요한 부분은 C, C++ 등의 언어로 개발하여 융합할 수 있다.

2.7 텔레그램

Telegram은 보안 및 속도에 중점을 둔 클라우드 기반 모바일 및 데스크톱 메시징 응용 프로그램이다. 완벽한 동기화 기능을 갖춘 클라우드 기반 메신저이다. 결과적으로, 태블릿, 컴퓨터를 포함하여, 한 번에 여러 장치에서 메시지에 액세스, 최대 1.5 GB의 사진, 비디오 및 파일 (문서, 우편, MP3 등)의 무제한 공유 할 수 있습니다 각각 . 또한 모든 데이터를 장치에 저장하지 않으려면 언제나 클라우드에 보관할 수 있다.

다중 데이터 센터 인프라와 암호화 덕분에 Telegram은보다 빠르고 안전하다. 뿐만 아니라, 텔레 그램은 무료이며 영원히 광고도 구독료도 무료이다. 텔레그램은 API가 공개되어 있다.

텔레그램의 큰 강점인 보안도 다른 메시지에 비해 빠지지 않는다. 전자프론티어재단(EFF)은 지난 2014년 11월 처음으로 '보안메시지 서비스 평가표'를 발표하면서 텔레그램의 보안 점수를 7점 만점에 4점으로 매겼다. 텔레그램의 일반 채팅 기능은 보안 점수 4점에 그쳤지만, 텔레그램의 비밀대화 기능은 보안 평가에서 만점을 받았다. 페이스북 채팅과 구글 행아웃이 2점을 받은 것과 비교하면 놀라운 보안 수준이다.

일반 메시지 기능은 상대방이 바로 확인하지 않아도 메시지를 보낼 수 있지만, 비밀대화 기능은 상대방이 비밀대화 상태임을 알아야만 채팅을 주고받을 수 있다. 서버에 대화 내용을 저장하지 않고, 사용자의 스마트폰을 P2P로 연결해 보안성을 높였다. 전송한 메시지를 자동으로 삭제할 수 있도록 한 것도 특징이다. 자동 삭제 시간은 짧게는 1초에서 길게는 1주일까지 설정할 수 있다.

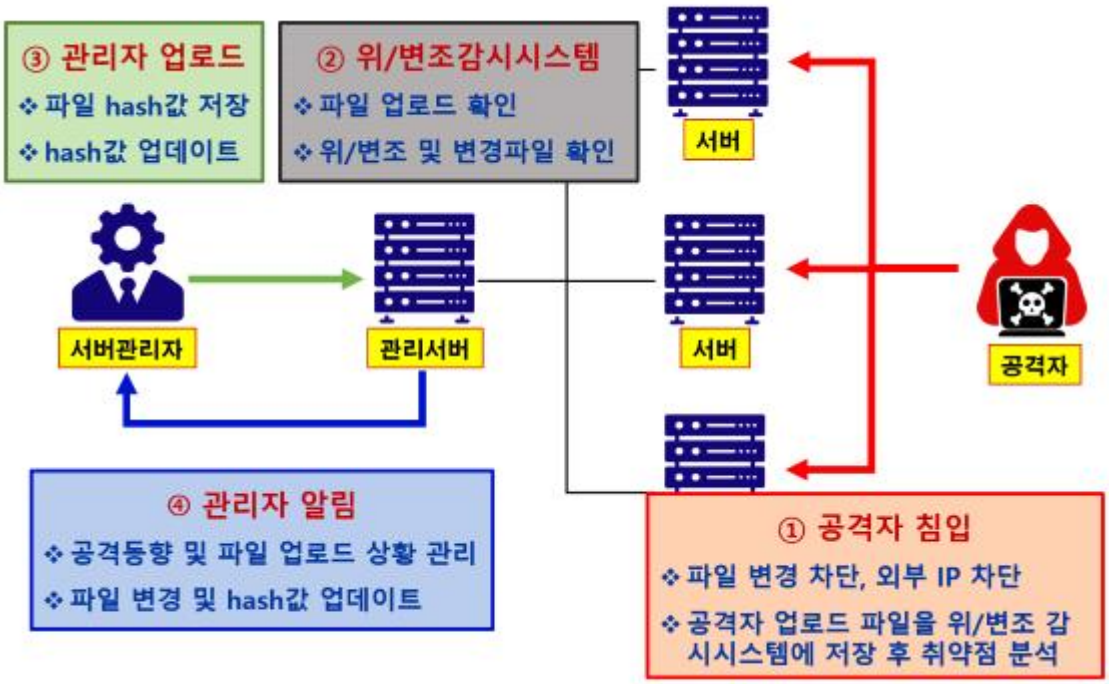
여기에 텔레그램은 메시지가 다른 사람에게 보이지 않도록 2중 암호화를 실행한다. 서버 클라이언트 간 보안, 클라이언트와 클라이언트 간 이중으로 보안 과정을 거친다. 텔레그램 메시지는 256비트 AES(Advanced Encryption Standard), RSA 2048, 디피-헬맨(Diffie-Hellman)

키 교환 방식으로 암호화 키를 교환한다. 텔레그램은 이 방식을 이용해 비밀대화 기능을 만들었다.

디피-헬맨 방식을 이용하면 암호화 키가 일종의 시각화 방식으로 변환돼 이미지로 암호 코드가 교환된다. 메시지를 보내는 사람과 받는 사람 간 기기에서 암호화 코드 이미지가 동일해야 메시지 전송이 이뤄지는 만큼, 중간자 공격을 피할 수 있다.

3. 본론

3.1 시스템 구상도 설명



<그림 3.1 위변조 감시 시스템 구상도>

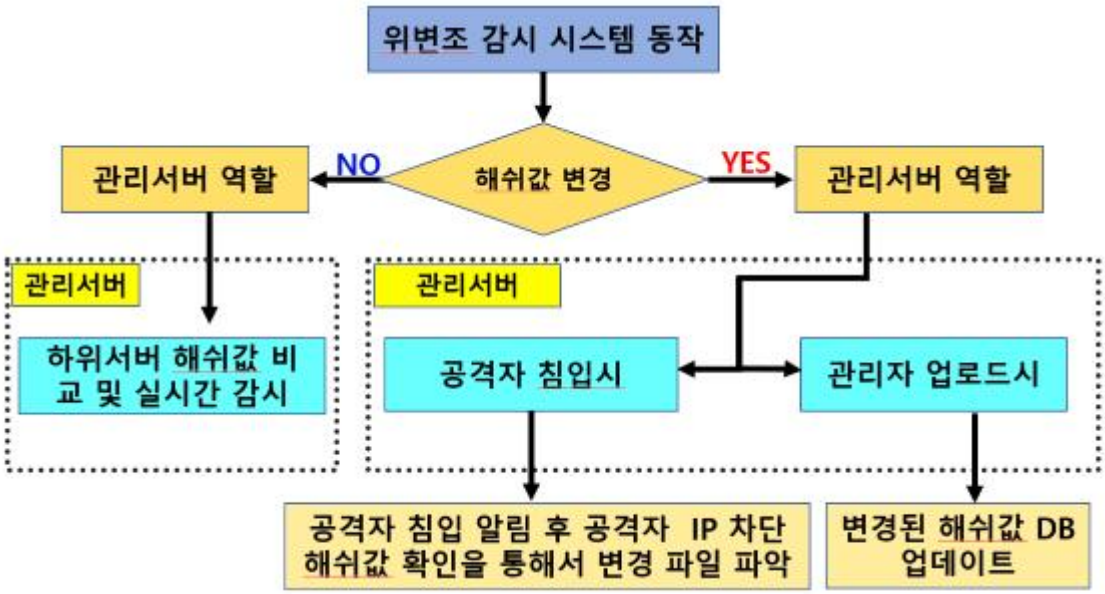
서버 관리자는 관리서버를 통하여 서버들의 중요파일의 해쉬값을 관리한다. 관리서버 DB에 저장되어있는 해쉬값과 서버의 해쉬값을 비교하여 위변조 여부를 파악한다.

서버 관리자가 서버에 파일 업로드시 변경된 파일의 해쉬값을 관리서버에서 관리하고있던 해쉬값 DB를 최신화 한다.

공격자 침입시 파일 변경을 차단하고 외부 IP를 차단후 관리자에게 알림을 보낸다. 또한 공격자가 업로드 하려던 파일을 저장하여 취약점 분석 용도로 사용한다.

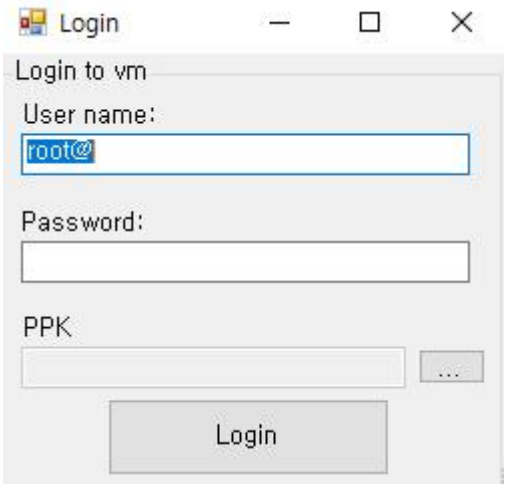
서버관리자는 프로그램을 이용하여 좀더 편리하게 관리서버를 관리할수 있고 실시간 감시를 통해서 텔레그램을 이용하여 원격 명령과 실시간 정보를 받아볼 수 있다.

3.2 시스템 작업 계통도



<그림 3.2 위변조 감시 시스템 작업 계통도>

위변조 감시 시스템의 작동 계통도로 위변조 감시 시스템은 크기 해쉬값 변경의 기점으로 나눌수 있는데 해쉬값이 실시간으로 비교 하고있는중 관리자가 업로드시 변경된 파일의 해쉬값을 DB에 업데이트해서 서버의 해쉬값을 최신화한다. 공격자가 침입시 관리서버는 외부 IP가 접속했다고 알림후 IP를 차단하고 해쉬값 확인을 통해서 변경 파일을 파악 한다.



3.3 로그인

<그림 3.3 위변조 감시 시스템 로그인창>

로그인으로 관리서버에 접속이 가능하고 PPK 인증서가 있는 관리서버에서도 사용이 가능하다.

3.4 위변조 감시 시스템 메뉴 설명

<그림 3.4 위변조 감시 시스템 메뉴>

LOGIN & 관리서버에 로그인한다.

AUTO START & 위변조 실시간 감시를 시작한다.(실시간)

MANUAL START & 위변조 감시를 시작한다. (1회 수행)

STATUS & 위변조 감시의 작동 여부를 확인한다.

STOP & 위변조 감시 정지한다.

HASH & 해쉬값을 출력한다.

LOG & 변경된 해쉬값을 확인한다.

ADD, DEL & 블랙리스트 IP를 등록 및 삭제한다.

3.5 위변조 감시 시스템 동작

<그림 3.5 위변조 감시 시스템 동작>

AUTO START, MANUAL START시 해쉬값 변경되었다고 알림 및 IP정보 출력된다.

3.6 위변조 감시 시스템 해쉬값 확인

<그림 3.6 위변조 감시 시스템 해쉬값 출력>

해쉬값을 확인하여 서버의 상태를 확인할 수 있다.

3.7 위변조 감시 시스템 로그 확인

<그림 3.7 위변조 감시 시스템 로그 출력>

서버의 변경된 해쉬값을 정보를 확인하여 관리자가 상황에 대비 할 수 있게 한다.

3.8 위변조 감시 시스템 블랙리스트 추가/삭제

<그림 3.8-1 위변조 감시 시스템 블랙리스트 추가>

서버에 관리자가 블랙리스트를 등록하여 서버에 Iptables에 등록시킨다. 관리자가 불필요한 블랙리스트를 삭제하면 서버에 Iptables에서 삭제가 된다.

```
root@seo-virtual-machine:~/.seven# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.5          anywhere
DROP       all  --  192.168.0.6          anywhere
DROP       all  --  192.168.0.7          anywhere
DROP       all  --  192.168.0.8          anywhere
DROP       all  --  192.168.0.9          anywhere
DROP       all  --  192.168.0.10         anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

<그림 3.8-2 블랙리스트에 등록한 IP>

관리자가 블랙리스트에 등록한 IP가 서버의 Iptables에 등록된 화면이다.

3.9 텔레그램 알림

<그림 3.9-1 텔레그램을 통해 위변조 알림>

<그림 3.9-2 텔레그램을 통해 관리자 및 외부IP 접속 알림>

<그림 3.9-3 텔레그램을 통해 해쉬값이 변경된 파일 확인>

텔레그램을 이용하여 실시간 알림을 받아볼수 있다.

3.10 텔레그램 원격 명령 실행

<그림 3.10 텔레그램을 통해 원격으로 명령 실행>

텔레그램을 통해서 원격으로 관리서버에 원격 명령을 실행시킬 수 있다.

4. 결론

CentOS, ubuntu 운영체제를 기반으로 shell, python, c# 언어로 위변조 감시 시스템을 개발을 하였다. 관리서버와 서버를 구축하였고 여기에 다양한 작업을 구현하였다.

프로젝트에서 중요한 부분은 총 3가지로 정리할 수 있다.

먼저 위변조 감시 부분이다. 공격자 방어 쉘을 실행시켜 실시간 감시를 하고, 감시 중에 공격자에 의해서 파일이 변경되면 미리 복제해둔 해쉬값 파일과 해쉬값을 비교 후 위변조를 화인하여 실시간으로 알림을 보낸다. 또한 프로세스 실행여부를 알 수 있고, 위변조 감시 시스템을 정지시킬 수 있고 의심되는 아이피는 차단한다.

두 번째는 관리자 알림이다. 이 부분은 실시간으로 정보를 받거나, 원격 명령을 실행하기 위해 만들어야 했다. 다양한 명령어를 쉘로 만들어 원격으로 실행이 가능하게 하였다. Telegram API를 사용하였다. 원래 대한민국에서 가장 친숙한 메신저인 '카카오톡'을 사용하여 관리자

알림을 시도하였지만 한계가 있었다. 결국 오픈소스를 제공하는 Telegram을 사용하게 되었는데 오히려 좋은 경험을 하였다. 오픈소스를 제공하는 만큼 우리가 원하는 방향으로 좀 더 편리하게 제작할 수 있었서 더 좋았다.

세 번째는 GUI다. 프로그램을 GUI로 제작하여 사용할 수 있게 하였다. 누구나 사용하기 쉽게 버튼을 구성하였고 PPK 인증서를 기반으로 로그인을 가능하게 제작하였다. 블랙리스트를 추가하거나 삭제하는 기능도 제작하였다. 서버를 주로 다루는 우리에게 GUI 제작은 신선한 경험이었고 누구나 편리하고 쉽게 서버를 관리할 수 있게 GUI 만들어서 빠른 대처가 가능할 것이라고 본다.

위변조 감시 시스템을 구현함으로써 실시간 알림을 통해 즉각적인 대처를 할 수 있고, 누구나 편리하고 안전하게 서버를 관리할 수 있음을 기대해본다.

5. 참고자료

5.1 참고 문헌

김태용의 리눅스 쉘 스크립트 프로그래밍 입문 & 김태용 저자

UNIX/Linux 시스템 관리자를 위한 쉘 스크립트 활용 가이드 & 정해주 저자

5.2 참고 사이트

GitHub & python-telegram-bot
https://github.com/python-telegram-bot/python-telegram-bot

한국정보통신기술협회(TTA) http://www.tta.or.kr/

위키백과, 우리 모두의 백과사전 https://ko.wikipedia.org/wiki/

Stack Overflow & Where Developers Learn, Share, &Build Careers
https://stackoverflow.com/

고급 Bash 스크립팅 가이드
https://wiki.kldp.org/HOWTO/html/Adv-Bash-Scr-HOWTO/index.html

Python으로 telegram bot 활용하기
https://blog.psangwoo.com/coding/2016/12/08/python-telegram-bot-1.html

6. 첨부자료

6.1 발표자료

위·변조 감시 시스템

2018. 11. 7

중부대학교 정보보호학과
지도교수 : 양환석 교수님
감프(감시프로그램)
(서규현, 조은서, 설희운, 장정윤, 류현호)

1

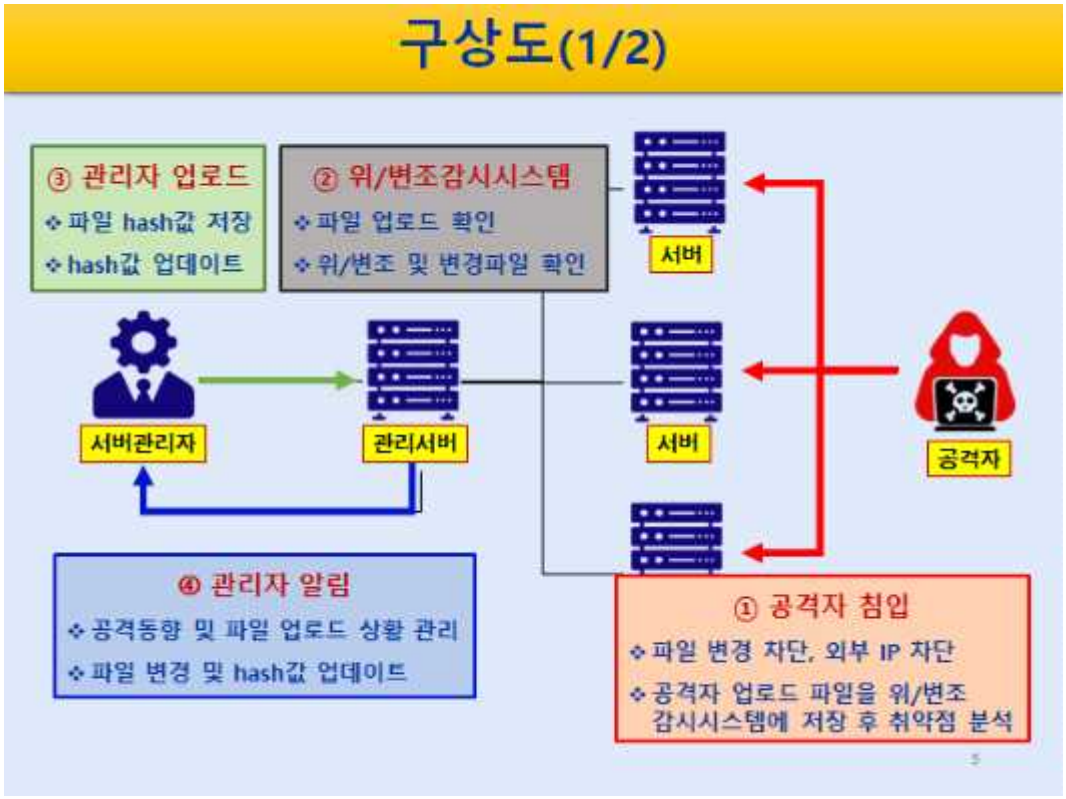
목 차

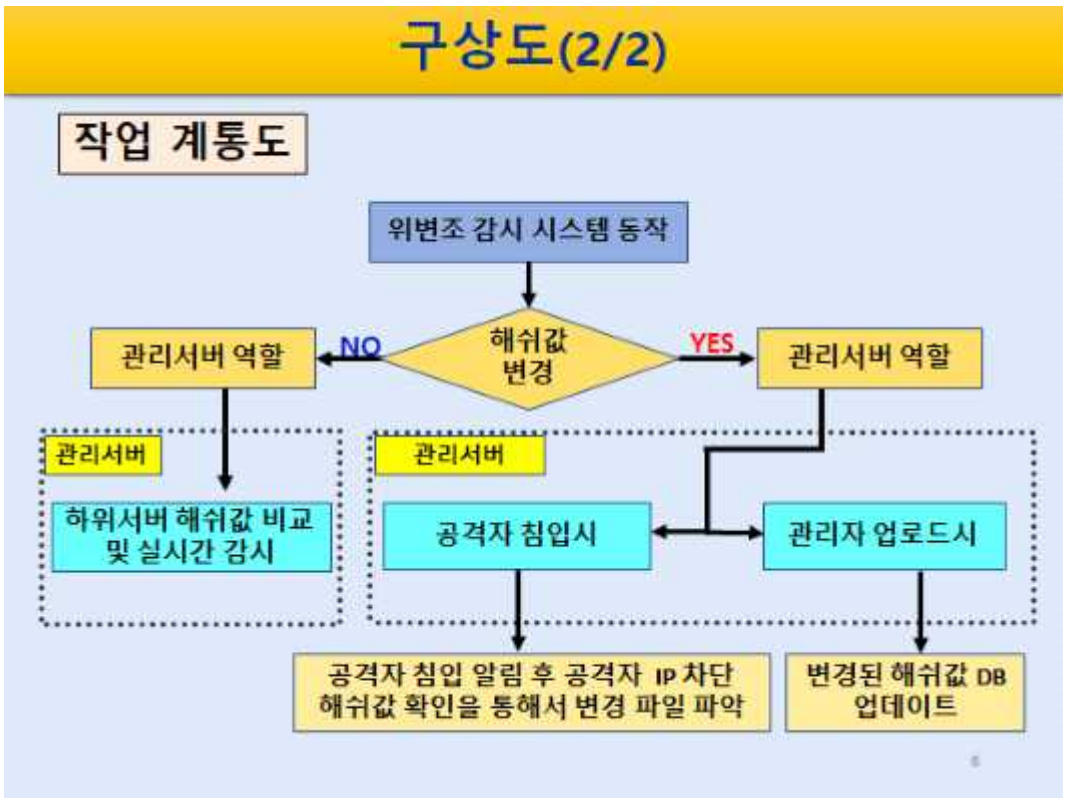
2

조원 편성

이 름	담당 임무
서규현	프로젝트 총괄, 서버 연동, hash값 업데이트, DB구축
류현호	서버 구축, 파일 차단/변경, 관리자 알림
조은서	위•변조 파일 저장, 관리자 알림
장정윤	GUI(HASH/LOG연동, 블랙리스트 관리, 쉘 검사 선택) 파일 차단/변경, 외부 공격 IP차단 외부 업로드파일 위•변조 감시 기능 구현
설희운	파일 차단/변경, DB구축

주제 선정


구상도(1/2)
③ 관리자 업로드
❖ 파일 hash값 저장
❖ hash값 업데이트
② 위/변조감시시스템
❖ 파일 업로드 확인
❖ 위/변조 및 변경파일 확인
서버
서버관리자
관리서버
서버
공격자
④ 관리자 알림
❖ 공격동향 및 파일 업로드 상황 관리
❖ 파일 변경 및 hash값 업데이트
① 공격자 침입
❖ 파일 변경 차단, 외부 IP 차단
❖ 공격자 업로드 파일을 위/변조 감시시스템에 저장 후 취약점 분석


구상도(2/2)
작업 계통도
위변조 감시 시스템 동작
해쉬값 변경
NO
YES
관리서버 역할
관리서버 역할
관리서버
관리서버
하위서버 해쉬값 비교 및 실시간 감시
공격자 침입시
관리자 업로드시
공격자 침입 알림 후 공격자 IP 차단 해쉬값 확인을 통해서 변경 파일 파악
변경된 해쉬값 DB 업데이트

추진 경과

구분 \ 기간(월)	3	4	5	6	7	8	9	10
구상도 설계								
자료 조사								
서버 구축								
프로그램 개발								
연동체계 구축								
프로그램 수정/보완								
연동 확인								
시스템 종합								
PPT, 보고서 작성								

개발 환경 및 시스템 개발(1/11)

개발 환경

운영 체제

CentOS ubuntu

개발 언어

python C#

개발 환경 및 시스템 개발(2/11)

관리서버 구축

❖ Hash값 저장을 위한 관리서버 DB 구축

DB 구축

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18
Server version: 5.7.24-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| subhash            |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

Table 생성

```
mysql> use subhash;
Reading table information for completion of table and
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-------------------+
| Tables_in_subhash |
+-------------------+
| sub_server1       |
+-------------------+
1 row in set (0.00 sec)
```

개발 환경 및 시스템 개발(3/11)

관리서버와 하위서버 연동(1/2)

❖ 관리서버에서 하위서버로 save명령어 실행 요청

```
#!/bin/bash

ssh root@192.168.40.134 ./save.sh
```

❖ 관리서버에서 하위서버로 hash값 파일 요청

```
#!/bin/bash

ssh root@192.168.40.134 ./md5.sh

scp root@192.168.40.134:/root/md5test.txt /var/lib/mysql-files/log/

read pw

mysql -uroot -p$pw -D MD5 -e"LOAD DATA INFILE '/var/lib/mysql-files/log/md5test.
txt' INTO TABLE submd5 FIELDS TERMINATED BY ' ' LINES TERMINATED BY '\n';"
```

❖ 관리서버에서 하위서버로 위변조 감시 명령어 실행

```
#!/bin/bash

ssh root@192.168.40.134 "./intDtc.sh start &"
```

- 관리서버에서 하위서버로 명령어 실행 시 SSH를 이용
- 파일 요청 및 수신 시 SCP 사용
- 하위서버 hash값은 관리서버에 저장

개발 환경 및 시스템 개발(4/11)

관리서버와 하위서버 연동(2/2)

❖hash값 업데이트 및 관리서버로 전송

```
#!/bin/bash

ori="./md5test.txt"
copy="./md5copy.txt"

cp "$ori" "$copy"

NOW=$(date +"%Y%m%d%H%M%S")

md5log=`find /etc/ -type f -printf '"%p"\n'|xargs md5sum -b > ./md5compare$NOW.txt`;

md5compare=`find /etc/ -type f -printf '"%p"\n'|xargs md5sum -b > ./md5compare.txt`;
compare="./md5compare.txt"

log="./md5compare$NOW.txt"

if [ -z "`diff $copy $compare`" ]
then
        echo "변경된 값이 없습니다."
else
        echo "변경된 값이 있습니다."
        #diff $copy $compare
        cp "$compare" "$ori"
        scp $log root@192.168.40.141:/roo
```

❖서버 관리자가 하위서버 파일을 변경할 경우 변경된 파일의 hash값을 업데이트 후 관리서버로 전송

11

개발 환경 및 시스템 개발(5/11)

GUI 처리부 구현

메인 화면 버튼 구성 등 각종 기능 버튼 디자인 및 설계

12

개발 환경 및 시스템 개발(6/11)

로그인 기능처리

```
private void LoginBtn_Click(object sender, EventArgs e
{
    var loginForm = new
    loginForm.ShowDia

    var splitString =
    if (splitString.Le
    {
        AppendToOutput
    }

    mUserName = splitS
    mHost = splitStrin
    mPassword = loginF
```

로그인 유효성 확인

```
    if (string.IsNullOrEmpty(loginForm.PpkFilePath) == tru
        mSshClient = new SshClient(mHost, mUserName, mPassword);
        mScpClient = new ScpClient(mHost, mUserName, mPassword); }
    else {
        mPPK = new PrivateKeyFile(loginForm.PpkFilePath);
        mSshClient = new SshClient(mHost, mUserName, mPPK);
        mScpClient = new ScpClient(mHost, mUserName, mPPK); }

    mSshClient.ConnectionInfo.Timeout = new TimeSpan(0, 0, 5);
    mScpClient.ConnectionInfo.Timeout = new TimeSpan(0, 0, 5);

    AppendToOutputTextBox("로그인 중...");
    mSshClient.Connect();
    mScpClient.Connect();
    AppendToOutputTextBox("로그인 성공");

    var ret = mSshClient.RunCommand(string.Format("[ ! -d {0} ] && echo not_found",
    if (ret.Result == "not_found\n") {
        RunCommand($"mkdir {GetScriptsPath()}"); }
    else {
```

정상 로그인 확인

로그인 정보 확인/인증 등 로그인 처리 기능 설계

13

개발 환경 및 시스템 개발(7/11)

HASH 값 관리

❖ 관리서버에서 하위서버로 hash값 파일 요청 및 DB에 저장

```
#!/bin/bash

ssh root@192.168.40.134 ./md5.sh

scp root@192.168.40.134:/root/md5test.txt /var/lib/mysql-files/log/

read pw

mysql -uroot -p$pw -D MD5 -e"LOAD DATA INFILE '/var/lib/mysql-files/log/md5test.txt'
 INTO TABLE submd5 FIELDS TERMINATED BY ' ' LINES TERMINATED BY '\n';"
```

```
mysql> describe sub_server1
    -> ;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| Hash_value | text | YES  |     | NULL    |       |
| Path       | text | YES  |     | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

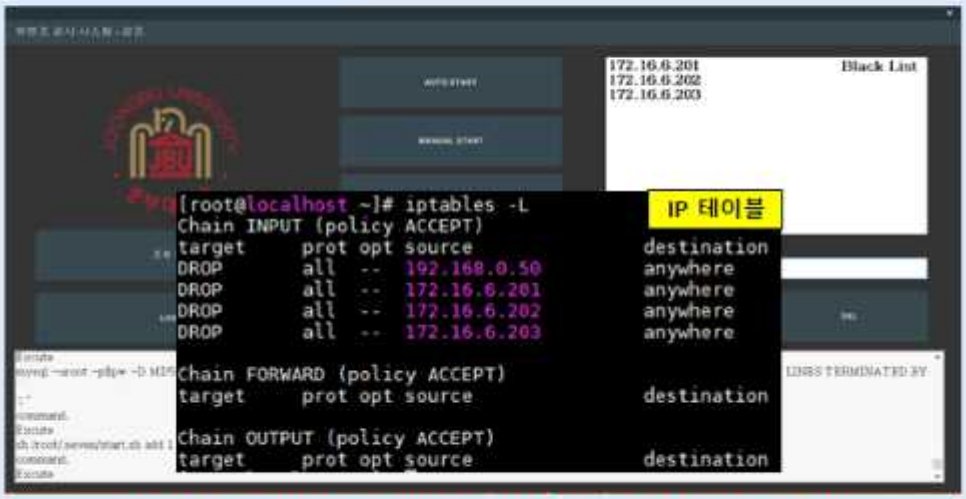
❖ 관리자가 하위서버에 hash값 요청 후 하위서버의 hash값을 DB에 저장

❖ 관리서버 DB의 테이블에 hash값과 경로를 관리

14

개발 환경 및 시스템 개발(8/11)

블랙리스트 관리



블랙리스트 등록시 관리서버 및 하위서버 iptables에서 IP 관리

15

개발 환경 및 시스템 개발(9/11)

위·변조 감시(1/2)

감시 스크립트

```
#!/bin/bash

case "$1" in

        start)
        pschv`ps -ef | grep $0 | grep -c start`

        if [ $psch -gt "2" ]
        then
                echo "already working"
        exit 1
        fi


        if [ -f /root/md5test.txt ]
        then
                echo "already copied"
        else
        find /etc/ -type f -printf '"%p"\n'|xargs md5sum -b > ./md5test.txt
        chmod 400 ./md5test.txt
        fi

while true
        do
                mdch1="./md5test.txt"
                mdch2="./md5ori.txt"
                if [ -z "`diff $mdch1 $mdch2`" ]
                then
                        echo "변경된 값이 없습니다."
                else
                        echo "hash file changed"
                        echo ""
```

실시간 감시 및 공격자 파일 교체시 관리자에게 알림

16

개발 환경 및 시스템 개발(10/11)

위·변조 감시(2/2)

감시 스크립트

```
status)
psch=`ps -ef |grep $0 |grep -c start`
if [ $psch = "1" ]
        then
        echo "working"
else
        echo "maybe stopped"
fi
;;


stop)
psch=`ps -ef |grep $0 |grep -c start`
if [ $psch = "1" ]
        then
        kill -9 `ps -ef | grep $0 | grep start | awk '{print $2}'`
else
        echo "already stopped"
fi
;;


add)
echo "enter the ip address"
read vicip
iptables -A INPUT -s $vicip -j DROP
;;


del)
read vicip
iptables -A INPUT -s $vicip -j ACCEPT
;;
```

의심 IP 차단, 관리자의 IP 허용, 차단한 IP 해제 등 조치

17

개발 환경 및 시스템 개발(11/11)

Telegram 알림

Telegram 통신 프로그램

```
# message reply function
def get_message(bot, update) :
    update.message.reply_text('get text')
    update.message.reply_text(update.message.text)

def start_detection(bot,update):
    update.message.reply_text('start')
    t = Thread(target=enqueue_output, args=(update,))
    t.daemon = True
    t.start()
    update.message.reply_text(update.message.text)

def enqueue_output(update):
    global isRun
    isRun = True
    while isRun:
        print isRun
        p = subprocess.Popen('start', shell=True, stdout=subprocess.PIPE)
        text = p.stdout.read()
        if text != '':
```

```
def end_detection(bot, up
    global isRun
    isRun= False
    update.message.reply_text('end')
    update.message.reply_text(isRun)

t_record(bot,update):
ate.message.reply_text('Record the high range')
= subprocess.Popen('arecord -d 5', shell=True, stdout=subprocess.PIPE
ate.message.reply_text(update.message.text)

t_check(bot,update):
ate.message.reply_text('check')
p2 = subprocess.Popen('find -ctime -1', shell=True, stdout=subprocess.PIPE)
text1 = p2.stdout.read()
if text1 != '':
    update.message.reply_text(text1)
```

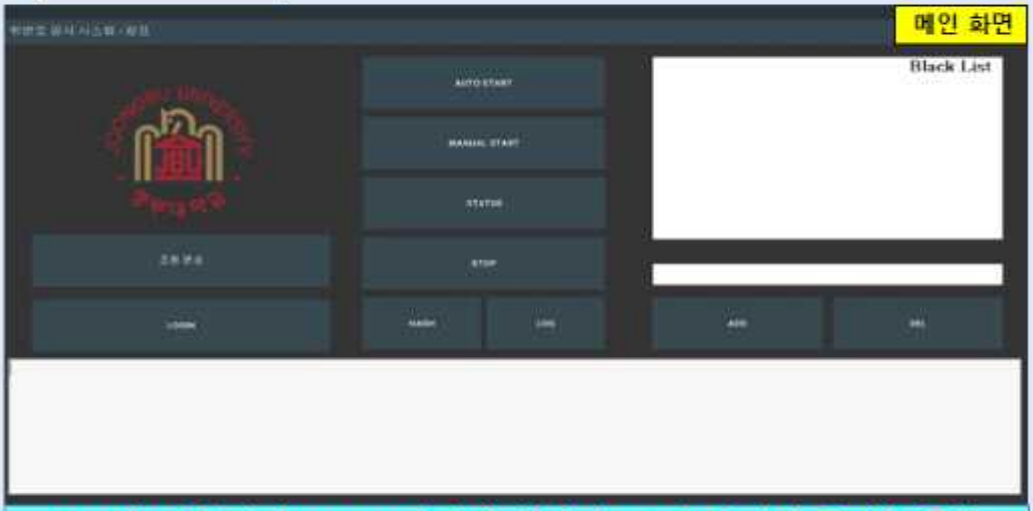
❖Telegram을 이용, 관리자 알림
- 위·변조 상황 안내
- 외부에서 관리서버 접속알림
※ IP, 접속시간, 국가, 계정 등

❖Telegram을 이용, 명령 전달
- 관리서버 → 하위서버

18

개발 시스템 운영(1/7)

시스템 기동



로그인, 시작/정지, HASH 값 출력, 블랙리스트 추가/삭제 등 기능 구현

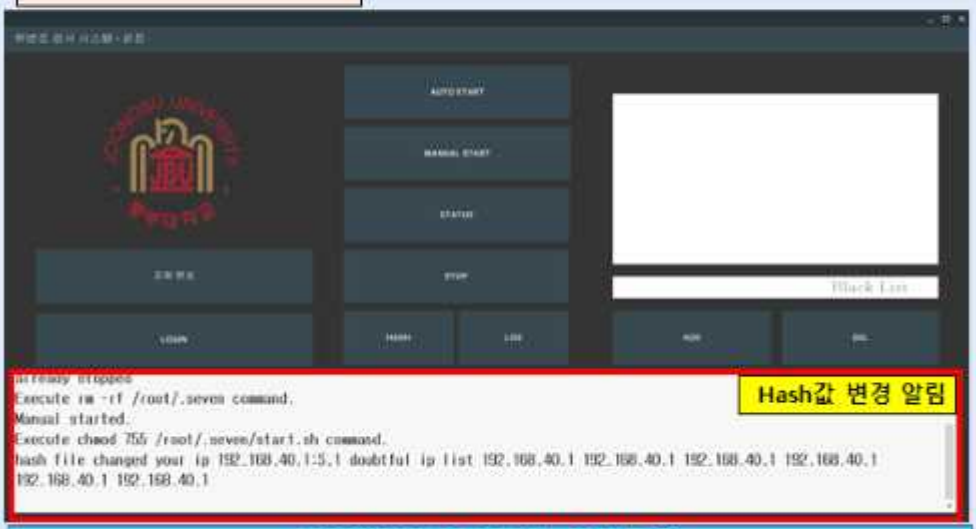
개발 시스템 운영(2/7)

관리서버 로그인

관리서버의 ID(유저명@IP주소), 패스워드(유저PW) 입력 시
정상적으로 로그인

개발 시스템 운영(3/7)

감시 시스템 동작

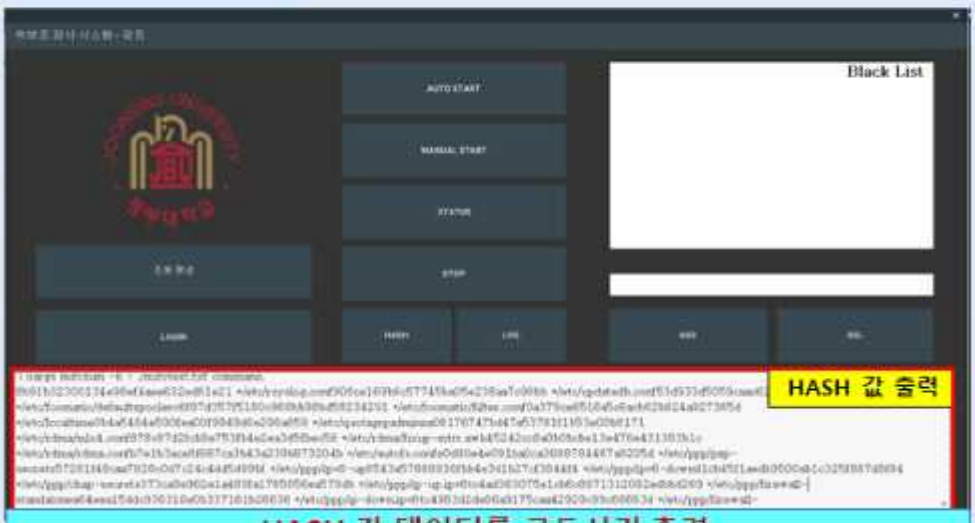


MANUAL START시 1회 동작

21

개발 시스템 운영(4/7)

HASH 값 관리

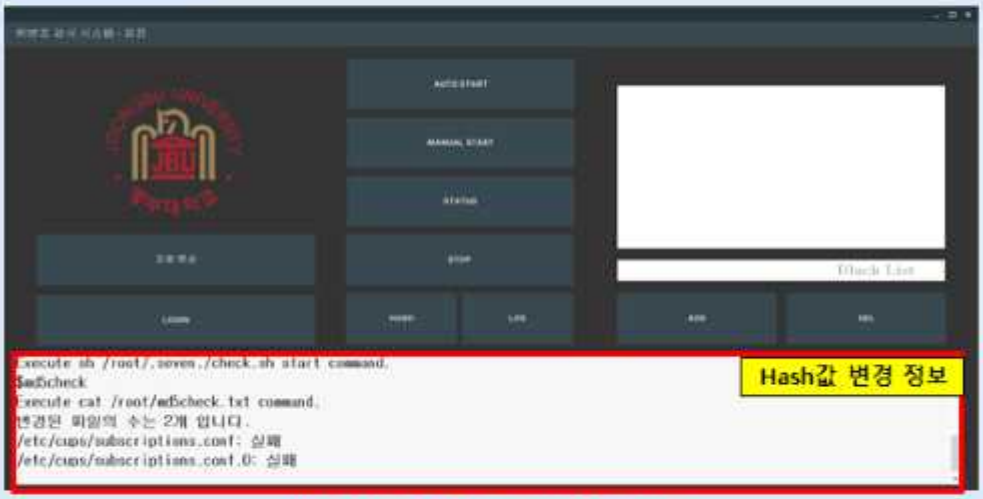


HASH 값 데이터를 로드시켜 출력

22

개발 시스템 운영(5/7)

로그 확인



변경된 HASH 값 정보를 확인

23

개발 시스템 운영(6/7)

블랙리스트 관리

공격자로 의심되는 IP를 관리자가 추가 및 삭제 가능

24

개발 시스템 운영(7/7)

Telegram 알림

텔레그램을 활용하여 실시간 정보 알림 및 명령 전달 가능

25

결론 및 기대효과

❖ 위/변조 감시 시스템 개발 성과

- 개발 시스템은 외부 공격자의 위/변조 공격을 감시, 관리자에게 실시간 알리는 등 즉각적인 대응이 가능하게 하는 보안관리 환경을 제공

- 또한 감시 자료를 활용하여 취약점 분석에 활용하게 하는 등 관련 기능을 부가하여 시스템의 기능을 확대

❖ 기대 효과

- 시스템을 GUI 방식으로 구현하여 비전문가도 쉽게 활용할 수 있을 것으로 기대

- 모든 조원들이 임무를 분담하여 맡은 부분의 시스템을 책임성 있게 구현, 실무 능력 향상시키는 계기

26

6.2 소스코드

adminmenu.sh

```
#!/bin/bash
menu()
{
        echo " ┌──────────────────────────────────────────────────────────
──────┐ "
        echo " |                                                  | "
        echo " |    프로그램감시시스템Ver.관리자용      | "
        echo " |                                                  | "
        echo " ├──────────────────────────────────────────────────────────
──────┤ "
        echo " |                                                  | "
        echo " |                                                  | "
        echo " |    1.하위 서버 Hash값 가져오기          | "
        echo " |                                                  | "
        echo " |                                                  | "
```

```
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "|                                        |"
        echo "|                                        |"
        echo "|    2.가져온 Hash값 확인하기              |"
        echo "|                                        |"
        echo "|                                        |"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "|                                        |"
        echo "|                                        |"
        echo "|    3.하위 서버 업데이트                  |"
        echo "|                                        |"
        echo "|                                        |"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "|                                        |"
        echo "|                                        |"
        echo "|    4.하위 서버 침입감지실행              |"
        echo "|                                        |"
        echo "|                                        |"
        echo "└──────────────────────────────────────────────────────────────
──────┘"
        echo "├──────────────────────────────────────────────────────────────
──────┤"
        echo "|                                        |"
        echo "|                                        |"
        echo "|    5.(        )                        |"
```

```
        echo " |                                                    | "
        echo " |                                                    | "
         echo " └────────────────────────────────────────────────────
─────────┘ "
}
while :
do
        menu
        echo "============================="
        echo "사용할 시스템을 선택해주세요."
        echo "============================="
        echo "종료하기 : q , quit"
        echo "============================="
        echo " 홈 : h , home"
        echo "============================="
        read MenuCase
        case "$MenuCase" in
                "q" | "quit"    ) echo "종료 합니다."
                          exit 1;;
                "h" | "home"    ) echo "홈으로 돌아갑니다."
                          menu;;
                "1"     ) echo "Hash값을 가져옵니다."
                          ./adminmd5.sh;;
                "2"     ) echo "Hash값을 확인합니다."
                          ./adminshow.sh;;
                "3"     ) echo "하위 서버 업데이트를 실행합니다."
                          ./adminsave.sh;;
                "4"     ) echo "하위 서버 침입감지를 실행합니다."
                          ./insExc.sh;;
                *       ) echo "다시 입력 해주세요."
        esac
done
```

adminmd5.sh

```
#!/bin/bash
ssh root@(IP주소) ./md5.sh
scp root@(IP주소):/root/md5test.txt /var/lib/mysql-files/log/
read pw
mysql -uroot -p$pw -D MD5 -e"LOAD DATA INFILE '/var/lib/mysql-files/log/md5test.txt' INTO TABLE submd5 FIELDS TERMINATED BY ' ' LINES TERMINATED BY '₩n';"
```

adminsave.sh

```
#!/bin/bash
ssh root@(IP주소) ./save.sh
```

insExc.sh

```
#!/bin/bash
ssh root@(IP주소) "./intDtc.sh start &"
```

check.sh

```
#!/bin/bash
mysql -uroot -p$pw -D MD5 -e"SELECT * FROM submd5 INTO OUTFILE '/var/lib/mysql-files/md5ori.txt' FIELDS TERMINATED BY ' ' LINES TERMINATED BY '₩n';"
md5=`find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >./md5test.txt`;
md5check=`md5sum -c ./md5test.txt|sed -e '/OK/d' -e '/성공/d'`
diff md5ori.txt md5text.txt | grep "^>" | cut -d ""-f2,3
echo $md5check
```

save.sh

```
#/bin/bash
read pw
mysql -uroot -p$pw -D MD5 -e"SELECT * FROM submd5 INTO OUTFILE '/var/lib/mysql-files/md5ori.txt' FIELDS TERMINATED BY ' ' LINES TERMINATED BY '₩n';"₩
ori="/var/lib/mysql-files/md5ori.txt"
copy="./md5copy.txt"
cp "$ori" "$copy"
md5compare=`find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b
```

```
>./md5compare.txt`;
compare="./md5compare.txt"
diff=`diff ./md5copy.txt ./md5compare.txt >./diffResult.txt`;
res="./diffResult.txt"
if [ -z "`diff $copy $compare`" ]
then
        echo "변경된 값이 없습니다."
else
        echo "변경된 값이 있습니다."
        cp "$compare" "$ori"
fi
rm -rf /var/lib/mysql-files/md5ori.txt
```

subsrvmenu.sh

```
#!/bin/bash
menu()
{
        echo "┌──────────────────────────────────────────────────────────
────────┐"
        echo "|                                                |"
        echo "|          프로그램감시시스템                      |"
        echo "|                                                |"
        echo "├──────────────────────────────────────────────────────────
────────┤"
        echo "|                                                |"
        echo "|                                                |"
        echo "|     1.Hash값 가져오기                           |"
        echo "|                                                |"
        echo "|                                                |"
        echo "├──────────────────────────────────────────────────────────
────────┤"
        echo "├──────────────────────────────────────────────────────────
────────┤"
        echo "|                                                |"
```

```
        echo "|                                    |"
        echo "|    2.가져온 Hash값 확인하기          |"
        echo "|                                    |"
        echo "|                                    |"
        echo "├──────────────────────────────────────────────────────────────────────────────┤"
        echo "├──────────────────────────────────────────────────────────────────────────────┤"
        echo "|                                    |"
        echo "|                                    |"
        echo "|    3.업데이트                       |"
        echo "|                                    |"
        echo "|                                    |"
        echo "├──────────────────────────────────────────────────────────────────────────────┤"
        echo "├──────────────────────────────────────────────────────────────────────────────┤"
        echo "|                                    |"
        echo "|                                    |"
        echo "|    4.침입감지실행                   |"
        echo "|                                    |"
        echo "|                                    |"
        echo "└──────────────────────────────────────────────────────────────────────────────┘"
        echo "├──────────────────────────────────────────────────────────────────────────────┤"
        echo "|                                    |"
        echo "|                                    |"
        echo "|    5.Hash값 관리서버로 전송          |"
        echo "|                                    |"
        echo "|                                    |"
        echo "└──────────────────────────────────────────────────────────────────────────────┘"
}
```

```
while :
do
        menu
        echo "============================="
        echo "사용할 시스템을 선택해주세요."
        echo "============================="
        echo "종료하기 : q , quit"
        echo "============================="
        echo " 홈 : h , home"
        echo "============================="
        read MenuCase
        case "$MenuCase" in
                "q" | "quit"    ) echo "종료 합니다."
                        exit 1;;
                "h" | "home"    ) echo "홈으로 돌아갑니다."
                        menu;;
                "1"     ) echo "Hash값을 가져옵니다."
                        ./md5.sh;;
                "2"     ) echo "Hash값을 확인합니다."
                        ./md5show.sh;;
                "3"     ) echo "Hash값 업데이트를 실시합니다. "
                        ./save.sh;;
                "4"     ) echo "침입감지를 실행합니다."
                        ./intDtc.sh start &;;
                "5"     ) echo "Hash값을 관리서버로 전송합니다. "
                        ./subServer.sh;;
                *    ) echo "다시 입력 해주세요."
        esac
done
```

subServer.sh

```
#!/bin/bash
```

```
ori="./md5test.txt"
copy="./md5copy.txt"
cp "$ori" "$copy"
NOW=$(date +"%Y%m%d%H%M%S")
md5log=`find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >./md5compare$NOW.txt`;
md5compare=`find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >./md5compare.txt`;
compare="./md5compare.txt"
log="./md5compare$NOW.txt"
if [ -z "`diff $copy $compare`" ]
then
        echo "변경된 값이 없습니다."
        scp $log root@(IP주소):/root
else
        echo "변경된 값이 있습니다."
        #diff $copy $compare
        cp "$compare" "$ori"
        scp $log root@(IP주소):/root
fi
```

intDtc.sh

```
#!/bin/bash
case "$1" in
        start)
        psch=`ps -ef | grep $0 | grep -c start`
        if [ $psch -gt "2" ]
        then
                echo "already working"
        exit 1
        fi
        if [ -f /root/md5test.txt ]
        then
```

```
                echo "already copied"
        else
        find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >/root/md5test.txt
        chmod 400 /root/md5test.txt
        fi
while true
        do
                mdch1="/root/md5test.txt"
                mdch2="/root/md5ori.txt"
                if [ -z "`diff $mdch1 $mdch2`"]
                then
                        echo "변경된 값이 없습니다."
                else
                        echo "hash file changed"
                        echo ""
                          echo "your ip `who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e
$'s/)//g'`"
                        echo ""
                        yip=`who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e $'s/)//g'`
                        echo ""
                        echo "doubtful ip list"
                           echo `last | grep -v $yip | grep -v ":0.0" | awk '{print $3}' |
grep &v [a-z] | head`
                fi
        sleep 10
        done <"/root/md5test.txt"
        ;;
        status)
        psch=`ps -ef |grep $0 |grep -c start`
        if [ $psch = "1" ]
                then
                echo "working"
        else
```

```
                echo "maybe stopped"
        fi
        ;;
        stop)
        psch=`ps -ef |grep $0 |grep -c start`
        if [ $psch = "1" ]
                then
                kill -9 `ps -ef | grep $0 | grep start | awk '{print $2}'`
        else
                echo "already stopped"
        fi
        ;;
        add)
        echo "enter the ip address"
        read vicip
        iptables -A INPUT -s $vicip -j DROP
        ;;
        del)
        read vicip
        iptables -D INPUT -s $vicip -j DROP
        ;;
        *)
        echo $"$0 start.. {start | status | stop | add | del}"
        exit 1
esac
```

md5.sh

```
#!/bin/bash
md5=`find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >./md5ori.txt`;
echo $md5
```

forgery.py

```
from threading import Thread
from telegram.ext import Updater, MessageHandler, Filters, CommandHandler  #
```

```
import modules
import subprocess
from Queue import Queue, Empty
from time import sleep
my_token = '668710228:AAE_G51yd0DpufvGVIgC20YLKSS2OAOPf5s'
updater = Updater(my_token)
print('start telegram chat bot')
isRun = True
isRun1 = True
q = Queue()
# message reply function
def get_message(bot, update) :
    update.message.reply_text("got text")
    update.message.reply_text(update.message.text)
def start_detection(bot,update):
    update.message.reply_text("got /start")
    t = Thread(target=enqueue_output, args=(update,))
    t.daemon = True
    t.start()
    update.message.reply_text(update.message.text)
def enqueue_output(update):
    global isRun
    isRun = True
    while isRun:
        print isRun
                p = subprocess.Popen('exec ./detection.sh', shell=True,
stdout=subprocess.PIPE)
        text = p.stdout.read()
        if text != "":
            update.message.reply_text(text)
        sleep (3)
def end_detection(bot, update):
```

```
    global isRun
    isRun= False
    update.message.reply_text("got /end")
    update.message.reply_text(isRun)
def start_record(bot,update):
    update.message.reply_text("Record the hash value.")
            p1 = subprocess.Popen('exec ./record.sh', shell=True,
stdout=subprocess.PIPE)
    update.message.reply_text(update.message.text)
def start_check(bot,update):
    update.message.reply_text("check")
    t1 = Thread(target=enqueue_output1, args=(update,))
    t1.daemon = True
    t1.start()
def enqueue_output1(update):
    p2 = subprocess.Popen('exec ./check.sh', shell=True, stdout=subprocess.PIPE)
    text1 = p2.stdout.read()
    if text1 != "":
        update.message.reply_text(text1)
start_handler = CommandHandler('start_detection', start_detection)
updater.dispatcher.add_handler(start_handler)
end_handler = CommandHandler('end_detection', end_detection)
updater.dispatcher.add_handler(end_handler)
start_handler = CommandHandler('start_record', start_record)
updater.dispatcher.add_handler(start_handler)
start_handler = CommandHandler('start_check', start_check)
updater.dispatcher.add_handler(start_handler)
message_handler = MessageHandler(Filters.text, get_message)
updater.dispatcher.add_handler(message_handler)
updater.start_polling(timeout=3, clean=True)
updater.idle()
```

detection.sh

```
#/bin/bash
        mdch1="./md5test.txt"
        mdch2="./md5ori.txt"
        if [ -z "`diff $mdch1 $mdch2`" ]
        then
                echo `cat /home/seo/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                echo `cat /root/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                echo "변경된 값이 없습니다."
        else
                echo `cat /home/seo/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                echo `cat /root/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                echo "hash file changed"
                echo ""
                  echo "your ip `who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e $'s/)//g'`"
                echo ""
                yip=`who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e $'s/)//g'`
                echo ""
                echo "doubtful ip list"
                   echo `last | grep -v $yip | grep -v ":0.0" | awk '{print $3}' | grep -v [a-z] | head`
        fi
```

record.sh

```
#/bin/bash
 if [ -f /kao/md5test.txt ]
        then
                echo "already copied"
        else
        find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >./md5test.txt
        chmod 400 ./md5test.txt
```

```
        fi
```

check.sh

```
#!/bin/bash
md5check=`md5sum -c ./md5test.txt|sed -e '/OK/d' -e '/성공/d' >./md5check.txt`;
echo $md5check
echo "변경된 파일의 수는 `cat ./md5check.txt |wc -l`개 입니다."
echo "`cat ./md5check.txt`"
test.txt
        fi
```

Form1.cs

```
using Renci.SshNet;
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using System.Windows.Forms;
using MaterialSkin.Controls;
using MaterialSkin;
using System.Diagnostics;
namespace seven
{
    public partial class Form1 : MaterialForm
    {
        ScpClient mScpClient = null;
        SshClient mSshClient = null;
        string mHost = string.Empty;
        string mUserName = string.Empty;
        string mPassword = string.Empty;
        PrivateKeyFile mPPK = null;
        System.Windows.Forms.Timer mTimer;
           Dictionary<string, EventHandler> mCmdHandlers = new Dictionary<string,
```

```
EventHandler>();
        public Form1()
        {
            InitializeComponent();
            var materialSkinManager = MaterialSkinManager.Instance;
            materialSkinManager.AddFormToManage(this);
            materialSkinManager.Theme = MaterialSkinManager.Themes.DARK;
            pictureBox1.ImageLocation = "Resource₩₩dddd.png";

            pictureBox1.SizeMode = PictureBoxSizeMode.CenterImage;
            this.MaximizeBox = true;
            this.MinimizeBox = true;
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            mCmdHandlers.Add("autostart", OnCmd_AutoStart);
            mCmdHandlers.Add("manualstart", OnCmd_ManualStart);
            mCmdHandlers.Add("stop", OnCmd_Stop);
            mCmdHandlers.Add("status", OnCmd_Status);
            mCmdHandlers.Add("add", OnCmd_Add);
            mCmdHandlers.Add("del", OnCmd_Del);
            mCmdHandlers.Add("adminmd5", OnCmd_Adminmd5);
            mCmdHandlers.Add("login", OnCmd_Login);
            mCmdHandlers.Add("hash", OnCmd_Hash);
            mCmdHandlers.Add("check", OnCmd_Check);
            mTimer = new System.Windows.Forms.Timer();

            mTimer.Interval = 10000;
            mTimer.Tick += new EventHandler(timer_Tick);
        }
        void timer_Tick(object sender, EventArgs e)

        {
```

```
    RunCommand("sh "+ GetScriptsPath() + "/start.sh start");
}
private void AppendToOutputTextBox(string str)

{
    outputTextBox.AppendText(str + Environment.NewLine);
}
private void OnCmd_Login(object sender, EventArgs e)

{
}
private void OnCmd_AutoStart(object sender, EventArgs e)

{
    if (mTimer.Enabled == true)
    {
        return;
    }
    mTimer.Start();
    RunCommand($"chmod 755 /root/.seven/start.sh");
    AppendToOutputTextBox("Auto started.");
}
private void OnCmd_ManualStart(object sender, EventArgs e)

{
    AppendToOutputTextBox("Manual started.");
    RunCommand($"chmod 755 /root/.seven/start.sh");
    RunCommand("sh "+ GetScriptsPath() + "/start.sh start");
}

private void OnCmd_Stop(object sender, EventArgs e)

{
```

```
    Button ManualStart = sender as Button;
    if (mTimer.Enabled == false | ManualStart != null)
    {
        return;
    }
    mTimer.Stop();
    AppendToOutputTextBox("stopped.");
}
private void OnCmd_Status(object sender, EventArgs e)

{
    if (mTimer.Enabled == true)
    {
        AppendToOutputTextBox("working");
    }
    Button ManualStart = sender as Button;
    if (ManualStart != null)
    {
        AppendToOutputTextBox("working");
    }
    else
    {
        AppendToOutputTextBox("maybe stopped");
    }
}
private void OnCmd_Add(object sender, EventArgs e)

{
    if (ipListBox.Items.Contains(ipTextBox.Text) == true)
    {
        AppendToOutputTextBox("이미 등록된 IP 입니다.");
    }
    if (string.IsNullOrEmpty(ipTextBox.Text) == true)
```

```
    {
        AppendToOutputTextBox("IP 주소를 입력해주세요.");
    }
    else
    {
        RunCommand($"sh {GetScriptsPath()}/start.sh add {ipTextBox.Text}");
        ipListBox.Items.Add(ipTextBox.Text);
    }
}
private void OnCmd_Del(object sender, EventArgs e)
{
    if (ipListBox.Items.Contains(ipTextBox.Text) == false)
    {
        AppendToOutputTextBox("등록되지 않은 IP 입니다.");
    }
    if (string.IsNullOrEmpty(ipTextBox.Text) == true)
    {
        AppendToOutputTextBox("IP 주소를 입력해주세요. (add or del)");
    }
    else
    {
        RunCommand($"sh {GetScriptsPath()}/start.sh del {ipTextBox.Text}");
        ipListBox.Items.Remove(ipTextBox.Text);
    }
}
private void OnCmd_Hash(object sender, EventArgs e)

{
}
private void OnCmd_Adminmd5(object sender, EventArgs e)

{
       RunCommand("find /etc/ -type f -printf '₩"%p₩"₩n'|xargs md5sum -b
```

```
>./md5test.txt");
                RunCommand("mysql -uroot -p$pw -D MD5 -e₩"LOAD DATA INFILE
'/var/lib/mysql-files/md5test.txt' INTO TABLE submd5 FIELDS TERMINATED BY ' ' LINES
TERMINATED BY '₩n'; ₩"");
        }
        /*
        private void OnCmd_Check(object sender, EventArgs e)
        {
            RunCommand($"{GetScriptsPath()} /etc/passwd/check.sh {ipTextBox.Text}");
            AppendToOutputTextBox("변경사항이 없습니다.");
            try
            {
                                                        string sysFolder =
Environment.GetFolderPath(Environment.SpecialFolder.System);
                ProcessStartInfo pInfo = new ProcessStartInfo();
                pInfo.FileName = sysFolder + @"₩check.sh";
                pInfo.UseShellExecute = true;
                Process p = Process.Start(pInfo);

                  var ret = mSshClient.RunCommand(string.Format("[! -d {0} ] &&echo
not_found", GetScriptsPath()));
                if (ret.Result == "not_found₩n")
                {
                    RunCommand($"mkdir {GetScriptsPath()}");
                    }
                else
                {
                        RunCommand($"sh {GetScriptsPath() + "/etc/passwd/check.sh"}
start");
                    RunCommand($"rm -rf {GetScriptsPath()}");
                }
                                            mScpClient.RemotePathTransformation =
RemotePathTransformation.ShellQuote;
```

```
                    mScpClient.Upload(new DirectoryInfo(Directory.GetCurrentDirectory() +
"₩₩Scripts"), GetScriptsPath());
                }
            catch (Exception ex)
            {
                AppendToOutputTextBox($"다시 시도해주세요. ({ex.Message})");
            }
        }
          */
        private void OnCmd_Check(object sender, EventArgs e)
        {
            // AppendToOutputTextBox("check start.");
            RunCommand("sh "+ GetScriptsPath() + "./check.sh start");

                var md5check = mSshClient.RunCommand(string.Format("`md5sum -c
/root/md5test.txt | sed -e 'OK/d' -e '/성공/d' >/root/md5check.txt`"));
            AppendToOutputTextBox("$md5check");
            if (md5check.Result == null)
            {
                AppendToOutputTextBox("변경된 파일이 없습니다.");
            }
            else
            {
                RunCommand("cat /root/md5check.txt");
            }
        }
        private string RunCommand(string cmd)

        {
            string command = cmd + ">log.txt";
            SshCommand ret = mSshClient.RunCommand(command);
             AppendToOutputTextBox("Execute "+ cmd + "command.");
```

```
    if (string.IsNullOrEmpty(ret.Result) == false)
    {
        AppendToOutputTextBox(ret.Result);
    }
    else
    {
        ret = mSshClient.RunCommand("cat log.txt");

        if (string.IsNullOrEmpty(ret.Result) == false)
        {
            var ret1 = ret.Result.Replace("₩n", string.Empty);
            AppendToOutputTextBox(ret1);
        }
    }
    return ret.Result;
}
private string GetScriptsPath()
{
    return "/root/.seven";
}
private bool CheckConnectedSsh()
{
    if (mSshClient == null ||
       mSshClient.IsConnected == false ||
       mScpClient == null ||
       mScpClient.IsConnected == false)
    {
        AppendToOutputTextBox("연결되어 있지 않습니다.");
        return false;
    }
    return true;
}
```

```
private void AutoStartBtn_Click(object sender, EventArgs e)

{
    if (CheckConnectedSsh() == false)
    {
        return;
    }
    mCmdHandlers["autostart"]?.Invoke(this, null);
}
private void ManualStartBtn_Click(object sender, EventArgs e)

{
    if (CheckConnectedSsh() == false)
    {
        return;
    }
    mCmdHandlers["manualstart"]?.Invoke(this, null);
}
private void StatusBtn_Click(object sender, EventArgs e)
{
    if (CheckConnectedSsh() == false)
    {
        return;
    }
    mCmdHandlers["status"]?.Invoke(this, null);
}
private void StopBtn_Click(object sender, EventArgs e)
{
    if (CheckConnectedSsh() == false)
    {
        return;
    }
```

```
        mCmdHandlers["stop"]?.Invoke(this, null);
    }
    private void HashBtn_Click(object sender, EventArgs e)
    {
        if (CheckConnectedSsh() == false)
        {
            return;
        }
        mCmdHandlers["adminmd5"]?.Invoke(this, null);
    }
    private void CheckBtn_Click(object sender, EventArgs e)

    {
        if (CheckConnectedSsh() == false)
        {
            return;
        }
        mCmdHandlers["check"]?.Invoke(this, null);
    }

    private void AddBtn_Click(object sender, EventArgs e)
    {
        if (CheckConnectedSsh() == false)
        {
            return;
        }
        mCmdHandlers["add"]?.Invoke(this, null);
    }
    private void DelBtn_Click(object sender, EventArgs e)
    {
        if (CheckConnectedSsh() == false)
        {
```

```
                return;
            }
            mCmdHandlers["del"]?.Invoke(this, null);
        }
        private void outputTextBox_TextChanged(object sender, EventArgs e)

        {
            outputTextBox.SelectionStart = outputTextBox.Text.Length;
            outputTextBox.ScrollToCaret();
        }
        private void LoginBtn_Click(object sender, EventArgs e)

        {
            var loginForm = new Form2();
            loginForm.ShowDialog();

            var splitString = loginForm.UserName.Split('@');
            if (splitString.Length != 2)
            {
                                        AppendToOutputTextBox("유효하지 않은 호스트입니다.
(UserName@Host)");
            }
            mUserName = splitString[0];
            mHost = splitString[1];
            mPassword = loginForm.Password;
            try
            {
                if (string.IsNullOrEmpty(loginForm.PpkFilePath) == true)
                {
                    mSshClient = new SshClient(mHost, mUserName, mPassword);
                    mScpClient = new ScpClient(mHost, mUserName, mPassword);
                }
                else
```

```
                {
                    mPPK = new PrivateKeyFile(loginForm.PpkFilePath);
                    mSshClient = new SshClient(mHost, mUserName, mPPK);
                    mScpClient = new ScpClient(mHost, mUserName, mPPK);
                }
                mSshClient.ConnectionInfo.Timeout = new TimeSpan(0, 0, 5);
                mScpClient.ConnectionInfo.Timeout = new TimeSpan(0, 0, 5);
                AppendToOutputTextBox("로그인 중...");
                mSshClient.Connect();
                mScpClient.Connect();
                AppendToOutputTextBox("로그인 성공");

                 var ret = mSshClient.RunCommand(string.Format("[ ! -d {0} ] &&echo
not_found", GetScriptsPath()));
                if (ret.Result == "not_found₩n")
                {
                    RunCommand($"mkdir {GetScriptsPath()}");
                }
                else
                {
                    RunCommand($"sh {GetScriptsPath() + "/start.sh"} stop");
                    RunCommand($"rm -rf {GetScriptsPath()}");
                }
                                                    mScpClient.RemotePathTransformation =
RemotePathTransformation.ShellQuote;
                  mScpClient.Upload(new DirectoryInfo(Directory.GetCurrentDirectory() +
"₩₩Scripts"), GetScriptsPath());
            }
            catch (Exception ex)
            {
                AppendToOutputTextBox($"로그인 실패. ({ex.Message})");
            }
        }
```

```
        private void label1_Click(object sender, EventArgs e)
        {
        }
        private void Zoone_Click(object sender, EventArgs e)
        {
            var Form = new Form3();
              RunCommand($"류현호 91316941₩n서규현 91317036₩n조은서 91413309₩n
장정윤 91421772₩n설희운 91406978");
        }
        private void ipListBox_SelectedIndexChanged(object sender, EventArgs e)
        {
        }
        private void outputTextBox_TextChanged_1(object sender, EventArgs e)
        {
        }
    }
}
```

Form2.cs

```
using Renci.SshNet;
using System;
using System.Windows.Forms;
using MaterialSkin.Controls;
namespace seven
{
    public partial class Form2 : Form
    {
        public string UserName = "root@";
        public string Password = "";
        public string PpkFilePath = null;
        public Form2()
        {
            InitializeComponent();
```

```
        }
        private void Form2_Load(object sender, EventArgs e)
        {
            UserNameBox.Text = UserName;
            pwBox.Text = Password;
            openFileDialog1.Filter = "Private Key Files(*.ppk)|*.ppk";
        }
        private void OpenPpkBtn_Click(object sender, EventArgs e)
        {
            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                ppkBox.Text = openFileDialog1.FileName;
            }
        }
        private void LoginBtn_Click(object sender, EventArgs e)
        {
            UserName = UserNameBox.Text;
            Password = pwBox.Text;
            if(string.IsNullOrEmpty(ppkBox.Text) == false)
            {
                PpkFilePath = ppkBox.Text;
            }
            Close();
        }
        private void groupBox1_Enter(object sender, EventArgs e)
        {
        }
    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace seven
{
    static class Program
    {
        /// <summary>
        /// 해당 응용 프로그램의 주 진입점입니다.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

start.sh

```
#!/bin/bash
case "$1" in
        start)
        #if [ -f /root/md5test.txt ]
        if [ /root/md5test.txt -ot /root/md5ori.txt ]
        then
                echo "already copied"
        else
        find /etc/ -type f -printf '"%p"₩n'|xargs md5sum -b >/root/md5test.txt
        chmod 400 /root/md5test.txt
        fi
                mdch1="/root/md5test.txt"
```

```
                mdch2="/root/md5ori.txt"
                if [ -z "`diff $mdch1 $mdch2`" ]
                then
                                echo `cat /home/seo/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                        echo `cat /root/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                        echo "변경된 값이 없습니다."
                else
                                echo `cat /home/seo/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                        echo `cat /root/.bash_history | tail | grep "mv₩|cp₩|ftp₩|scp"`
                        echo "hash file changed"
                        echo ""
                          echo "your ip `who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e $'s/)//g'`"
                        echo ""
                        yip=`who -m |awk '{print $5}'|sed -e $'s/(//g'|sed -e $'s/)//g'`
                        echo ""
                        echo "doubtful ip list"
                           echo `last | grep -v $yip | grep -v ":0.0" | awk '{print $3}' | grep -v [a-z] | head`
                fi
                ;;
        stop)
        psch=`ps -ef |grep $0 |grep -c start`
        if [ $psch = "1" ]
                then
                kill -9 `ps -ef | grep $0 | grep start | awk '{print $2}'`
        else
                echo "already stopped"
        fi
        ;;
        add)
```

```
        iptables -A INPUT -s $2 -j DROP
        ;;
        del)
        iptables -D INPUT -s $2 -j DROP
        ;;
        *)
        echo $"$0 start.. {start | status | stop | add | del}"
        exit 1
esac
```