취약점 진단 자동화 시스템

팀 명 : N2TG

지도교수 : 유승재 교수님

팀 장 : 손현수

팀 원 : 이경수

이혜빈

이문호

한규범

유승하

유재명

2018. 11.

중부대학교 정보보호학과

목 차

1. 서론

1-1. ISMS (정보보호관리 체계)란?

ISMS란 기업·기관이 각종 위협으로 부터 주요 정보자산을 보호하기 위해 수립·관리·운영하는 종합적인 체계이다.

< 그림 1-1 : ISMS 인증 로고 >

1-2. ISMS 인증 법적 근거

ISMS(Information·Security·Management·System)정보통신망 이용촉진 및 정보보호 등에 관한 법률(이하 "정보통신망법") 제47조를 법적 근거로 하고 있다.
정보통신망법 시행령은 정보보호 관리체계 인증의 방법·절차·범위, 인증 수수료, 의무 인증 대상자, 인증표시 및 홍보, 인증기관의 지정 등에 대한 사항을 규정하고 있다. 정보보호 관리체계 인증에 관하여 필요한 세부사항은 정보보호 관리체계 인증 등에 관한 고시에서 다루고 있다.

1-3. ISMS 인증심사의 종류

최초심사란 정보보호 관리체계 인증을 처음으로 취득할 때 진행하는 심사이며, 인증의 범위에 중요한 변경이 있어 다시 인증을 신청할 때에도 실시한다. 최초심사를 통해 인증을 취득하면 3년의 유효기간이 부여된다.

사후심사란 인증을 취득한 이후 정보보호 관리체계가 지속적으로 유지되고 있는지 확인하는 것을 목적으로 인증 유효기간 중 매년 1회 이상 시행하는 심사이다. 갱신심사는 정보보호 관리체계 인증 유효기간 연장을 목적으로 심사를 말한다.

< 그림 1-2 : ISMS 인증심사의 종류 >

1-4. ISMS 인증기준

ISMS 인증기준은 관리과정과 정보보호대책으로 구성되어 있다.

정보보호 관리과정은 관리체계의 메인 프레임으로서 전반적인 관리체계 운영 라이프사이클을 구성하고 있다. 정보보호대책과정은 총 13개 분야에 대한 인증기준으로써 정책, 조직, 교육 등 관리적 부문과 개발, 보안통제, 운영 통제 등 물리적·기술적 부문에 대한 정책, 준수 및 검토 등 분야별 관리·운영에 대한 라이프사이클을 구성하고 있다.

< 그림 1-3 : 정보보호 관리체계 인증심사 기준 >

1-5. ISMS 인증대상

임의신청자 : 정보보호 관리체계를 구축·운영하여 적합성 여부의 판단을 원하는 모든 조직
의무대상자 : 정보통신망에 미치는 영향이 크고 사회·경제적 파급력이 큰 조직
*인증 의무대상자는「전기통신사업법」제2조제8호에 따른 전기통신사업자와 전기통신사업 자의 전기통신역무를 이용하여 정보를 제공하거나 정보의 제공을 매개하는 자로서 표2에서 기술한 의무대상자 기준에 하나라도 해당되는 자이다.

[표 2] 정보보호 관리체계 의무대상자 기준

구분	의무대상자 기준
ISP	「전기통신사업법」 제6조제1항에 따른 허가를 받은 자로서 서울특별시 및 모든 광역시에서 정보통신망서비스를 제공하는 자
IDC	정보통신망법 제46조에 따른 집적정보통신시설 사업자
다음의 조건 중 하나라도 해당하는 자	연간 매출액 또는 세입이 1,500억원 이상인 자 중에서 다음에 해당되는 경우 - 「의료법」 제3조의4에 따른 상급종합병원 - 직전연도 12월 31일 기준으로 재학생 수가 1만명 이상인 「고등교육법」 제2조에 따른 학교
	정보통신서비스 부문 전년도(법인인 경우에는 전 사업연도를 말한다) 매출액이 100억원 이상인 자
	전년도 직전 3개월간 정보통신서비스 일일평균 이용자 수가 100만명 이상인 자

※ 정보통신망법 제47조제2항 및 시행령 제49조 참조

< 그림 1-4 : 정보보호 관리체계 의무대상자 기준 >

1-6. ISMS 인증절차 및 단계별 소요기간

[표 7] 인증 절차 및 단계별 소요기간

인증 절차 내용	① 준비			② 심사					③ 인증	
	ISMS 운영	인증 신청	인증 신청	심사 준비	인증 심사	보완 조치	조치 확인	심사 결과보고서 작성	인증위원회 심의 준비	인증위원회 심의 및 인증서 교부
소요 시간	2개월 (최소)	5일	5일	30일	5일	30일	5일	5일	30일	2일

※ 위 소요기간은 평균적인 수치이며, 인증범위 및 규모, 정보보호 환경 등 내·외부 요인에 따라 변동 될 수 있음

< 그림 1-5 : 인증 절차 및 단계별 소요기간 >

1-7. ISMS 기대효과

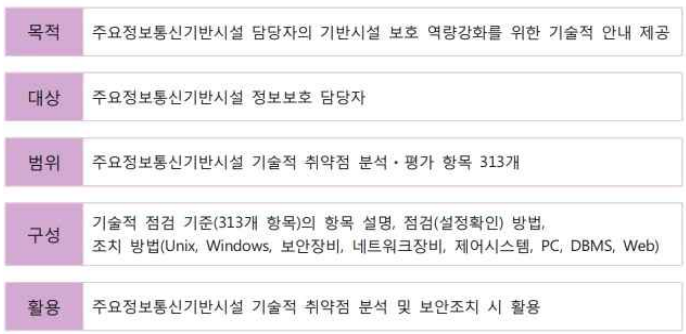
단순 일회적 정보보호대책에서 벗어나 체계적, 종합적인 정보보호 대책을 구현함으로써 기업·기관의 정보보호관리 수준을 향상시킬 수 있다. 기업·기관은 지속적이고 체계적인 정보보호 관리체계 구축을 통해 해킹, DDoS 등의 침해 사고 발생 시 신속하게 대응할 수 있으며, 피해 및 손실을 최소화할 수 있다. 기업·기관 경영진이 직접 정보보호 의사결정에 참여함으로써 정보보호 업무에 대한 책임 성과 신뢰성을 향상시킬 수 있다. 인증을 취득한 기업·기관은 입찰 시 가산점 부여 등의 인센티브를 얻을 수 있다.

1-8. 주요정보통신기반시설이란?

주요정보통신기반시설 관리기관은 [정보통신기반 보호법] 제9조에 따라 주요정보통신 기반시설로 신규 지정된 후 6개월 이내, 매년 취약점 분석 및 평가를 실시하여야 한다. 취약점 분석 및 평가

는 453개의 관리적/물리적/기술적 점검항목에 대한 주요정보통신기반 시설의 취약여부를 점검하여 악성코드 유포, 해킹 등의 사이버 위협 대응을 위한 종합적 개선 과정이다.

1-9. 주요정보통신기반 분석 목적 및 구성



< 그림 1-5 : 주요정보통신기반 분석 목적 및 구성 >

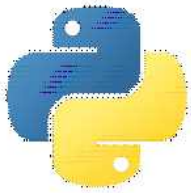
1-10. 목표 및 기대효과

ISMS 인증토대로 파생되어 진 정보통신기반 보호법 기반으로 한 취약점 분석 및 평가 기준 상세 가이드_2017 (18년 기준 최신버젼) 참조. 참조자료를 통한 스크립트 개발 및 해당 스크립트 모듈화 실시. 모듈화로 인해 스크립트 변경 및 유지보수 용의 할 것으로 보여짐. 또한 모듈화를 통한 각각의 취약점분석 보고서를 엑셀형태로 자동으로 변환시켜 실무자들의 편의성 증대 및 점검 대상자들의 비용감소도 이루어질 것으로 보여짐.

2. 관련연구

2-1. Python

파이썬은 네덜란드 개발자 귀도 반 로섬(Guido van Rossum)이 만든 언어다. 귀도 반 로섬은 1989년부터 본격적으로 파이썬을 개발하기 시작했고, 1990년 파이썬의 첫 버전을 공개했다. 현재 파이썬은 대형 글로벌 기업부터 스타트업까지 다양하게 안정적으로 활용되고 있다. 구글, 야후, 유럽 입자 물리 연구소(CERN), 미국항공우주국(NASA) 등이 파이썬을 이용해 서비스를 구축했다. 간결한 문법으로 입문자가 이해하기 쉽고, 다양한 분야에 활용할 수 있기 때문이다. 이 외에도 파이썬은 머신러닝, 그래픽, 웹 개발 등 여러 업계에서 선호하는 언어로 꾸준히 성장하고 있다.



< 그림 2-1 : 파이썬 로고 >

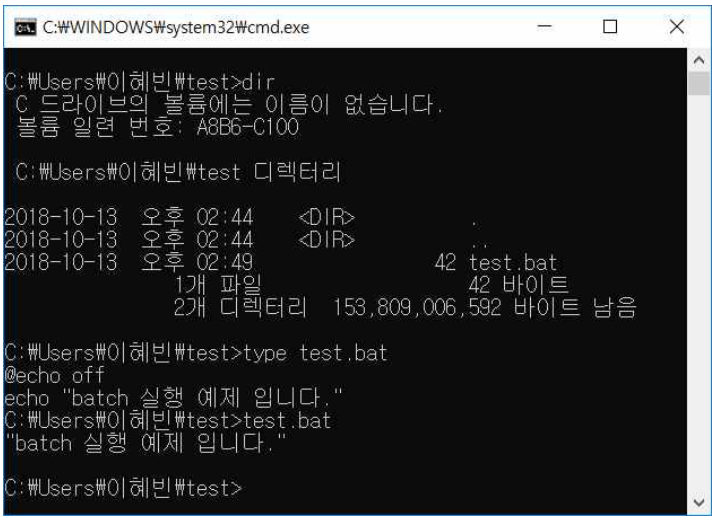
2-2. Shell Script

셸 스크립트(shell script)는 셸이나 명령 줄 인터프리터에서 돌아가도록 작성되었거나 한 운영 체제를 위해 쓰인 스크립트이다. 셸 스크립트가 수행하는 일반 기능으로는 파일 이용, 프로그램 실행, 문자열 출력 등이 있다. 셸 스크립트를 기록하는 것은 다른 프로그래밍 언어의 같은 코드로 쓰인 것보다 훨씬 더 빠른 경우가 많다. 다른 해석 언어에 비해, 셸 스크립트는 컴파일 단계가 없기 때문에 스크립트는 디버깅을 하는 동안 빠르게 실행할 수 있다. 그러나 셸 스크립트를 사용함으로써 몇 가지 중대한 단점이 존재한다. 한 가지 단점으로는 실행되는 각 명령에 대한 잠재적으로 새로운 하부 프로세스의 수많은 필요에 따라 속도가 저하될 수 있다. 단순 sh 스크립트는 다양한 종류의 유닉스, 리눅스, BSD 운영 체제, therof 버전, 시스템 유틸리티와 잘 호환된다는 장점이 있지만 더 복잡한 셸 스크립트는 셸, 유틸리티, 다른 필수 요소 간의 약간의 차이가 많은 경우 실패할 가능성이 있다.

< 그림 2-2 : Shell Script >

2-3. Batch

MS-DOS, OS/2, 윈도우에서 쓰이는 배치 파일(batch file)은 명령 인터프리터에 의해 실행되게끔 고안된 명령어들이 나열되어 있는 텍스트 파일이다. 배치 파일이 실행될 때, COMMAND.COM 또는 cmd.exe와 같은 셸 프로그램이 파일을 읽어 명령어를 줄 단위로 실행한다. 배치 파일은 보통 실행 파일을 자동으로, 연속적으로 실행할 때 유용하며 시스템 관리자가 따분한 일들을 자동화하기 위해 자주 사용한다.



< 그림 2-3 : Batch >

2-4. C++

C++는 처음 소개된 이후로 1985년과 1989년, 그리고 C++에 ANSI표준을 적용하기 시작한 시기 등 3번에 거쳐 개정되었다. 1994년에 최초로 표준안이 발표되었으며 ANSI C++위원회는 사실상 스트루스트럽(C개발자)이 설정한 모든 사양을 그대로 수용하고 별도로 약간의 사양을 덧붙였다. 그렇기 때문에 C++는 C프로그래머가 쉽게 C++를 사용할 수 있다는 관련성에서 큰 장점이 있다. C의 대부분의 특징을 포함하고 있으므로 시스템 프로그래밍에 적합할 뿐만 아니라 클래스, 연산자 중복, 가상 함수 등과 같은 특징을 갖추고 있어 객체 지향 프로그래밍에 적합하다. 또한 C와 일치하는 부분이 C++를 널리 대중적인 언어가 되는데 도움을 주었다.

2-5. dialog

Dialog는 유닉스, 리눅스, POSIX 상에서 간단한 한 줄의 명령어로 동적인 디스플레이 화면을 구성하여 표현할 수 있게 만드는 유틸리티이며, 텍스트 사용자 인터페이스 위젯을 보여주는 셸 스크립트에서 사용되는 응용 프로그램이다. Curses와 Ncurses를 사용한다. 대부분의 GNU/Linux 배포

본에 dialog가 포함되있다. Dialog는 사비오 램(Savio Lam)이 만들었으며 1994년에 0.3 버전으로 처음 올린 것으로 알려졌다. 그 뒤에 몇 명의 사람들에 의해 수정되었다. 1999년 이후로 토마스 디케이에 의해 관리되고 있다.

2-6. GTK

GTK+는 김프 툴킷(GIMP Toolkit)의 준말로, 초기에 김프를 위해서 만든 툴킷이었으며 X 윈도 시스템을 위한 위젯 툴킷 가운데 하나이다. GTK+는 C언어로 작성된 객체지향 위젯 툴킷이다. X11 디스플레이 서버 상에서, GTK+는 위젯들을 그리는데 Xlib를 사용한다. Xlib는 유연하고 X 윈도 시스템이 작동하지 않는 플랫폼에서도 GTK+가 사용될 수 있도록 한다. GTK+는 Qt와 마찬가지로 (다른 많은 위젯 툴킷들과 달리) Xt에 기반을 두지 않는다. 그래서 GTK+를 많은 다른 환경으로 이식할 수 있었다. 하지만 전통적인 X11 응용 프로그램의 사용자 설정 방식인 X 리소스 데이터베이스에 접근할 수 없다는 단점이 있다.

2-7. QT

Qt는 컴퓨터 프로그래밍에서 GUI 프로그램 개발에 널리 쓰이는 크로스 플랫폼 프레임워크이다. 서버용 콘솔과 명령 줄 도구와 같은 비GUI 프로그램 개발에도 사용된다. 그래픽 사용자 인터페이스를 사용하는 경우에는 Qt를 위젯 툴킷으로 분류한다. Qt는 KDE, Qtopia, OPIE에 이용되고 있다, Qt는 C++를 주로 사용하지만, 파이썬, 루비, C, 펄, 파스칼과도 연동된다. 수많은 플랫폼에서 동작하며, 상당히 좋은 국제화를 지원한다. SQL 데이터베이스 접근, XML 처리, 스레드 관리, 단일 크로스 플랫폼 파일 관리 API를 제공한다.

< 그림 2-4 : Qt 로고>

2-8. Windows Server

윈도우 서버(Windows Server)는 마이크로소프트사가 공개한 서버 운영 체제의 상품 이름이다. 윈도우 서버는 여러 가지 버전이 있다. 윈도우 2000서버는 윈도우 2000 기반으로써 현재 지원이 종료되었고, 윈도우 서버 2003 (R2)는 윈도우 XP 기반으로 마찬가지로 지원이 종료되었다. 윈도우 서버 2008는 윈도우 VISTA 기반이고, 윈도우 서버 2008 R2는 윈도우 7 기반, 윈도우 서버 2012는 윈도우 8 기반, 윈도우 서버 2012 R2는 윈도우 8.1 기반, 윈도우 서버 2016는 윈도우 10 기반으로 2016년 9월 출시되었다. 윈도우 스몰 비즈니스 서버는 중소기업을 위한 마이크로소프트 서버 통합형 윈도우 서버 기반 운영 체제를 뜻한다. 윈도우 이센셜 비즈니스 서버는 스몰 비즈니스 서버와 비슷한 제품이지만 중간 크기의 기업에서만 쓸 수 있는 운영 체제를 뜻한다. 윈도우 홈 서버는 파일 공유, 스트리밍, 자동화 백업, 원격 제어 등을 위한 홈 서버 운영 체제이다.

2-9. Solaris

솔라리스(Solaris)란 Sun Microsystems에서 개발한 운영체제로써 유닉스의 일종이다. 지원 플랫폼은 Sun의 자체 하드웨어인 SPARC(스팍)플랫폼과 X86 등을 지원한다. 2010년 이후로는 오라클이 Sun을 인수하여 오라클 솔라리스로 불리기도 한다. 솔라리스 10 이전부터는 Sun의 마케팅 정책이 스팍 프로세서와 솔라리스를 함께 파는 정책이었기에 솔라리스의 대중성을 확보하지 못했다는 단점이 있지만 반대로 이 정책으로 인해 기타 다른 OS처럼 하드웨어 관리를 위해 별도의 하드웨어 관리 S/W (BIOS를 모니터링하는 하드웨어 관리용 S/W)설치하지 않아도 OS log나 OBP 만으로도 하드웨어를 제어하는 등 대부분의 장애진단, 조치 등을 할 수 있는 장점이 있다. 가격이 매우 비쌌고 구하기 어려웠기 때문에, 개인사용자들에게는 여전히 친숙하지 않은 OS였지만 1999년 Sun의 Free 배포 정책선언으로 현재는 라이센스 없이도 무료로 사용이 가능하다.

< 그림 2-5 : Solaris >

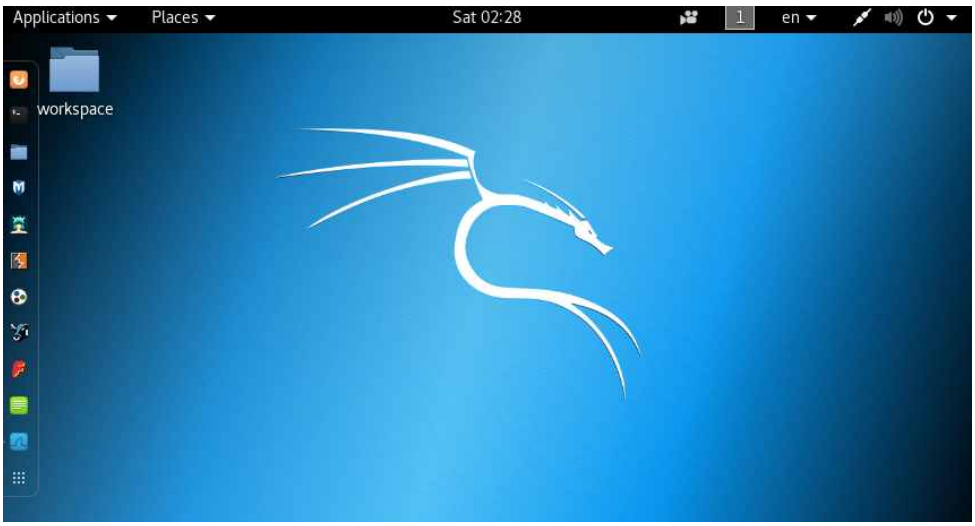
2-10. CentOS

CentOS는 CentOS프로젝트에서 레드햇 제휴로 개발한 컴퓨터 운영 체제이다. 업스트림 소스인 레드햇 엔터프라이즈 리눅스와 완벽하게 호환되는 무료 기업용 컴퓨팅 플랫폼을 제공할 목적으로 만들어진 리눅스계 운영 체제 가운데 하나다. 6.4 버전부터 베타 버전은 파워PC에서 사용 가능할 것으로 예상되지만, 공식적으로 물리 주소 확장 기능을 가진 x86과 x86-64 아키텍처를 지원한다. 사용하는 꾸러미 형식은 RPM이다.

< 그림 2-6 : CentOS >

2-11. Kali linux

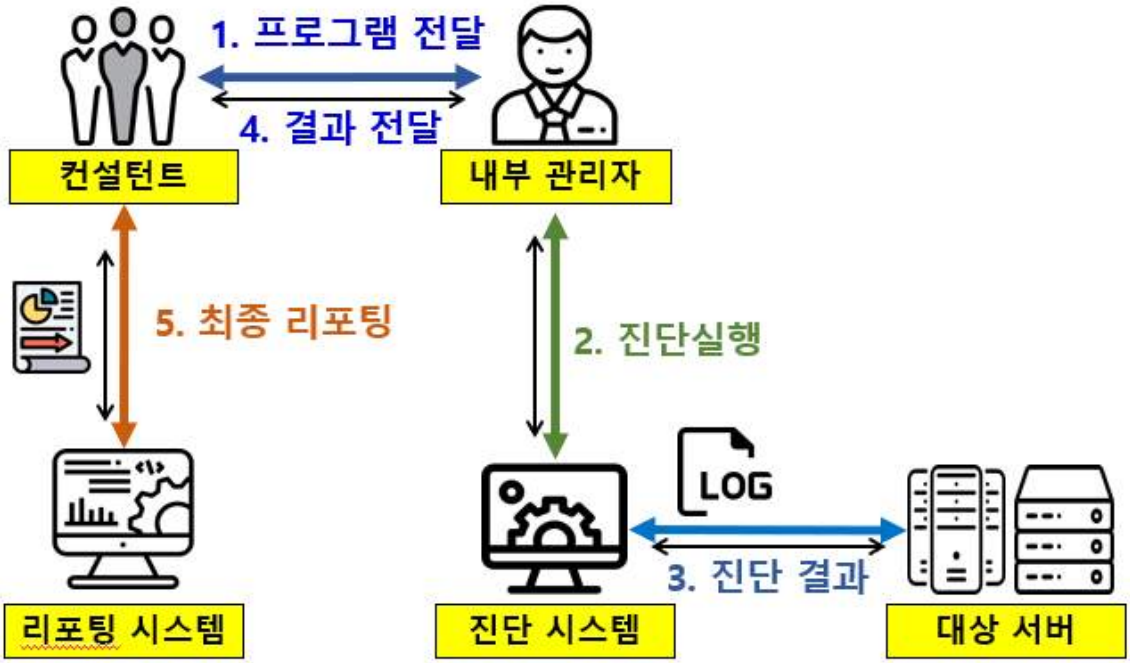
Kali linux는 BackTrack의 후속버전으로 BackTrack이 운영체제의 완성도는 타 배포판의 베타정도로 떨어졌었지만 Kali에서 끌어올렸다. 백트랙과 마찬가지로 모의 침투 테스트에 필요한 툴들을 모아두었다. 여기에는 해킹을 하는데 필요한 툴 외에도, 발견한 취약점을 의뢰인에게 보고하기 위한 레포팅 툴 등도 포함된다. 나머지는 BackTrack과 거의 동일하지만 Kali Linux 만의 특징이라고 한다면 BackTrack에서 동작하지 않거나 중복되는 것들을 제거하고 300개 이상의 침투테스트툴들을 잡아넣었으며 백트랙에서 상당히 미약했던 무선장치 지원을 대폭 강화하였다. 또한 설치하지 않더라도 Live CD나 Live USB와 같은 형태로 PC 등에 설치하지 않고도 부팅할 수 있어 부팅 가능한 Kali Linux 이미지가 들어 있는 DVD나 USB 메모리를 가지고 다니면서 침투 테스트를 진행할 수도 있다.



< 그림 2-7 : Kali linux >

3. 본론

3-1. 구상도 설명



< 그림 3-1 : 구상도 >

첫번째로 점검자가 해당 관리자에게 점검 대상에 맞는 Server Script을 전달하게 되면 관리자가 스크립트 실행 후 나온 출력된 결과 값을 다시 점검자에게 전달한다. 점검자는 리포팅 자동화 도구를 실행하여 엑셀 형식 변환되어 보고서를 출력하고, 해당 보고서를 토대로 점검자가 관리자와 컨설팅을 진행한다.

3-2. 운영체제 스크립트 작성 및 실행

Unix 서버 취약점 분석·평가 항목

분류	점검항목	항목 중요도	항목코드
1. 계정관리	root 계정 원격 접속 제한	상	U-01
	패스워드 복잡성 설정	상	U-02
	계정 잠금 임계값 설정	상	U-03
	패스워드 파일 보호	상	U-04
	root 이외의 UID가 '0' 금지	중	U-05
	root 계정 su 제한	하	U-06
	패스워드 최소 길이 설정	중	U-07
	패스워드 최대 사용 기간 설정	중	U-08
	패스워드 최소 사용기간 설정	중	U-09
	불필요한 계정 제거	하	U-10
	관리자 그룹에 최소한의 계정 포함	하	U-11
	계정이 존재하지 않는 GID 금지	하	U-12
	동일한 UID 금지	중	U-13
	사용자 shell 점검	하	U-14
	Session Timeout 설정	하	U-15
2. 파일 및 디렉터리 관리	root 홈, 패스 디렉터리 권한 및 패스 설정	상	U-16
	파일 및 디렉터리 소유자 설정	상	U-17
	/etc/passwd 파일 소유자 및 권한 설정	상	U-18
	/etc/shadow 파일 소유자 및 권한 설정	상	U-19
	/etc/hosts 파일 소유자 및 권한 설정	상	U-20
	/etc/(x)inetd.conf 파일 소유자 및 권한 설정	상	U-21
	/etc/syslog.conf 파일 소유자 및 권한 설정	상	U-22
	/etc/services 파일 소유자 및 권한 설정	상	U-23
	SUID,SGID,Stick bit 설정 파일 점검	상	U-24
	사용자, 시스템 시작파일 및 환경파일 소유자 및 권한 설정	상	U-25
	world writable 파일 점검	상	U-26
	/dev에 존재하지 않는 device 파일 점검	상	U-27
	$HOME/.rhosts, hosts.equiv 사용 금지	상	U-28
	접속 IP 및 포트 제한	상	U-29
	hosts.lpd 파일 소유자 및 권한 설정	하	U-30
	NIS 서비스 비활성화	중	U-31
	UMASK 설정 관리	중	U-32
	홈디렉토리 소유자 및 권한 설정	중	U-33
	홈디렉토리로 지정한 디렉토리의 존재 관리	중	U-34
	숨겨진 파일 및 디렉토리 검색 및 제거	하	U-35
3. 서비스 관리	finger 서비스 비활성화	상	U-36
	Anonymous FTP 비활성화	상	U-37
	r 계열 서비스 비활성화	상	U-38
	cron 파일 소유자 및 권한설정	상	U-39
	Dos 공격에 취약한 서비스 비활성화	상	U-40
	NFS 서비스 비활성화	상	U-41
	NFS 접근 통제	상	U-42
	automountd 제거	상	U-43
	RPC 서비스 확인	상	U-44
	NIS , NIS+ 점검	상	U-45
	tftp, talk 서비스 비활성화	상	U-46
	Sendmail 버전 점검	상	U-47
	스팸 메일 릴레이 제한	상	U-48
	일반사용자의 Sendmail 실행 방지	상	U-49
	DNS 보안 버전 패치	상	U-50
	DNS Zone Transfer 설정	상	U-51
	Apache 디렉토리 리스팅 제거	상	U-52
	Apache 웹 프로세스 권한 제한	상	U-53
	Apache 상위 디렉토리 접근 금지	상	U-54
	Apache 불필요한 파일 제거	상	U-55
	Apache 링크 사용 금지	상	U-56
	Apache 파일 업로드 및 다운로드 제한	상	U-57
	Apache 웹 서비스 영역의 분리	상	U-58
	ssh 원격접속 허용	중	U-59
	ftp 서비스 확인	하	U-60
	ftp 계정 shell 제한	중	U-61
	Ftpusers 파일 소유자 및 권한 설정	하	U-62
	Ftpusers 파일 설정	중	U-63
	at 파일 소유자 및 권한 설정	중	U-64
	SNMP 서비스 구동 점검	중	U-65
	SNMP 서비스 커뮤니티스트링의 복잡성 설정	중	U-66
	로그온 시 경고 메시지 제공	하	U-67
	NFS설정파일접근권한	중	U-68
	expn, vrfy 명령어 제한	중	U-69
	Apache 웹서비스 정보 숨김	중	U-70
4. 패치 관리	최신 보안패치 및 벤더 권고사항 적용	상	U-71
5. 로그 관리	로그의 정기적 검토 및 보고	상	U-72
	정책에 따른 시스템 로깅 설정	하	U-73

< 그림 3-2 : Unix 서버 취약점 분석·평가 항목 >

KISA에서 발간한 주요정보통신기반시설 기술적 취약점 분석·평가 방법 상세가이드를 토대로 해당 스크립트 작성하였다.

```
#!/bin/sh


# Default 계정 삭제

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"


echo "[U-01]" >> ${RESULT}
echo "Default 계정 삭제" >> ${RESULT}
echo "[점검 현황]" >> ${RESULT}
cat /etc/passwd >> ${RESULT}
echo "" >> ${RESULT}

### 상태
echo "[상태]" >> ${RESULT}

default="adm|lp|sync|shutdown|halt|news|uucp|operator|games|gopher|nfsnobody|squid"

if [ -z "`cat /etc/passwd | egrep ${default}`" ]; then >> ${RESULT}
        echo "양호" >> ${RESULT}
else
        echo "취약" >> ${RESULT}
fi

#### 점검 기준
echo "[점검 기준]" >> ${RESULT} 2>&1
echo "[양호]" >> ${RESULT} 2>&1
echo "- 불필요한 계정이 존재하지 않는 경우" >> ${RESULT} 2>&1
echo "[취약]" >> ${RESULT} 2>&1
echo "- 불필요한 계정이 존재하는 경우" >> ${RESULT} 2>&1
```

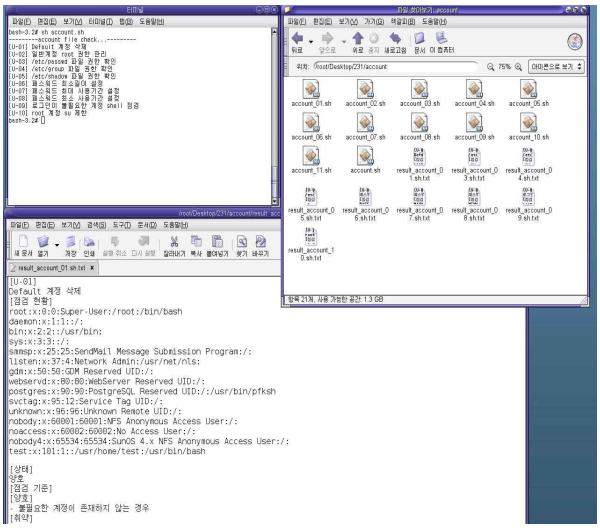
< 그림 3-3 : Unix Account script의 일부 >

```
#!/bin/sh

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"

echo "---------account file check...----------"

echo "[U-01] Default 계정 삭제"
./account/account_01.sh
echo "[U-02] 일반계정 root 권한 관리"
##./account/account_02.sh
echo "[U-03] /etc/passwd 파일 권한 확인"
./account/account_03.sh
echo "[U-04] /etc/group 파일 권한 확인"
./account/account_04.sh
echo "[U-05] /etc/shadow 파일 권한 확인"
./account/account_05.sh
echo "[U-06] 패스워드 최소길이 설정"
./account/account_06.sh
echo "[U-07] 패스워드 최대 사용기간 설정"
./account/account_07.sh
echo "[U-08] 패스워드 최소 사용기간 설정"
./account/account_08.sh
echo "[U-09] 로그인이 불필요한 계정 shell 점검"
./account/account_09.sh
echo "[U-10] root 계정 su 제한"
./account/account_10.sh
```

메인 스크립트를 작성하여 스크립트를 실행한다.

< 그림 3-4 : Unix main script의 일부 >

스크립트 작성 후 모든 스크립트들을 실행하기 위한 메인 스크립트 작성 후 실행한다.



< 그림 3-5 : Unix Script 실행 결과 >

실행 후 해당 출력 결과물이 .txt 형식으로 저장되는 것을 볼 수 있다.

3-3. 엑셀파일 형식으로 전환

출력된 결과 값이 프로그램을 통하여 엑셀 파일에 해당 형식으로 저장된다. 한눈에 알아보기 쉽게 진단결과 값이 출력되는 것을 볼 수 있다.

.

no.	점검항목	점검현황	상태	양호	취약
[LCv7-1.02]	일반계정 root권한 관리 (상)	1. UID가 0인 계정 목록 root:x:0:0:root:/root:/bin/bash	양호	root 계정과 동일한 UID를 갖는 계정이 존재하지 않는 경우	root 계정과 동일한 UID를 갖는 계정이 존재하는 경우
[LCv7-1.03]	/etc/passwd 파일 권한 확인 (상)	1. /etc/passwd 파일 소유자와 권한 -rw-r--r--. 1 root root 2788 9월 15 17:09 /etc/passwd	양호	/etc/passwd 파일의 소유자가 root이고, 권한이 644 이하인 경우	/etc/passwd 파일의 소유자가 root가 아니거나, 권한이 644 이하가 아닌 경우
[LCv7-1.04]	/etc/group 파일 권한 확인 (상)	1. /etc/group 파일 소유자와 권한 -rw-r--r--. 1 root root 1086 9월 15 17:09 /etc/group	양호	/etc/group 파일의 소유자가 root이고, 권한이 644 이하인 경우	/etc/group 파일의 소유자가 root가 아니거나, 권한이 644 이하가 아닌 경우
[LCv7-1.05]	/etc/shadow 파일 권한 확인 (상)	1. /etc/shadow 파일 소유자와 권한 -rw-rw-r--. 1 root root 1527 9월 16 20:19 /etc/shadow	취약	/etc/shadow 파일의 소유자가 root이고, 권한이 400인 경우	/etc/shadow 파일의 소유자가 root가 아니거나, 권한이 400이 아닌 경우
[LCv7-1.06]	패스워드 사용 규칙 적용 (상)	1. /etc/login.defs파일의 패스워드 최소 길이 설정 PASS_MIN_LEN 5 2. /etc/login.defs파일의 패스워드 최대 사용 기간 설정 PASS_MAX_DAYS 99999 3. /etc/login.defs파일의 패스워드 최소 사용 기간 설정 PASS_MIN_DAYS 0 4. /etc/pam.d/system-auth파일의 계정 잠금 임계 값 설정	취약	1. 패스워드 최소 길이	패스워드 최소 길이가 8자 이상으로 설정되어 있는 경우
[LCv7-1.07]	로그인이 불필요한 계정 shell 제한 (중)	1. /bin/false(nologin) shell이 부여되지 않은 계정	양호	로그인이 필요하지 않은 계정에 /bin/false(nologin) 쉘이 부여되어 있는 경우	로그인이 필요하지 않은 계정에 /bin/false(nologin) 쉘이 부여되지 않은 경우
[LCv7-1.08]	SU(Select User) 사용 제한 (상)	1. wheel그룹(su 명령어 사용 그룹) 및 그룹 내 구성원 존재 여부 wheel:x:10:hyebeen 2. /bin/su 권한과 소유 그룹 확인 -rwsr-xr-x. 1 root root 32184 8월 17 03:47 /bin/su 3. /usr/bin/su 권한과 소유 그룹 확인 -rwsr-xr-x. 1 root root 32184 8월 17 03:47 /usr/bin/su 4. 허용 그룹(su 명령어 사용 그룹) 설정 여부 확인 #auth sufficient pam_wheel.so trust use_uid #auth required pam_wheel.so use_uid	취약	su 명령어를 특정 그룹에 속한 사용자만 사용하도록 제한되어 있는 경우	su 명령어를 모든 사용자가 사용하도록 설정되어 있는 경우
[LCv7-5.01]	Anonymous FTP 비활성화 (상)	1. Default FTP를 사용하는 경우 1-1. ftp또는 anonymous 계정의 존재 여부 anonymous:x:51:51::/var/spool/mqueue/sbin/nologin ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin 2. ProFTP 를 사용하는 경우 2-1. ftp또는 anonymous 계정의 존재 여부 ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin 3. vsFTP를 사용하는 경우 3-1. /etc/vsftpd/vsftpd.conf 파일 설정 anonymous_enable=NO	취약	Anonymous FTP (익명 ftp) 접속을 차단한 경우	Anonymous FTP (익명 ftp) 접속을 차단하지 않은 경우
[LCv7-5.02]	SNMP(Simple Network Management Protoc	1. SNMP Community String 설정 확인 com2sec notConfigUser default public	취약	SNMP Community 이름이 public, private 이 아닌 경우	SNMP Community 이름이 public, private 인 경우

< 그림 3-6 : Reporting System 실행 결과 >

4. 결론

4-1. 결론

사이버위협의 증가와 ISMS인증의 의무화에 따라 효율적 관리의 필요성과 신속한 점검 및 대응이 필요하기에 취약점을 진단해주는 스크립트를 만들었다. 또한 각각의 스크립트를 확인하는 시간을 줄이고 어려움을 해소 시키기 위해 보고서 자동화 도구를 구현하였다.

스크립트의 모듈화로 취약점 점검 내용 변경이나 스크립트의 수정사항이 있을 경우 스크립트 파일에 대해 빠르고 간편하게 수정을 하게 되어 유지보수가 용이해 질것으로 보이며, 주요통신기반시설 취약점 가이드를 토대로 만든 스크립트로 모든 리스트 점검이 가능하다. 또한 점검 후 산출되어진 각각의 보고서 파일들을 자동으로 합친 후 엑셀 형식으로 변환하기 때문에 실무자들은 큰

어려움 없이 간단하게 조작 할 수 있어 편의성이 증가 되는 효과를 볼 수 있다. 마지막으로 엑셀화된 보고서만 받아 볼 수 있어서 과거에 비해 정확하고 간단하게 점검 항목을 확인 할 수 있고 취약점에 대해 신속하고 정확한 대응이 가능 할 것으로 기대 되어진다.

5. 첨부

5-1-1. Reporting System

<Reporting System>

```
# -*- coding: utf-8 -*-
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import *
from PyQt5.QtCore import *

import sys
import codecs
import os
from openpyxl import Workbook
from openpyxl.styles import PatternFill,Color
from openpyxl.styles import Border,Side
from openpyxl.styles import Font, Alignment

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(786, 529)
        Form.setStyleSheet("QWidget {₩n"
        "    background-color: #222222;₩n"
        "}₩n"
        "₩n"
        "QPushButton {₩n"
        "    background-color: #8b0000;₩n"
        "    color: #ffffff;₩n"
        "    height: 25px;₩n"
```

```
"}₩n"
"₩n"
"QCheckBox {₩n"
"    background-color: #222222;₩n"
"    color: #FF5E00;₩n"
"}₩n"
"₩n"
"QTextBrowser {₩n"
"    background-color: #ffffffff;₩n"
"    color: #222222;₩n"
"}₩n"
"")

self.centralwidget = QtWidgets.QWidget(Form)
self.centralwidget.setObjectName("centralwidget")
self.ing_list = QtWidgets.QTextBrowser(self.centralwidget)
self.ing_list.setGeometry(QtCore.QRect(20, 20, 331, 461))
self.ing_list.setObjectName("ing_list")
self.window_check = QtWidgets.QCheckBox(self.centralwidget)
self.window_check.setGeometry(QtCore.QRect(390, 320, 96, 19))
self.window_check.setObjectName("window_check")
self.linux_check = QtWidgets.QCheckBox(self.centralwidget)
self.linux_check.setGeometry(QtCore.QRect(500, 320, 96, 19))
self.linux_check.setObjectName("linux_check")
self.unix_check = QtWidgets.QCheckBox(self.centralwidget)
self.unix_check.setGeometry(QtCore.QRect(390, 370, 96, 19))
self.unix_check.setObjectName("unix_check")
self.all_check = QtWidgets.QCheckBox(self.centralwidget)
self.all_check.setGeometry(QtCore.QRect(500, 370, 96, 19))
self.all_check.setObjectName("all_check")
self.start_button = QtWidgets.QPushButton(self.centralwidget)
self.start_button.setGeometry(QtCore.QRect(380, 440, 161, 41))
self.start_button.setObjectName("start_button")
self.cancle_button = QtWidgets.QPushButton(self.centralwidget)
self.cancle_button.setGeometry(QtCore.QRect(600, 440, 161, 41))
self.cancle_button.setObjectName("cancle_button")
self.file_view = QtWidgets.QTextBrowser(self.centralwidget)
self.file_view.setGeometry(QtCore.QRect(390, 20, 381, 261))
```

```
        self.file_view.setObjectName("file_view")
        self.File_search_button = QtWidgets.QPushButton(self.centralwidget)
        self.File_search_button.setGeometry(QtCore.QRect(600, 320, 161, 41))
        self.File_search_button.setObjectName("File_search_button")
        self.File_start_button = QtWidgets.QPushButton(Form)
        self.File_start_button.setGeometry(QtCore.QRect(600, 380, 161, 41))
        self.File_start_button.setObjectName("pushButton_2")
        self.menubar = QtWidgets.QMenuBar(Form)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 786, 26))
        self.menubar.setObjectName("menubar")
        self.statusbar = QtWidgets.QStatusBar(Form)
        self.statusbar.setObjectName("statusbar")

        self.file_name_save = [] #This file_search save file name

        #excell common from
        self.box = Border(left=Side(border_style="thin",color='FF000000'),
                          right=Side(border_style="thin",color='FF000000'),
                          top=Side(border_style="thin", color='FF000000'),
                          bottom=Side(border_style="thin", color='FF000000'),
                          diagonal=Side(border_style="thin", color='FF000000'),

                          diagonal_direction=0,

                          outline=Side(border_style="thin", color='FF000000'),
                          vertical=Side(border_style="thin",color='FF000000'),
                          horizontal=Side(border_style="thin", color='FF000000')
                    )


        self.excell_test = ['no.' , '점검항목', '점검현황', '상태', '양호', '취약']
        self.excell_merge = ['1', '2', '4', '5', '6' ]

        self.file_view_result = [] # 파일내용 저장



        self.cell_row = 1
        self.cell_line = 1
```

```
        self.save_len = self.cell_row

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

        self.start_button.clicked.connect(self.start)
        self.cancle_button.clicked.connect(self.cancle)
        self.File_search_button.clicked.connect(self.file_search)
        self.File_start_button.clicked.connect(self.file_search_start)



        self.all_check.stateChanged.connect(self.allcheck)
        self.window_check.stateChanged.connect(self.checkbox)
        self.linux_check.stateChanged.connect(self.checkbox)
        self.unix_check.stateChanged.connect(self.checkbox)

    def fixing(self,wb,ws): #윗단락 고정
        self.cell_row = 1
        self.cell_line = 1

        for i in range(0, len(self.excell_test)):
            ws.cell(row=self.cell_row, column = self.cell_line).value = self.excell_test[i]
            ws.cell(row=self.cell_row, column = self.cell_line).fill = PatternFill(patternType='solid',fgColor=Color('FFF612'))
            ws.cell(row=self.cell_row, column = self.cell_line).border = self.box
            self.cell_line += 1

        return wb, ws


    def excell_start(self,wb,ws,stack_cell_row,stack_arry): #엑셀 생성 부분

        for j in range(0,5):

            if self.file_view_result[stack_arry] == '[점검 현황]':
                stack_cell_row = self.cell_row
                stack_arry += 1
```

```
                    while True:
                        if self.file_view_result[stack_arry] == "[상태]":
                            self.save_len = self.cell_row
                            self.cell_row = stack_cell_row
                            self.cell_line += 1
                            stack_arry += 1
                            break

                        ws.cell(row = self.cell_row, column = self.cell_line).value = self.file_view_result[stack_arry]
                        ws.cell(row= self.cell_row, column = self.cell_line).border = self.box
                        self.cell_row += 1
                        stack_arry += 1
                if self.file_view_result[stack_arry] == "[점검 기준]":
                    stack_arry += 2

                if self.file_view_result[stack_arry] == "[취약]":
                    stack_arry += 1

                ws.cell(row = self.cell_row, column = self.cell_line).value = self.file_view_result[stack_arry]
                if self.cell_line == 6:
                    stack_arry += 1
                    break
                else :
                    self.cell_line += 1
                    stack_arry += 1

            for i in range(0, len(self.excell_merge)):
                ws.merge_cells(start_row = self.cell_row, start_column=int(self.excell_merge[i]), end_row = self.save_len-1, end_column= int(self.excell_merge[i]))

            for i in range(self.cell_row, self.save_len):
                for j in range(0, len(self.excell_merge)):
                    ws.cell(row= i , column = int(self.excell_merge[j])).border = self.box
                    ws.cell(row= i , column = int(self.excell_merge[j])).alignment = Alignment(vertical='center')
```

```
        ws.freeze_panes = 'A2'
        return wb,ws,stack_cell_row,stack_arry


    def linux_unix(self,wb,ws,stack_cell_row,stack_arry,file_name, file_dir_num):
        #파일 읽어서 이제 시작하는 부분
        view_list_name = []

        linux_report_name= file_name%(str(file_dir_num))

        view_list_name.append(linux_report_name.split('/'))
        view_len = len(view_list_name[0])

        self.ing_list.insertPlainText(view_list_name[0][view_len-1]+"₩n")
        report = codecs.open(linux_report_name, "r", "utf-8")

        while True:
            linux_report = report.readline()
            self.file_view_result.append(linux_report[:-1].strip())

            if not linux_report:
                break
        report.close()

        self.excell_start(wb,ws,stack_cell_row,stack_arry) #엑셀 내용 집어 넣는 부분

        self.cell_row = self.save_len
        self.cell_line = 1
        stack_arry = 0
        self.file_view_result = []

        return wb,ws,stack_cell_row,stack_arry


    def file_search(self):
        fname = QFileDialog.getOpenFileName()
        self.file_view.insertPlainText(fname[0]+"₩n")
        self.file_name_save.append(fname[0])
```

```
    def file_search_start(self):

        wb = Workbook()
        ws = wb.active

        self.fixing(wb,ws)

        wb.save('file_search.xlsx')
        wb.close()



    def allcheck(self):
        if self.all_check.isChecked()==True:
            self.window_check.setChecked(True)
            self.linux_check.setChecked(True)
            self.unix_check.setChecked(True)

    def checkbox(self):

        if self.window_check.isChecked() == False or self.linux_check.isChecked() == False or
self.unix_check.isChecked() == False:
            self.all_check.setChecked(False)

        if self.window_check.isChecked() == True and self.linux_check.isChecked() == True and
self.unix_check.isChecked() == True:
            self.all_check.setChecked(True)

    def start(self):
        if self.window_check.isChecked()==True:
            self.window()
        else:
            pass

        if self.linux_check.isChecked()==True:
            self.linux()
```

```
        else:
            pass

        if self.unix_check.isChecked()==True:
            self.unix()
        else:
            pass


    def cancle(self):
        sys.exit()


    def window(self): #윈도우임
        self.file_view_result = []

        wb = Workbook()
        ws = wb.active

        self.fixing(wb,ws) #윗단락 고정

        self.cell_line = 1
        self.cell_row += 1

        stack_cell_row = 0
        self.save_len = self.cell_row
        stack_arry = 0

        window_report_name = "C:/Users/moonho/Desktop/졸작/레포트/window/%s"

        file_dir_num = os.listdir('C:/Users/moonho/Desktop/졸작/레포트/window')
        file_dir_len = len(file_dir_num)

        for ran in range(0,file_dir_len):
            #파일 읽어서 이제 시작하는 부분
            view_list_name = []

            window_report_result = window_report_name%(str(file_dir_num[ran]))
```

```
            print(window_report_result)

            view_list_name.append(window_report_result.split('/'))
            view_len = len(view_list_name[0])
            self.ing_list.insertPlainText(view_list_name[0][view_len-1]+"₩n")
            report = codecs.open(window_report_result, "r", "utf-8")


            while True:
                window_report = report.readline()
                self.file_view_result.append(window_report[:-1].strip())

                if not window_report:
                    break
            report.close()

            self.excell_start(wb,ws,stack_cell_row,stack_arry) #엑셀 내용 집어 넣는 부분

            self.cell_row = self.save_len
            self.cell_line = 1
            stack_arry = 0
            self.file_view_result = []

        wb.save('window.xlsx')
        wb.close()

    def linux(self):
        self.file_view_result = []

        wb = Workbook()
        ws = wb.active

        self.fixing(wb,ws) #윗단락 고정

        self.cell_line = 1
        self.cell_row += 1

        stack_cell_row = 0
```

```
        self.save_len = self.cell_row
        stack_arry = 0
        file_dir_num = os.listdir('C:/Users/moonho/Desktop/졸작/레포트/linux')
        file_dir_len = len(file_dir_num)

        for file_num in range(1,file_dir_len):
            file_name = "C:/Users/moonho/Desktop/졸작/레포트/linux/%s"
            self.linux_unix(wb, ws, stack_cell_row, stack_arry, file_name, file_dir_num[file_num])

        wb.save('linux.xlsx')
        wb.close()


    def unix(self):
        self.file_view_result = []

        wb = Workbook()
        ws = wb.active

        self.fixing(wb,ws) #윗단락 고정

        self.cell_line = 1
        self.cell_row += 1

        stack_cell_row = 0
        self.save_len = self.cell_row
        stack_arry = 0
        file_dir_num = os.listdir('C:/Users/moonho/Desktop/졸작/레포트/linux')
        file_dir_len = len(file_dir_num)

        for file_num in range(1,file_dir_len):
            file_name = "C:/Users/moonho/Desktop/졸작/레포트/linux/%s"
            self.linux_unix(wb, ws, stack_cell_row, stack_arry, file_name, file_dir_num[file_num])

        wb.save('unix.xlsx')
        wb.close()

    def retranslateUi(self, Form):
```

```
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.window_check.setText(_translate("Form", "윈도우"))
        self.linux_check.setText(_translate("Form", "리눅스"))
        self.unix_check.setText(_translate("Form", "유닉스"))
        self.all_check.setText(_translate("Form", "ALL"))
        self.start_button.setText(_translate("Form", "Start"))
        self.cancle_button.setText(_translate("Form", "Cancle"))
        self.File_search_button.setText(_translate("Form", "File_search"))
        self.File_start_button.setText(_translate("Form", "File_Start"))


if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec_())
```

5-1-2. Windows

<Windows Server GUI>

```
Gui, New
GUI,Submit,nohide
Gui, color, 000000
Gui, Font, bold
Gui, Font, s15
Gui, Add, GroupBox, x15 y15 w420 h280 CFF0000, Vulnerability Check System

Gui, Font, s13
Gui, Add, GroupBox, x80 y55 w303 h155 CFFFFFF, Choose Option



btnHelp = x95 y85 w130 h60 cwhite
Gui, Font, bold
Gui, Font, S11
```

```
Gui, Add, Button, %btnHelp%, Inspect
Gui, Font, S11
Gui, Add, Button, x238 y85 w130 h60, Result
Gui, Add, Button, x95 y155 w90 h40, Readme
Gui, Add, Button, x187 y155 w90 h40, Info
Gui, Add, Button, x279 y155 w90 h40, Exit

Gui, Add, Picture, x25 y252 w110 h40, ./logo.bmp

Gui, Font, s13
Gui, Add, Text, x330 y275 w100 h15 CEEEEEE, Team N2TG
Gui, Show, w1 y1 w450 h310, Vulnerability Check System_v1.0
Gui, +ToolWindow
return


ButtonInspect:
GUI, New
GUI, Submit, Nohide
Gui, Color, 000000
Gui, Font, bold
Gui, Add, GroupBox, x5 y5 w110 h290 cFF0000 , Setting
Gui, Font, bold
Gui, Add, GroupBox, x5 y20 w110 h275 cFF0000, Version
Gui, Add, CheckBox, x10 y40 w100 h20 vh1 cEEEEEE, windows2008
Gui, Add, CheckBox, x10 y66 w100 h20 vh2 cEEEEEE, windows2010
Gui, Add, CheckBox, x10 y92 w100 h20 vh3 cEEEEEE, windows2012

Gui, Font, bold
Gui, Add, GroupBox, x5 y122 w110 h173 cFF0000, Selection
Gui, Add, CheckBox, x10 y144 w100 h20 vh4 cEEEEEE, ACCOUNT
Gui, Add, CheckBox, x10 y170 w100 h20 vh5 cEEEEEE, SERVICE
Gui, Add, CheckBox, x10 y196 w100 h20 vh6 cEEEEEE, PATCH
Gui, Add, CheckBox, x10 y222 w100 h20 vh7 cEEEEEE, LOGGING
Gui, Add, CheckBox, x10 y248 w100 h20 vh8 cEEEEEE, SECURITY



Gui, Add, Button, x18 y270 w88 h20 gstart, Inspect start
```

```
Gui, Add, Text, x105 y388 w55 h15 CEEEEEE, Team N2TG
Gui, Show, w1 y1 w135 h300, Vulnerability Check System
return


ButtonInfo:
Gui, New
Gui, submit, Nohide
Gui, Color, 000000
Gui, Font, bold
Gui, Font, S11
Gui, Add, Text, x10 y10 w180 h20 CEEEEEE, [Profile]
Gui, Add, Text, x20 y32 w140 h20 CEEEEEE, -Team : N2TG
Gui, Add, Text, x20 y51 w300 h20 CEEEEEE, -Name : Hyun Su Son, Seung Ha Yoo
Gui, Add, Text, x20 y70 w320 h20 CEEEEEE, -Department : Information Security
Gui, Add, Text, x20 y89 w250 h20 CEEEEEE, -E-mail : gustn4203@naver.com
Gui, Add, Text, x95 y108 w160 h20 CEEEEEE,                seungha06@nate.com

Gui, Show, w1 y1 w350 h130, Team.N2TG
return


start:
GUI, Submit, Nohide
if(h1 = 1){
if(h4=1 && h5=1 && h6=1 && h7=1 && h8=1){
run total.bat, ₩w08
}
else{

if(h4=1){
run 1_account, .₩w08
}

if(h5=1){
run 2_service.bat, .₩w08
}
```

```
if(h6=1){
run 3_patch.bat, .₩w08
}

if(h7=1){
run 4_log.bat, .₩w08
}

if(h8=1){
run 5_security.bat, .₩w08
}

}
}

if(h2 = 1){
if(h4=1 && h5=1 && h6=1 && h7=1 && h8=1){
run total.bat, .₩w10
}

else{
if(h4=1){
run 1_account.bat, .₩w10
}

else{
if(h5=1){
run 2_service.bat, .₩w10
}

if(h6=1){
run 3_patch.bat, .₩w10
}

if(h7=1){
run 4_log.bat, .₩w10
}
```

```
if(h8=1){
run 5_security.bat, .₩w10
}

}
}
}

if(h3 = 1){
if(h4=1 && h5=1 && h6=1 && h7=1 && h8=1){
run total.bat, .₩w12
}

else{
if(h4=1){
run 1_account.bat, .₩w12
}

if(h5=1){
run 2_service.bat, .₩w12
}

if(h6=1){
run 3_patch.bat, .₩w12
}

if(h7=1){
run 4_log.bat, .₩w12
}

if(h8=1){
run 5_security.bat, .₩w12
}

}
}

msgbox,,t,clear
```

```
return



ButtonResult:
GUI, Submit, Nohide
run ,.₩
return

ButtonReadme:
run , .₩readme.txt
GUI, Submit, Nohide

ButtonExit:
Gui, Submit, Nohide
exitapp
return
```

<Windows Server 2012 batch - Account>

```
:: W12-01 administrator 계정 점검

@echo off

set ROOT_PATH=%CD%

set CURRENT_PATH=%~dp0

set FILENAME=result_%~n0.txt

set ETC_PATH=%CURRENT_PATH%..₩etc

set RESULT="%CURRENT_PATH%%FILENAME%"

echo [W12-01]>>%RESULT%
echo Administrator 계정 확인 (상)>> %RESULT%



echo [점검 현황]>>%RESULT%
```

```
echo ## administrator 계정 정보 조회 ## >> %RESULT%
        net user | findstr /V "account ---- command" >> %RESULT%
        net user | findstr /V "account ---- command" | findstr /I "administrator"

        echo [상태]>>%RESULT%
                If %errorlevel% == 0 echo 취약>>%RESULT%
                If Not %errorlevel% == 0 echo 양호>> %RESULT%
echo [점검 기준]>>%RESULT%

echo [양호]>>%RESULT%
echo 윈도우의 기본 관리자 계정명 administrator 존재하지 않음 >> %RESULT%

echo [취약]>>%RESULT%
echo 윈도위의 기본 관리자 계정명 administrator 존재함 >> %RESULT%
```

<Windows Server 2012 batch - Log>

```
:: W12-57 로그의 정기적 검토 및 보고
@echo off

set ROOT_PATH=%CD%

set CURRENT_PATH=%~dp0

set FILENAME=result_%~n0.txt

set ETC_PATH=%CURRENT_PATH%..₩etc

set RESULT="%CURRENT_PATH%%FILENAME%"

echo [W12-57]>>%RESULT%
echo 로그의 정기적 검토 및 보고(상)>> %RESULT%
::############################ ETC 파일 체크 ########################
set RESOURCE="%ETC_PATH%₩command_secedit_export_cfg.txt"

if not exist %RESOURCE% (
        secedit /export /cfg "%RESOURCE%"
)
```

```
::#################################################################

echo [점검 현황]>>%RESULT%
echo ## 보안설정 감사정책 ## >> %RESULT%
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process"   >>
%RESULT%
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:"AuditLogonEvents = 0"
if %errorlevel% == 0 (goto :result)
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:" AuditPrivilegeUse = 0"
if %errorlevel% == 0 (goto :result)
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:"AuditPolicyChange = 0"
if %errorlevel% == 0 (goto :result)
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:"AccountManage = 0"
if %errorlevel% == 0 (goto :result)
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:"AuditDSAccess = 0"
if %errorlevel% == 0 (goto :result)
type %RESOURCE% | findstr /I /C:"audit" | findstr /I /V "machine ea system object process" |
findstr /I /C:"AuditAccountLogon = 0"
if %errorlevel% == 0 (goto :result)

:result
echo [상태]>>%RESULT%
        if %errorlevel% == 0 echo 취약>>%RESULT%
        if not %errorlevel% == 0 echo 양호>> %RESULT%
echo [점검 기준]>>%RESULT%

echo [양호]>>%RESULT%
echo 로그온 이벤트, 계정 로그온 이벤트, 정책 변경: 성공/실패 감사(3)>> %RESULT%
echo 계정 관리, 디렉터리 서비스 액세스, 권한 사용: 실패 감사(2)>> %RESULT%

echo [취약]>>%RESULT%
echo  이벤트에 대한 감사 설정이 되어 있지 않은 경우(0)>> %RESULT%
```

<Windows Server 2012 batch - Patch>

```
:: W12-55   최신 HOT FIX가 적용
@echo off

set ROOT_PATH=%CD%

set CURRENT_PATH=%~dp0

set FILENAME=result_%~n0.txt

set ETC_PATH=%CURRENT_PATH%..₩etc

set RESULT="%CURRENT_PATH%%FILENAME%"

echo [W12-55]>>%RESULT%
echo 최신 HOT FIX 적용 (상)>> %RESULT%
::############################ ETC 파일 체크 ########################
set RESOURCE="%ETC_PATH%₩command_wmic_qfe_get_hotfixid.txt"

if not exist %RESOURCE% (
        wmic qfe get HotFixID,Description,InstalledOn > %RESOURCE%
)
::#####################################################################

echo [점검 현황]>>%RESULT%
echo ## 설치된 hotfix 정보 ## >> %RESULT%
type %RESOURCE% >> %RESULT%
type %RESOURCE% | findstr /C:"1/1"
echo [상태]>>%RESULT%
        If %errorlevel% == 0 echo 취약>>%RESULT%
        If Not %errorlevel% == 0 echo 양호>> %RESULT%

echo [점검 기준]>>%RESULT%

echo [양호]>>%RESULT%
echo 최근 6개월 이내 hotfix 설치이력이 존재함>> %RESULT%

echo [취약]>>%RESULT%
```

```
echo 최근 hotfix 설치 이력이 6개월을 초과함>> %RESULT%
```

<Windows Server 2012 batch - Security>

```
:: W12-W62 백신 프로그램 설치
@echo off

set ROOT_PATH=%CD%

set CURRENT_PATH=%~dp0

set FILENAME=result_%~n0.txt

set ETC_PATH=%CURRENT_PATH%..₩etc

set RESULT="%CURRENT_PATH%%FILENAME%"

echo [W12-62]>>%RESULT%
echo 백신 프로그램 설치(상)>> %RESULT%

::###################### ETC 체크 ############################
set RESOURCE="%ETC_PATH%₩command_net_start.txt"

if not exist %RESOURCE% (
        net start > %RESOURCE%
)
::############################################################

echo [점검 현황]>>%RESULT%
echo ## 백신 프로그램 확인 ##
type %RESOURCE% | findstr /I "virus anti alyac hauri ahn v3 mcafee escan"
if not errorlevel 1 set check="running"
if errorlevel 1 set check="stop"

if !check!=="stop" (
        set check_exe="null"
)
```

```
if !check!=="running" (
        if not exist "net start" (
        echo debug : 체크중
        set check_exe="dddd" >>%RESULT%

        echo debug: !check_exe! )

)
echo debug %check%
echo debug %check_exe%

if %check%=="stop" (
        echo debug: disable
        echo 백신이 존재 하지 않음 >> %RESULT%
        ) else (
        if exist "%ETC_PATH%" (
                type %RESOURCE% | findstr /I "virus anti alyac hauri ahn v3 mcafee escan"
>>%RESULT%
        )
)
type %RESOURCE% |findstr /I "virus anti alyac hauri ahn v3 mcafee escan"

echo [상태]>>%RESULT%
if %errorlevel% == 0 echo "양호" >>%RESULT%
if not %errorlevel% == 0 echo "취약">>%RESULT%
echo [점검 기준]>>%RESULT%

echo [양호]>>%RESULT%
echo 백신이 설치되어 있음 >> %RESULT%

echo [취약]>>%RESULT%
echo 백신이 설치되어 있지 않음 >> %RESULT%
```

<Windows Server 2012 batch - Service>

```
:: W12-19 공유 권한 및 사용자 그룹 설정

@echo off
```

```
set ROOT_PATH=%CD%

set CURRENT_PATH=%~dp0

set FILENAME=result_%~n0.txt

set ETC_PATH=%CURRENT_PATH%..₩etc

set RESULT="%CURRENT_PATH%%FILENAME%"

echo [W12-19]>> %RESULT%
echo 공유 권한 및 사용자 그룹 설정 (상)>> %RESULT%

::################ ETC 파일 체크 ####################
set RESOURCE="%ETC_PATH%₩command_net_share.txt"

if not exist %RESOURCE% (
        net share > %RESOURCE%
)

set TEMP_FILE="%ETC_PATH%₩temp_command_net_share.txt"

type %RESOURCE% | findstr /I "₩ $" > %TEMP_FILE%
::################################################

echo [점검 현황]>>%RESULT%
echo ## 공유폴더 목록 ## >> %RESULT%
type %TEMP_FILE% >> %RESULT%
echo. >> %RESULT%
echo ## 공유폴더 권한 확인 ## >> %RESULT%
        net share test |findstr /V "completed" >> %RESULT%
        net share test |findstr /V "completed" |findstr /I /C:"everyone, full"
echo [상태]>> %RESULT%
        if %errorlevel% == 0 echo "취약" >>%RESULT%
        if not %errorlevel% == 0 echo "양호" >> %RESULT%

echo [점검 기준]>> %RESULT%
```

```
echo [양호]>> %RESULT%
echo 공유폴더에 everyone에 대한 권한이 존재하지 않음 >> %RESULT%

echo [취약]>> %RESULT%
echo 공유폴더에 everyone에 대한 권한이 존재함 >> %RESULT%
```

5-1-3. Linux

<CentOS GUI>

```
#!/bin/sh

ACCOUNT=(1 "계정 관리"  "account" "Account Management" 8)
FILESYSTEM=(2 "파일 시스템" "filesystem" "filesystem" 16)
SERVICE=(3 "네트워크 서비스"  "service" "Network Service" 9)
LOGGING=(4 "로그 관리" "logging" "Log Management" 2)
ETC=(5 "주요 응용 설정" "etc" "application settings" 7)
PATCH=(6 "보안 패치" "patch" "Security Patches" 1)

TAB="₩t"

function bar()
{
        PRINT_BAR=`expr $PRINT_BAR + $PERCENT`
        echo /bin/echo " " | dialog --backtitle "ⓒ 2018.HYEBEEN. all rights reserved." --title
"Vulnerability Check system" --gauge "₩n₩n₩n            Loading..." 15 30 "$PRINT_BAR"
}

function range_num()
{
        NUM=""

        for i in $(seq $1)
        do
                if [ $i -ge 10 ]
                then
                        NUM="$NUM""$i"" "
```

```
                else
                        NUM="$NUM""0""$i"" "
                fi
        done
}

function item_check()
{
        echo        "=============================="        "$1".        ""$2"
"==============================" >> ${RESULT}
        range_num $4

        for FILE_COUNT in $NUM
        do
                ${3}/${3}_${FILE_COUNT}.sh
                cat ${3}/result_${3}_${FILE_COUNT}.txt >> ${RESULT}
                echo "" >> ${RESULT}
                bar $1 $2
                sleep 1
                if [ $FILE_COUNT -eq ${4} ]
                then
                        continue
                fi
                echo    "--------------------------------------------------------------------"    >>
${RESULT}
                echo "" >> ${RESULT}
        done
}

function start_check()
{
        PRINT_BAR=0
        RESULT="$(dirname "$0")/result/result_$(date '+%y-%m-%d').txt"
        rm -r "$RESULT"

        for ITEM in $CHECKED_ITEM
        do
                case $ITEM in
```

```
                1)
                        item_check "${ACCOUNT[0]}" "${ACCOUNT[1]}" "${ACCOUNT[2]}" "${ACCOUNT[4]}"
                        ;;
                2)
                        item_check "${FILESYSTEM[0]}" "${FILESYSTEM[1]}" "${FILESYSTEM[2]}" "${FILESYSTEM[4]}"
                        ;;
                3)
                        item_check "${SERVICE[0]}" "${SERVICE[1]}" "${SERVICE[2]}" "${SERVICE[4]}"
                        ;;
                4)
                        item_check "${LOGGING[0]}" "${LOGGING[1]}" "${LOGGING[2]}" "${LOGGING[4]}"
                        ;;
                5)
                        item_check "${ETC[0]}" "${ETC[1]}" "${ETC[2]}" "${ETC[4]}"
                        ;;
                6)
                        item_check "${PATCH[0]}" "${PATCH[1]}" "${PATCH[2]}" "${PATCH[4]}"
                        ;;
        esac
    done
    dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --msgbox "₩n₩n complete!" 10 30
}

function get_percent()      #PERCENT를 구하기 위한 사전 값 구하기
{
    TOTAL_NUM=0

    for ITEM in $CHECKED_ITEM
    do
        case $ITEM in
                1)
                        TOTAL_NUM=`expr $TOTAL_NUM + ${ACCOUNT[4]}`
```

```
                ;;
            2)
                TOTAL_NUM=`expr $TOTAL_NUM + ${FILESYSTEM[4]}`
                ;;
            3)
                TOTAL_NUM=`expr $TOTAL_NUM + ${SERVICE[4]}`
                ;;
            4)
                TOTAL_NUM=`expr $TOTAL_NUM + ${LOGGING[4]}`
                ;;
            5)
                TOTAL_NUM=`expr $TOTAL_NUM + ${ETC[4]}`
                ;;
            6)
                TOTAL_NUM=`expr $TOTAL_NUM + ${PATCH[4]}`
                ;;
        esac
    done

    PERCENT=`expr 100 / $TOTAL_NUM`
}

function first_menu()
{
    #점검항목 출력
    CHECKED_ITEM=$(dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --title
"Vulnerability Check system" --checklist "
Please select items to check." 15 55 6 "${ACCOUNT[0]}" "${ACCOUNT[3]}"  off ${FILESYSTEM[0]}
"${FILESYSTEM[3]}" off "${SERVICE[0]}" "${SERVICE[3]}" off "${LOGGING[0]}" "${LOGGING[3]}"  off
"${ETC[0]}" "${ETC[3]}"  off "${PATCH[0]}" "${PATCH[3]}"  off 2>&1 >/dev/tty)

    if [ -n "$CHECKED_ITEM" ]
    then
        #bar의 percent구하기
        get_percent

        #취약점 점검 시작
        start_check
```

```
        fi
}

function print_report()
{
        FILE=$(dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --stdout --title "Vulnerability Check system" --fselect ./result/ 14 48)
        dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --textbox $FILE 80 80
}

function print_info()
{
        dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --title "Vulnerability Check system" --msgbox "

[Profile]
 ☞ Name : Hye Been Lee
 ☞ Team : N2TG
 ☞ Univ : Joonbu University
 ☞ Major : Information Security
 ☞ Birth : 1996.01.11
 ☞ Blog : itlearner.tistory.com
 ☞ E-mail : v_v111222@nate.com


If you have any question, contact me." 20 43
}

function main_menu()
{
        #main menu 출력
        SELECTED_MENU=$(dialog --backtitle "ⓒ 2018.HYEBEEN.all rights reserved." --title "Vulnerability Check System" --menu "₩n                    Joongbu univ(Team N2TG)" 19 45 10 "1" "Vulnerability Check" "2"  "Report"  "3" "Info" "4" "Exit" 2>&1 >/dev/tty)

        #선택한 menu 처리
        if [ "$SELECTED_MENU" == "1" ]
        then
```

```
                first_menu
        elif [ "$SELECTED_MENU" == "2" ]
        then
                print_report
        elif [ "$SELECTED_MENU" == "3" ]
        then
                print_info
        fi
}

function main()
{
        while [[ "$SELECTED_MENU" != "4" ]]
        do
                main_menu
        done

        clear
}

main
```

<CentOS Script - Account>

```
#!/bin/sh

# U-10 불필요한 계정 제거

F_NAME="$(dirname $0)/result_$(basename $0 | sed 's/.sh//').txt"

##### 제목 출력 #####
echo "[LCv7-1.01]" > ${F_NAME}
echo "Default 계정 삭제 (상)" >> ${F_NAME}

##### 점검 현황 출력#####
echo "[점검 현황]" >> ${F_NAME}
echo "1. /etc/passwd파일의 Default 계정 목록 " >> ${F_NAME}
RESULT=$(cat                    /etc/passwd                    |                    egrep
```

```
"adm|lp|sync|shutdown|halt|news|uucp|operator|games|gopher|nfsnobody|squid")
if [ -z "$RESULT" ]
then
        echo "Default 계정이 없음" >> ${F_NAME}
else
        echo "$RESULT" >> ${F_NAME}
fi

##### 상태 출력#####
echo "[상태]" >> ${F_NAME}
RESULT=$(cat /etc/passwd | egrep "adm|lp|sync|shutdown|halt|news|uucp|operator|games|gopher|nfsnobody|squid":q | awk -F : "{print ₩$1}")
if [ -n "$RESULT" ]
then
        echo "취약" >> ${F_NAME}
else
        echo "양호" >> ${F_NAME}
fi

##### 점검 기준 출력 #####
echo "[점검 기준]" >> ${F_NAME}
echo "[양호]" >> ${F_NAME}
echo "불필요한 계정이 존재하지 않는 경우" >> ${F_NAME}

echo "[취약]" >> ${F_NAME}
echo "불필요한 계정이 존재하는 경우" >> ${F_NAME}
```

<CentOS Script - Etc>

```
#!/bin/sh

F_NAME="$(dirname $0)/result_$(basename $0 | sed 's/.sh//').txt"

##### 제목 출력 #####
echo "[LCv7-5.01]" > ${F_NAME}
echo "Anonymous FTP 비활성화 (상)" >> ${F_NAME}
```

```
##### 점검 현황 출력 #####
echo "[점검 현황]" >> ${F_NAME}
echo "1. Default FTP를 사용하는 경우" >> ${F_NAME}
echo "1-1. ftp또는 anonymous 계정의 존재 여부" >> ${F_NAME}
DEFAULT=$(awk -F ":" '$1=="anonymous" {print $0}' /etc/passwd)
DEFAULT=$DEFAULT"
"$(awk -F ":" '$1=="ftp" {print $0}' /etc/passwd)
echo "$DEFAULT" >> ${F_NAME}

echo "2. ProFTP 를 사용하는 경우" >> ${F_NAME}
echo "2-1. ftp또는 anonymous 계정의 존재 여부" >> ${F_NAME}
PROFTP=$(awk -F ":" '$1=="ftp" {print $0}' /etc/passwd)
echo "$PROFTP" >> ${F_NAME}

echo "3. vsFTP를 사용하는 경우" >> ${F_NAME}
if [ -e "/etc/vsftpd/vsftpd.conf" ]
then
        echo "3-1. /etc/vsftpd/vsftpd.conf 파일 설정" >> ${F_NAME}
        VSFTP=$(sed -n '/anonymous_enable=[YES|NO]/p' /etc/vsftpd/vsftpd.conf)
elif [ -e "/etc/vsftpd.conf" ]
then
        echo "3-1. /etc/vsftpd.conf 파일 설정" >> ${F_NAME}
        VSFTP=$(sed -n '/anonymous_enable=[YES|NO]/p' /etc/vsftpd.conf)
fi
echo "$VSFTP" >> ${F_NAME}

##### 상태 출력 #####
echo "[상태]" >> ${F_NAME}
RESULT="양호"

#Default FTP 점점"
DEFAULT=$(echo "$DEFAULT" | tr -d ' ')
if [ -n "$DEFAULT" ]
then
        RESULT="취약"
fi

#ProFTP 점점"
```

```
DEFAULT=$(echo "$PROFTP" | tr -d ' ')
if [ -n "$PROFTP" ]
then
        RESULT="취약"
fi

#vsFTP 점점"
if [ "$VSFTP" == "anonymous_enable=YES" ]
then
        RESULT="취약"
fi

echo "$RESULT" >> ${F_NAME}

##### 점검 기준 출력 #####
echo "[점검 기준]" >> ${F_NAME}
echo "[양호]" >> ${F_NAME}
echo "Anonymous FTP (익명 ftp) 접속을 차단한 경우" >> ${F_NAME}
echo "[취약]" >> ${F_NAME}
echo "Anonymous FTP ( 익명 ftp) 접속을 차단하지 않은 경우" >> ${F_NAME}
```

<CentOS Script - Filesystem>

```
#!/bin/sh

# U-32 UMASK 설정 관리

F_NAME="$(dirname $0)/result_$(basename $0 | sed 's/.sh//').txt"

##### 제목 출력 #####
echo "[LCv7-2.01]" > ${F_NAME}
echo "사용자 UMASK(User MASK) 설정 (하)" >> ${F_NAME}

##### 점검 현황 출력#####
echo "[점검 현황]" >> ${F_NAME}
echo "1. /etc/profile파일의 SUID와 SGID설정"  >> ${F_NAME}
PROFILE_UMASK=$(cat /etc/profile | grep -i umask | sed -n "s/^.*₩(umask₩s[0-9]₩)/₩1/p")
echo "$PROFILE_UMASK" >> ${F_NAME}
```

```
echo "2. /etc/bashrc파일의 SUID와 SGID설정"  >> ${F_NAME}
BASHRC_UMASK=$(cat /etc/bashrc | grep -i umask | sed -n "s/^.*₩(umask₩s[0-9]₩)/₩1/p")
echo "$BASHRC_UMASK" >> ${F_NAME}

echo "3. umask 명령어 확인"  >> ${F_NAME}
UMASK=$(umask)
echo $UMASK >> ${F_NAME}

##### 상태 출력#####
echo "[상태]" >> ${F_NAME}
RESULT="양호"
function get_result(){
        for i in {0..2}
        do
                if [ ${i} -eq 0 ]
                then
                        if [ ${1:${i}:1} -gt 0 ]
                        then
                                RESULT="취약"
                        fi
                fi
                if [ ${i} -eq 1 ]
                then
                        if [ ${1:${i}:1} -gt 2 ]
                        then
                                RESULT="취약"
                        fi
                fi
                if [ ${i} -eq 2 ]
                then
                        if [ ${1:${i}:1} -gt 2 ]
                        then
                                RESULT="취약"
                        fi
                fi
done
}
```

```
get_result $(echo $PROFILE_UMASK | awk -F ' ' "{print ₩$2}")
get_result $(echo $PROFILE_UMASK | awk -F ' ' "{print ₩$4}")
get_result $(echo $BASHRC_UMASK | awk -F ' ' "{print ₩$2}")
get_result $(echo $BASHRC_UMASK | awk -F ' ' "{print ₩$4}")
get_result ${UMASK:1:3}

echo $RESULT >> ${F_NAME}

##### 점검 기준 출력 #####
echo "[점검 기준]" >> ${F_NAME}
echo "[양호]" >> ${F_NAME}
echo "/etc/profile 또는 /etc/bashrc 파일에 UMASK 값이 022 이하로 설정된 경우" >> ${F_NAME}
echo "[취약]" >> ${F_NAME}
echo "/etc/profile 또는 /etc/bashrc 파일에 UMASK 값이 022 이하로 설정되지 경우" >> ${F_NAME}
```

<CentOS Script - Logging>

```
#!/bin/sh

F_NAME="$(dirname $0)/result_$(basename $0 | sed 's/.sh//').txt"

##### 제목 출력 #####
echo "[LCv7-4-01]" > ${F_NAME}
echo "시스템 로그 설정 (상)" >> ${F_NAME}

##### 점검 현황 출력#####
echo "[점검 현황]" >> ${F_NAME}
echo "1. 로그 기록 정책 수립 현황" >> ${F_NAME}
echo "로그 기록 정책 미수립" >> ${F_NAME}



##### 상태 출력#####
echo "[상태]" >> ${F_NAME}
RESULT="취약"
echo $RESULT >> ${F_NAME}
```

```
##### 점검 기준 출력 #####
echo "[점검 기준]" >> ${F_NAME}
echo "[양호]" >> ${F_NAME}
echo "로그 기록 정책이 정책에 따라 설정되어 수립되어 있는 경우" >> ${F_NAME}
echo "[취약]" >> ${F_NAME}
echo "로그 기록 정책 미수립, 또는 정책에 따라 설정되어 있지 않은 경우" >> ${F_NAME}
```

<CentOS Script - Patch>

```
#!/bin/sh

F_NAME="$(dirname $0)/result_$(basename $0 | sed 's/.sh//').txt"

##### 제목 출력 #####
echo "[LCv7-6-01]" > ${F_NAME}
echo "보안 패치 적용 (중)" >> ${F_NAME}

##### 점검 현황 출력#####
echo "[점검 현황]" >> ${F_NAME}
echo "1.인터뷰를 통해 주기적으로 보안 패치 적용 여부 확인"  >> ${F_NAME}
echo "주기적으로 보안 패치 적용"  >> ${F_NAME}

##### 상태 출력#####
echo "[상태]" >> ${F_NAME}
RESULT="양호"
echo $RESULT >> ${F_NAME}

##### 점검 기준 출력 #####
echo "[점검 기준]" >> ${F_NAME}
echo "[양호]" >> ${F_NAME}
echo "패치 적용 정책을 수립하여 주기적으로 패치를 관리하고 있는 경우" >> ${F_NAME}
echo "[취약]" >> ${F_NAME}
echo "패치 적용 정책을 수립하지 않고 주기적으로 패치관리를 하지 않는 경우" >> ${F_NAME}
```

<Kali GUI>

```
#include "mainwindow.h"
```

```
#include "ui_mainwindow.h"
#include <QFile>
#include <QTextStream>
#include <QMessageBox>
#include <QProcess>
#include <time.h>
#include <math.h>

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->StartButton->setEnabled(false);
}
int p_value = 0;

bool task_stop = 0;
bool chk_account;
bool chk_file;
bool chk_network;
bool chk_log;
bool chk_app;
bool chk_patch;

MainWindow::~MainWindow()
{
    delete ui;
}
void MainWindow::chk_start()
{
    ui->StartButton->setEnabled(p_max>0);
}
void MainWindow::reset()
{
    ui->cb_account->setChecked(false);
    ui->cb_filesystem->setChecked(false);
    ui->cb_network->setChecked(false);
```

```
    ui->cb_log->setChecked(false);
    ui->cb_app->setChecked(false);
    ui->cb_patch->setChecked(false);
    p_max = 0;
    ui->CheckButton->setEnabled(true);
    ui->cb_account->setEnabled(true);
    ui->cb_filesystem->setEnabled(true);
    ui->cb_network->setEnabled(true);
    ui->cb_log->setEnabled(true);
    ui->cb_app->setEnabled(true);
    ui->cb_patch->setEnabled(true);
    chk_start();
}
void MainWindow::on_CheckButton_clicked()
{

    if(p_max > 0)
    {
        ui->cb_account->setChecked(false);
        ui->cb_filesystem->setChecked(false);
        ui->cb_network->setChecked(false);
        ui->cb_log->setChecked(false);
        ui->cb_app->setChecked(false);
        ui->cb_patch->setChecked(false);
        p_max = 0;
    }
    else
    {
    ui->cb_account->setChecked(true);
    chk_account = true;
    ui->cb_filesystem->setChecked(true);
    chk_file = true;
    ui->cb_network->setChecked(true);
    chk_network = true;
    ui->cb_log->setChecked(true);
    chk_log = true;
    ui->cb_app->setChecked(true);
    chk_app = true;
```

```
	ui->cb_patch->setChecked(true);
	chk_patch = true;
	p_max = 44;
	}
	chk_start();

}

void MainWindow::on_StartButton_clicked()
{
	time_t now_time;
	struct tm *now_date;
	time(&now_time);
	now_date = localtime(&now_time);

	QProcess process;
	process.startDetached("rm", QStringList()<< "./log");
	ui->progressBar->setValue(0);
	task_stop = 0;
	ui->CheckButton->setEnabled(false);
	ui->cb_account->setEnabled(false);
	ui->cb_filesystem->setEnabled(false);
	ui->cb_network->setEnabled(false);
	ui->cb_log->setEnabled(false);
	ui->cb_app->setEnabled(false);
	ui->cb_patch->setEnabled(false);
	ui->StartButton->setEnabled(false);
	ui->progressBar->setMaximum(p_max);

	QFile file("./log");
	if(!file.open(QIODevice::ReadWrite| QIODevice::Append))
		QMessageBox::information(0,"file info",file.errorString());
	QTextStream stream(&file);

	stream << asctime(now_date);
	stream << "Start Vulnerability Check..."<<endl;
	if(chk_account)
	{
```

```
            stream <<"Start check account..."<<endl;
        }

        process.startDetached("/bin/sh", QStringList()<< "./debian_script/account/account_01.sh");
        stream << "account check finish!"<<endl;

        file.close();
        file.open(QIODevice::ReadWrite);
        ui->textBrowser->setText(file.readAll());
        file.close();
        file.open(QIODevice::ReadWrite | QIODevice::Append);
        stream << "Vulnerability Check finish!"<<endl;
        file.close();
        file.open(QIODevice::ReadWrite);
        ui->textBrowser->setText(file.readAll());
        file.close();
        ui->progressBar->setValue(1);

        reset();
}

void MainWindow::on_StopButton_clicked()
{
        task_stop = 1;
        reset();
}

void MainWindow::on_InfoButton_clicked()
{
        QMessageBox::information(0,"Argaon's Info","Major : InformationSecurity &
InformationSecurity S/W₩nServed in Army : InformationSecurity Tech Sergeant 13-2nd₩nBoB
7th Security Consulting");
}

void MainWindow::on_cb_account_clicked(bool checked)
{
        if(checked)
        {
```

```
        p_max+=8;
        chk_account = true;
    }
    if(checked == false)
    {
        p_max-=8;
        chk_account = false;
    }
    chk_start();
}

void MainWindow::on_cb_filesystem_clicked(bool checked)
{
    if(checked)
    {
        p_max+=16;
        chk_file = true;
    }
    if(checked == false)
    {
        p_max-=16;
        chk_file = false;
    }
    chk_start();
}

void MainWindow::on_cb_network_clicked(bool checked)
{
    if(checked)
    {
        p_max+=9;
        chk_network = true;
    }
    if(checked == false)
    {
        p_max-=9;
        chk_network = false;
    }
```

```
    chk_start();
}

void MainWindow::on_cb_log_clicked(bool checked)
{
    if(checked)
    {
        p_max+=2;
        chk_log = true;
    }
    if(checked == false)
    {
        p_max-=2;
        chk_log = false;
    }
    chk_start();
}

void MainWindow::on_cb_app_clicked(bool checked)
{
    if(checked)
    {
        p_max+=8;
        chk_app = true;
    }
    if(checked == false)
    {
        p_max-=8;
        chk_app = false;
    }
    chk_start();
}

void MainWindow::on_cb_patch_clicked(bool checked)
{
    if(checked)
    {
        p_max+=1;
```

```
            chk_patch = true;
        }
        if(checked == false)
        {
            p_max-=1;
            chk_patch = false;
        }
        chk_start();
}
```

<Kali Script - Account>

```
#!/bin/sh
LANG=ko_KR.UTF-8

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

ETC_PATH="/etc"

RESULT="$CURRENT_PATH/result_$NAME.txt"
ACCOUNT="lp|uucp|nuucp";

#리눅스 버전 : Debian 7
#계정관리 1.1 Default 계정 삭제
#주요정보통신기반시설_25p참고

#ETC 파일 체크 #

#/etc에서 file_passwd.txt 파일을 출력한다.

if [ -z ${PATH_PASSWD+x} ]; then
            PATH_PASSWD=/etc/passwd
fi

if [ ! -r ${ETC_PATH}/file_passwd.txt ]; then
            cat ${PATH_PASSWD} > ${ETC_PATH}/file_passwd.txt
```

```
fi
echo "[LDv7-1.01]" > ${RESULT} 2>&1
echo "Default 계정 삭제 (하)" >> ${RESULT} 2>&1
echo "[점검현황]" >> ${RESULT} 2>&1
echo "**** 전체 계정 목록 ****"  >> ${RESULT} 2>&1

cat ${ETC_PATH}/file_passwd.txt | awk -F: '{print $1}' >> ${RESULT} 2>&1

echo "**** 불필요한 계정 점검 ****" >> ${RESULT} 2>&1
cat ${PATH_PASSWD} | egrep "lp|uucp|nuucp" | awk -F: '{print $1}' >> ${RESULT} 2>&1
echo "[상태]" >> ${RESULT} 2>&1
if [ -z "'cat ${PATH_PASSWD} | egrep ${ACCOUNT}'" ];            then
        echo "양호" >> ${RESULT} 2>&1
elif [ ! -z "'cat ${PATH_PASSWD} | egrep ${ACCOUNT}'" ];       then
        echo "취약" >> ${RESULT} 2>&1
fi

# echo "양호" >> ${RESULT} 2>&1
echo "[점검 기준 ]" >> ${RESULT} 2>&1
echo "양호: 불필요한 계정이 존재하지 않는 경우" >> ${RESULT} 2>&1
echo "취약: 불필요한 계정이 존재하는 경우" >> ${RESULT} 2>&1
echo "**** 불필요한 계정 점검 항목이 공란이 아닐 경우, userdel <user_name> 을 사용하여
삭제를 권고함 ****" >> ${RESULT} 2>&1
```

<Kali Script - Filesystem>

```
#!/bin/bash
LANG=ko_KR.UTF-8

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

ETC_PATH="/etc"

RESULT="$CURRENT_PATH/result_$NAME.txt"

#리눅스 버전 : Debian 7
```

```
#계정관리 2.1 사용자 UMASK(User Mask)설정
#주요정보통신기반시설_54p참고

echo "[LDv7-2.01]" > ${RESULT} 2>&1
echo "사용자 UMASK(User Mask)설정(중)" >> ${RESULT} 2>&1
echo "[점검현황]" >> ${RESULT} 2>&1
UMASK=$(umask)
echo $UMASK >> ${RESULT} 2>&1
echo "[상태]" >> ${RESULT} 2>&1
function g_result()
{
        for i in 0 1 2 3
        do
                if [ ${i} -eq 0 ]
                then
                        if [ ${1:${i}:1} -gt 0 ]
                        then
                                STATUS="취약"
                        fi
                fi
                if [ ${i} -eq 1 ]
                then
                        if [ ${1:${i}:1} -gt 0 ]
                        then
                                STATUS="취약"
                        fi
                fi
                if [ ${i} -eq 2 ]
                then
                        if [ ${1:${i}:1} -gt 2 ]
                        then
                                STATUS="취약"
                        fi
                fi
                if [ ${i} -eq 3 ]
                then
                        if [ ${1:${i}:1} -gt 2 ]
                        then
```

```
                                        STATUS="취약"
                                fi
                        fi
done
}
g_result ${UMASK:1:5}
echo $STATUS >> ${RESULT} 2>&1

echo "[점검 기준 ]" >> ${RESULT} 2>&1
echo "양호: UMASK 값이 0022이상일 경우" >> ${RESULT} 2>&1
echo "취약: UMASK 값이 0022이하일 경우" >> ${RESULT} 2>&1
```

<Kali Script - Log>

```
#!/bin/bash
LANG=ko_KR.UTF-8

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

ETC_PATH="/etc"

RESULT="$CURRENT_PATH/result_$NAME.txt"

#리눅스 버전 : Debian 7
#계정관리 4.1 시스템 로그 설정(상)
#41p
echo "[LDv7-4.1]" > ${RESULT} 2>&1
echo "4.1 시스템 로그 설정(상)" >> ${RESULT} 2>&1
echo "[점검현황]" >> ${RESULT} 2>&1
cat /etc/rsyslog.conf >> ${RESULT} 2>&1
echo "수동 점검 필요">> ${RESULT} 2>&1
echo "[상태]" >> ${RESULT} 2>&1
echo "취약" >> ${RESULT} 2>&1
echo "[점검기준 ]" >> ${RESULT} 2>&1
```

```
echo "양호: 로그 기록 정책이 정책에 따라 설정되어 수립되어 있는 경우" >> ${RESULT} 2>&1
echo "취약: 로그 기록 정책 미수립 , 또는 , 정책에 따라 설정되어 있지 않은 경우" >> ${RESULT} 2>&1
echo "info 및 alert 등에 대해 적절히 기록하도록 설정되어 있는지 점검" >>${RESULT} 2>&1
```

<Kali Script - Network Service>

```
#!/bin/bash
LANG=ko_KR.UTF-8

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

ETC_PATH="/etc"

RESULT="$CURRENT_PATH/result_$NAME.txt"

#리눅스 버전 : Debian 7
#계정관리 3.1 RPC(Remote Procedure Call) 서비스 제한(중)

echo "[LDv7-3.1]" > ${RESULT} 2>&1
echo "3.1 RPC(Remote Procedure Call) 서비스 제한(중)" >> ${RESULT} 2>&1
echo "[점검현황]" >> ${RESULT} 2>&1
FILE_LIST=(rpc.cmsd rpc.ttdbserverd sadmind rusersd walld sprayd rstatd rpc.nisd rexd
rpc.pcnfsd rpc.statd rpc.ypupdated rpc.rquotad kcms_server cachefsd)
ACTIVE_SERVICE=''
COMMA=','
STATUS="양호"
function check_service()
{
        FILE_CONTENTS=$(cat /etc/xinetd.d/$1 2>/dev/null)
        FILE_CONTENTS=$(echo $FILE_CONTENTS | sed 's/ //g')
        if [[ "$FILE_CONTENTS" =~ "disable=no" ]]
        then
                STATUS="취약"
                ACTIVE_SERVICE=$ACTIVE_SERVICE$COMMA$1
```

```
        fi
}

for FILE in ${FILE_LIST[@]}
do
        check_service $FILE
done

echo "${ACTIVE_SERVICE:1}" >> ${RESULT} 2>&1
echo "[상태]" >> ${RESULT} 2>&1
echo $STATUS >> ${RESULT} 2>&1
echo "[점검 기준 ]" >> ${RESULT} 2>&1
echo "양호: 불필요한 서비스가 비활성화 되어 있는 경우 " >> ${RESULT} 2>&1
echo "취약: 불필요한 서비스가 활성화 되어 있는 경우" >> ${RESULT} 2>&1
```

<Kali Script - Network Service>

```
#!/bin/sh
LANG=ko_KR.UTF-8

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

ETC_PATH="/etc"

RESULT="$CURRENT_PATH/result_$NAME.txt"

#리눅스 버전 : Debian 7



echo "[LDv7-6.1]" > ${RESULT} 2>&1
echo "6.1 보안 패치 적용 (상)" >> ${RESULT} 2>&1
echo "[점검현황]" >> ${RESULT} 2>&1
echo "수동 점검 필요" >> ${RESULT} 2>&1
echo "[상태]" >> ${RESULT} 2>&1
echo "양호" >> ${RESULT} 2>&1
echo "[점검기준 ]" >> ${RESULT} 2>&1
```

```
echo "양호: 패치 적용 정책을 수립하여 주기적으로 패치를 관리하고 있는 경우" >> ${RESULT} 2>&1
echo "취약: 패치 적용 정책을 수립하지 않고 주기적으로 패치관리를 하지 않는 경우" >> ${RESULT} 2>&1
```

5-1-4. Unix

<Solaris GUI>

```
#include <gtk/gtk.h>
#include <stdlib.h>
#include <stdio.h>

void button_clicked(GtkWidget *widget, gpointer data)
{
	system("./account/account*.sh");
}

void button_clicked2(GtkWidget *widget, gpointer data)
{
	system("./service/service*.sh");
}

void button_clicked3(GtkWidget *widget, gpointer data)
{
	system("./network/network*.sh");
}

void button_clicked4(GtkWidget *widget, gpointer data)
{
	system("./filesystem/filesystem*.sh");
}

void button_clicked5(GtkWidget *widget, gpointer data)
{
	system("./account/account*.sh");
}
```

```
void button_clicked6(GtkWidget *widget, gpointer data)
{
	system("./account/account*.sh");
}

void button_clicked7(GtkWidget *widget, gpointer data) {

	system("cat ./readme.txt");
}

void button_clicked8(GtkWidget *widget, gpointer data) {

	system("cat ./info.txt");
}


int main(int argc, char *argv[]) {

   GtkWidget *window;
   GtkWidget *fixed;
   GtkWidget *halign;
   GtkWidget *vbox;
   GtkWidget *hbox;
   GtkWidget *cb;
   GtkWidget *button;
   GtkWidget *button2;
   GtkWidget *button3;
   GtkWidget *button4;
   GtkWidget *button5;
   GtkWidget *button6;
   GtkWidget *button7;
   GtkWidget *button8;
   GtkWidget *button9;
   GtkWidget *align;
   GtkWidget *lbl;
   GtkWidget *frame;
```

```
gtk_init(&argc, &argv);

window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
gtk_window_set_default_size(GTK_WINDOW(window), 300, 300);
gtk_container_set_border_width(GTK_CONTAINER(window), 5);
gtk_window_set_position(GTK_WINDOW(window), GTK_WIN_POS_CENTER);



fixed = gtk_fixed_new();
gtk_container_add(GTK_CONTAINER(window), fixed);

button = gtk_button_new_with_label("Account");
gtk_fixed_put(GTK_FIXED(fixed), button, 0, 0);
gtk_widget_set_size_request(button, 200, 60);
button = g_signal_connect(G_OBJECT(button), "clicked",
      G_CALLBACK(button_clicked), NULL);

button2 = gtk_button_new_with_label("Service");
gtk_fixed_put(GTK_FIXED(fixed), button2, 245, 0);
gtk_widget_set_size_request(button2, 200, 60);
button2 = g_signal_connect(G_OBJECT(button2), "clicked",
      G_CALLBACK(button_clicked2), NULL);

button3 = gtk_button_new_with_label("Network Service");
gtk_fixed_put(GTK_FIXED(fixed), button3, 0, 70);
gtk_widget_set_size_request(button3, 200, 60);
button3 = g_signal_connect(G_OBJECT(button3), "clicked",
      G_CALLBACK(button_clicked3), NULL);

button4 = gtk_button_new_with_label("Filesystem");
gtk_fixed_put(GTK_FIXED(fixed), button4, 245, 70);
gtk_widget_set_size_request(button4, 200, 60);
button4 = g_signal_connect(G_OBJECT(button4), "clicked",
      G_CALLBACK(button_clicked4), NULL);

button5 = gtk_button_new_with_label("Log Management");
```

```
gtk_fixed_put(GTK_FIXED(fixed), button5, 0, 140);
gtk_widget_set_size_request(button5, 200, 60);
button5 = g_signal_connect(G_OBJECT(button5), "clicked",
      G_CALLBACK(button_clicked5), NULL);

button6 = gtk_button_new_with_label("Security Patches");
gtk_fixed_put(GTK_FIXED(fixed), button6, 245, 140);
gtk_widget_set_size_request(button6, 200, 60);
button6 = g_signal_connect(G_OBJECT(button6), "clicked",
      G_CALLBACK(button_clicked6), NULL);

button7 = gtk_button_new_with_label("Readme");
gtk_fixed_put(GTK_FIXED(fixed), button7, 0, 210);
gtk_widget_set_size_request(button7, 150, 60);
button7 = g_signal_connect(G_OBJECT(button7), "clicked",
      G_CALLBACK(button_clicked7), NULL);

button8 = gtk_button_new_with_label("Info");
gtk_fixed_put(GTK_FIXED(fixed), button8, 165, 210);
gtk_widget_set_size_request(button8, 150, 60);
button8 = g_signal_connect(G_OBJECT(button8), "clicked",
      G_CALLBACK(button_clicked8), NULL);

button9 = gtk_button_new_with_label("Exit");
gtk_fixed_put(GTK_FIXED(fixed), button9, 345, 210);
gtk_widget_set_size_request(button9, 100, 60);
button9 = g_signal_connect(G_OBJECT(button9), "clicked",
      G_CALLBACK(gtk_main_quit), NULL);

frame = gtk_frame_new(NULL);
gtk_container_add(GTK_CONTAINER(window), frame);

gtk_frame_set_label( GTK_FRAME(frame), "GTK Frame Widget" );

gtk_frame_set_label_align( GTK_FRAME(frame), 0.0, 0.0);

gtk_frame_set_shadow_type( GTK_FRAME(frame), GTK_SHADOW_ETCHED_OUT);
```

```
    gtk_widget_show(frame);
    gtk_widget_show_all(window);

    g_signal_connect(G_OBJECT(window), "destroy",
          G_CALLBACK(gtk_main_quit), NULL);



    gtk_main();

    return 0;
}
```

<Solaris Script - Account>

```
#!/bin/sh



# Default 계정 삭제

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"



echo "[U-01]" >> ${RESULT}
echo "Default 계정 삭제" >> ${RESULT}
echo "[점검 현황]" >> ${RESULT}
cat /etc/passwd >> ${RESULT}
echo "" >> ${RESULT}

### 상태
echo "[상태]" >> ${RESULT}

default="adm|lp|sync|shutdown|halt|news|uucp|operator|games|gopher|nfsnobody|squid"
```

```
if [ -z "`cat /etc/passwd | egrep ${default}`" ]; then >> ${RESULT}
        echo "양호" >> ${RESULT}
else
        echo "취약" >> ${RESULT}
fi

#### 점검 기준
echo "[점검 기준]" >> ${RESULT} 2>&1
echo "[양호]" >> ${RESULT} 2>&1
echo "- 불필요한 계정이 존재하지 않는 경우" >> ${RESULT} 2>&1
echo "[취약]" >> ${RESULT} 2>&1
echo "- 불필요한 계정이 존재하는 경우" >> ${RESULT} 2>&1
```

<Solaris Script - Filesystem>

```
#!/bin/sh



CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"

echo "[U-01]" >> ${RESULT}
echo "사용자 UMASK 설정 관리" >> ${RESULT}
echo "----------------------------" >> ${RESULT}
echo "[점검 현황]" >> ${RESULT}
cat /etc/profile | egrep "umask" >> ${RESULT}

echo "" >> ${RESULT}

#### 상태

echo "----------------------------" >> ${RESULT}
echo "[상태]" >> ${RESULT}
```

```
profile=
if [ `cat /etc/profile | egrep "umask" | cut -d' ' -f2` -le 022 ]; then >> ${RESULT}
        echo "양호" >> ${RESULT}
else
        echo "취약" >> ${RESULT}
fi


#### 점검 기준
echo "[점검 기준]" >> ${RESULT} 2>&1
echo "[양호]" >> ${RESULT} 2>&1
echo "- UMASK 값이 022 이하로 설정된 경우" >> ${RESULT} 2>&1
echo "[취약]" >> ${RESULT} 2>&1
echo "- UMASK 값이 022 이하로 설정되지 않은 경우" >> ${RESULT} 2>&1
```

<Solaris Script - Network>

```
#!/bin/sh

#### U-62

CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"

echo "[U-01]" >> ${RESULT}
echo "RPC 서비스 확인" >> ${RESULT}
echo "[점검 현황]" >> ${RESULT}
inetadm | egrep "ttdbserver|rex|rstat|rusers|spray|wall|rquota" | grep enabled >> ${RESULT}
2>&1

#### 상태

echo "[상태]" >> ${RESULT}
```

```
if [ -z "`inetadm | egrep "ttdbserver|rex|rstat|rusers|spray|wall|rquota" | grep enabled`" ]; then >>
${RESULT}
        echo "양호" >> ${RESULT}
else
        echo "취약" >> ${RESULT}
fi



#### 점검 기준
echo "[점검 기준]" >> ${RESULT} 2>&1
echo "[양호]" >> ${RESULT} 2>&1
echo "- 불필요한 RPC 서비스가 비활성화 되어 있는 경우" >> ${RESULT} 2>&1
echo "[취약]" >> ${RESULT} 2>&1
echo "- 불필요한 RPC 서비스가 활성화 되어 있는 경우" >> ${RESULT} 2>&1
```

<Solaris Script - Service>

```
#!/bin/sh



CURRENT_PATH=`dirname $0`

NAME=`basename $0`

RESULT="$CURRENT_PATH/result_$NAME.txt"

echo "[U-01]" >> ${RESULT}
echo "Dos 공격에 취약한 서비스 비활성화" >> ${RESULT}
echo "[점검 현황]" >> ${RESULT}
svcs -a | egrep "echo|daytime|discard|chargen" >> ${RESULT}
echo "" >> ${RESULT}

#### 상태

echo "[상태]" >> ${RESULT}

if [ -z "`svcs | grep echo|daytime|discard|chargen`" ]; then >> ${RESULT}
        echo "양호" >> ${RESULT}
```

```
else
        echo "취약" >> ${RESULT}
fi



#### 점검 기준
echo "[점검 기준]" >> ${RESULT} 2>&1
echo "[양호]" >> ${RESULT} 2>&1
echo "- SNMP 서비스를 사용하지 않는 경우" >> ${RESULT} 2>&1
echo "[취약]" >> ${RESULT} 2>&1
echo "- SNMP 서비스를 사용하는 경우 " >> ${RESULT} 2>&1
```