情 報 保 護 學 會 誌
第19卷 第2號, 2009. 4

SELinux 보안 정책 복잡성 개선을 위한 보안 정책 설정 도구

이 재 서*, 김 민 수**, 노 봉 남***

요 약

본 논문에서는 가장 대표적인 보안 운영체제인 SELinux를 보다 쉽게 이용할 수 있도록 하는 보안 정책 템플릿과 도구를 설명한다. 제안하는 방법론을 위해 먼저 SELinux를 보다 편리하게 이용할 수 있도록 하기 위해 연구된 기존에 도구들이 가지는 문제점을 분석하였다. 그리고 SELinux를 심도 깊게 분석하여 SELinux에서 이용하는 주체, 객체의 타입 그리고 권한을 객체 타입과 객체별, 권한별로 축약하였다. 이처럼 축약되어 간소화된 객체 타입과 권한을 바탕으로 리눅스 시스템이 이용하는 임의적인 접근통제 형태를 지원하는 보안 정책 템플릿 기술언어를 정의하여 SELinux 보안 정책의 복잡성을 개선하도록 하였으며 이를 통해 기존 리눅스 시스템 관리자가 이용하기 편리하도록 하였다. 또한 자동으로 제안된 보안 정책 템플릿 언어로 작성된 보안 정책을 SELinux의 보안 정책으로 변환하는 변환기와 자동으로 정책을 생성하는 생성기를 개발하여 SELinux 이용의 편리성을 최대화 하였다.

Ⅰ. 서 론

인터넷이 발달하고 보안에 대한 중요성이 강조 되면서 국내외에서 보안 운영체제에 관한 개발 움직임이 활발하게 진행되고 있다. 보안 운영체제의 가장 대표적인 것은 미국 국가안전보장국(NSA: National Security Agency) 주도로 개발된 SELinux(Security-Enhanced Linux)이다. SELinux는 유타(Utah)대학에서 개발한 Flask(Flux Advanced Security Kernel) 구조와 TE (Type Enforcement) 정책 모델을 리눅스 시스템에 적용한 것으로 TE 모델을 통해 강제적 접근통제(MAC: Mandantory Access Control) 및 다중등급보안(MLS: Multi-Level Security) 접근통제 정책을 지원하고 있다. 최근 리눅스 커널 2.6에서부터 바닐라(vanilla) 커널에 포함되면서 가장 널리 퍼진 보안 운영체제가 되었다[1,2].

SELinux는 주체(subject)와 객체(object)의 아이덴디티(identity)를 타입(type)으로 표현하고 주체와 객체의 권한(permission) 관계를 타입 간에 권한 집합(permissions-set)을 통하여 표현한다. SELinux는 주체의 대상을 프로세스(process)로하고 객체의 대상으로 파일(file), 디렉터리(directory) 그리고 소켓(socket) 등 리눅스 시스템의 모든 자원(resource)을 목표로 하고 있다. 그리고 권한을 시스템 콜(system call) 수준으로 정의하고 있다[1]. 이처럼 객체의 종류와 개수가 많고 수 많은 권한으로 이루어져있는 SELinux는 아주 복잡하여 전문지식을 갖고 있지 않은 일반 사용자(보안 정책 관리자)가 이용하기에는 어려움이 많다. 또한 기존 정책간의 복잡한 연관성은 사용자가 쉽게 규칙을 변경, 삭제하기 어렵게 할 뿐만 아니라 새로운 정책을 추가하기 위해서는 기존의 정책들에 대해서 분석해야하는 등 어려움이 있는 상황이다. 따라서 보다 쉽게 SELinux 이용하기 위한 연구가 요구되고 있다.

본 논문에서는 SELinux 복잡성을 개선하여 보다 쉽게 이용할 수 있도록 하는 보안 정책 템플릿을 제안한다. 제안된 도구는 SELinux의 주체, 객체의 타입 그리고 권한을 분석하여 기존의 리눅스 시스템에 적용되어 있던 카파빌러티(capability)와 권한(permission)으로 구성된 임의적 접근통제(DAC: Discretionary Access

* 전남대학교 시스템보안연구센터 (mirr1004@lsrc.jnu.ac.kr)
** 목포대학교 정보보호학과 (phoenix@mokpo.ac.kr)
*** 전남대학교 전자컴퓨터공학부 (bbong@chonnam.ac.kr)

Control) 형태로 단순화 하여 SELinux 복잡성을 개선하도록 하였다. 또한 제안된 도구를 보다 편리하게 이용할 수 있는 방법인 정책 자동 생성 방법을 제안하고 관련 도구들과 분석 결과를 보인다.

본 논문의 구성은 다음과 같다. 2장에서 SELinux 복잡성을 분석 방법을 설명하고, 3장에서 SELinux 복잡성 개산을 위한 기술을 살펴보고, 4장에서 제안하는 SELinux 복잡성 개선 방법과 도구를 설명하며 5장에서는 제안하는 방법의 성능을 분석한다. 마지막으로 6장에서 결론을 맺는다.

Ⅱ. SELinux 복잡성 분석

이 장에서는 대표적인 리눅스 배포판인 페도라 코어 4 버전에서 기본으로 포함되어 있는 정책 소스를 분석하여 SELinux가 가지는 복잡성을 설명하고 이러한 복잡성 개선을 위한 기술인 SETools[5]와 SEEdit[6] 그리고 polgen[7]에 대해서 설명한다.

2.1 SELinux 구조 복잡성 분석

최근의 SELinux는 리눅스 시스템 자원을 [표 1]과 같이 커널 객체 클래스 42개와 사용자 영역(userland) 객체 클래스 14개로 분류하고 있다[3]. 그리고 각 객체 클래스별로 권한의 개수는 1개부터 31개로 구성되며 전체적으로 212개의 권한이 이용되고 있다. 이처럼 SELinux내에는 많은 객체 클래스와 권한이 존재하고 있으며 이것은 숙달되지 않은 일반 사용자들에게 사용상의 어려움이 되는 주요 원인이 되고 있다. 이 절에서는 SELinux에서 정의하고 있는 객체의 클래스와 클래스별 권한들에 대한 복잡성을 분석한다.

SELinux에는 위의 [표 1]과 같이 총 56개의 객체 클래스가 존재하며 각 객체 클래스별로 1개부터 31개의 권한을 갖으며 총 권한의 개수는 212개이다. SELinux 전체의 복잡성을 두 개의 입력 변수인 총 클래스 개수: $n(classes)$와 총 권한의 개수: $n(permissions)$의 곱으로 계산하면 분석할 경우 SELinux 정책 복잡도(PC: policy complexity)는 11,872가 되며 이를 식 1과 같은 SELinux 정책 단순 복잡성(SPC: simple policy complexity)이라 정의한다.

$$SPC = n(classes) \cdot n(permissions) \quad (1)$$

또한 SELinux 정책 복잡성을 계산하기 위해 임의의 객체에 임의의 정책을 설정할 때 있어서 나타날 수 있는 모든 경우의 수를 계산할 필요가 있다. 이를 SELinux 구조 전체 복잡성(FPC: fully policy complexity)라 정의하며 계산 방법은 식 2와 같다.

$$FPC = 2^{n(permissions)} - 1 \quad (2)$$

임의의 객체에 설정될 수 있는 임의의 정책에는 최소한 한 개 이상의 클래스를 가져야하며 다른 모든 클래스들과 함께 설정될 수 있다. 마찬가지로 각 클래스별로 설정될 수 있는 권한은 최소한 한 개 이상의 권한을 가져야 하며 다른 모든 권한을 가질 수 있다. 따라서 임

〔표 1〕 SELinux 객체 클래스(class)와 클래스별 권한 개수

분 류	클래스(class)	개수	클래스	개수	클래스	개수	클래스	개수	클래스	개수
File classes	blk_file	17	chr_file	20	dir	22	fd	1	fifo_file	17
	file	20	filesystem	10	lnk_file	17	sock_file	17	–	–
IPC classes	ipc	9	msg	2	msgq	10	sem	9	shm	10
Network classes	key_socket	22	netif	6	netlink(11)	24	packet_socket	22	node	7
	rawip_socket	23	socket	22	tcp_socket	27	udp_socket	23	unix_dgram_socket	22
	unix_stream_socket	25	association	2	appletalk_socket	22	–	–	–	–
Userland classes	passwd	4	dbus	2	nscd	8	drawable	5	window	26
	gc	4	font	4	colormap	9	property	4	cursor	5
	xclient	1	xinput	11	xserver	8	xextension	2	–	–
System classes	capability	31	process	25	security	17	system	8	pax	6

의의 객체에 설정될 수 있는 권한의 모든 경우의 수는 모두 클래스별 권한 개수의 곱으로 계산될 수 있다. [표 1]에 대해 계산한 SELinux 정책 전체 복잡성은 $2^{212}-1$ 이 된다.

2.2 SELinux 공개 정책 복잡성 분석

SELinux는 기본 보안 정책으로 두 가지 종류에 정책을 공개되어 있다. 첫 번째는 엄격한 정책(strict policy)이다. 엄격한 정책은 보안 정책이 설정되어 있는 주체만을 대상으로 접근통제를 실시하며 이외의 모든 주체들은 전부 차단되도록 되어 있다. 두 번째는 지정 정책(targeted policy)이다. 지정 정책은 보안 정책이 설정되어 있지 않은 주체들은 모두 허용하고 지정되어 있는 주체들만으로 접근통제를 하도록 되어 있다[4]. SELinux 공개 정책 복잡성 분석에서는 위의 두 종류의 정책을 구성하고 있는 정책 파일의 개수와 설정되어 있는 타입의 개수 그리고 규칙의 개수를 정량적으로 분석한다.

[표 2]는 페도라 코어 4 배포판에 기본으로 제공하고 있는 SELinux 공개 정책을 정량적으로 분석한 결과이다. 엄격한 정책과 지정 정책에는 주요 서비스와 프로그램에 적용하기 위해 94개(*.te, *.fc)의 정책 파일로 구성되어 있으며 주체와 객체에 할당되는 타입의 개수는 각각 1,341개, 764개가 존재 한다. 그리고 임의의 주체가 임의의 객체에 허용할 권한을 명시하는 규칙의 개수는 각각 약 34만개, 약 18만개에 이른다. 이처럼 수 천, 수 만개의 타입과 규칙이 존재하는 SELinux의 정책을 설정하는 것은 굉장히 복잡하고 어려우며 많은 시간이 요구된다.

〔표 2〕 SELinux 공개 정책 복잡성 정량적 분석 결과

정책 이름	정책 개수	타입 개수	규칙 개수
엄격한 정책 (strict policy)	94개	1,341개	345,260개
지정 정책 (targeted policy)	94개	764개	180,461개

Ⅲ. SELinux 복잡성 개선을 위한 기술

SELinux 복잡성을 개선하기 위한 연구는 [표 3]과 같이 크게 세 가지 방법으로 연구가 진행되고 있다. 첫 번째는 SELinux 정책을 작성하는데 있어서 활용할 수 있도록 하기 위한 도구를 개발하는 것이다. 가장 대표적인 것으로는 Tresys Technology사에서 개발한 SETools가 있다[5]. SETools는 그래픽 사용자 인터페이스를 지원하여 정책을 설정할 수 있도록 하며 주체, 객체 그리고 규칙 등 SELinux 구성 요소별 검색 기능 등을 지원한다. 그러나 SETools는 SELinux 정책을 설정하는데 있어서 큰 도움이 되지만 SELinux 정책 자체의 복잡성을 개선하지는 않는다. 두 번째는 SELinux를 추상화하여 보다 기존 리눅스 사용자들에게 친숙한 언어를 개발하는 것이다. 이에 가장 대표적인 것으로는 Hitachi Software사에서 개발한 SEEditor가 있다[6,7]. SEEditor는 간소화된 규칙인 SPDL(Simplified Policy Description Language)를 개발하여 SELinux의 복잡성을 개선하고 있다. 그러나 SEEditor는 고정적인 오퍼레이션(operation) 축약으로 SELinux의 최대 장점인 세밀한(fine-grained) 접근통제가 가능하지 않는다는 단점이 있다. 마지막 세 번째는 SELinux 정책 생성을 자동화하는 도구를 개발하는 것이다. 이에 가장 대표적인 것으로는 MITRE사에서 개발한 polgen이 있다[8,9,10]. polgen은 정책을 설정하고자 하는 프로그램을 실행하고 프로세스에서 발생하는 정보흐름을 분석하여 SELinux 정책을 자동으로 생성한다. 그러나 polgen으로 만들어지는 SELinux 정책의 완성도가 떨어진다는 것과 polgen으로 만들어진 SELinux 정책이 어렵기 때문에 사용자가 보완(customizing)할 수 없다는 단점이 있다.

〔표 3〕 SELinux 복잡성 개선을 위한 기술

종 류	기 술	특 징
응용 도구 개발	Tresys Technology SETools	· X윈도우 환경 GUI 제공하여 SELinux 보안 정책 생성에 도움 · 정책 구성요소 검색기능 · 보고서 생성기능 · SELinux 보안 정책 복잡성 개선이 이루어지지 않음
추상화된 상위 언어 개발	Hitachi Software SEEditr	· 간소화된 규칙인 SPDL 이용하여 SELinux 복잡성 개선 · X윈도우, 웹 브라우저, 시스템 콘솔 환경 도구제공 · GPL을 따르는 공개소스 배포 · 고정적 오퍼레이션 축약으로 세밀한 접근통제가 불가능
자동 정책 생성 도구 개발	MITRE Corporation SLAT and polgen	· 정책 자동 생성을 통하여 SELinux 보안 정책 생성에 도움 · 정보흐름 검사기능 · SELinux 보안 정책 복잡성 개선이 이루어지지 않음

3.1 Tresys Technology의 SETools

SETools는 보안 정책 관리자가 SELinux 정책에 대한 세부적인 분석과 디버깅을 할 수 있도록 하는 도구들의 모음이다. SETools의 하나인 Apol은 정책파일을 분석하고 정책 구성요소들을 열람 및 검색할 수 있는 기능을 하며 SEDiff는 두개의 정책파일을 비교 분석하여 의미적인 차이점을 알려준다. 이외에도 SEAudit는 감사로그파일을 열람, 검색 및 실시간 감시를 지원하는 등 여러 가지 기능면에서 정책 생성을 하는데에 있어서 편리성을 주려고 노력하였다. 그러나 SETools는 SELinux와 SELinux 보안 정책이 가지고 있는 정책에 대한 복잡성을 개선한 것이 아니기 때문에 마찬가지로 일반적인 사용자에게는 SETools 자체만으로도 부담을 줄 수 있다[5].

3.2 Hitachi Software의 SEEditor(SELinux Policy Editor)

SEEditor는 간소화된 규칙언어인 SPDL을 이용하여 SELinux의 클래스에 따른 권한을 r, w, x, s와 같이 임의적 접근통제 형태로 단순화하였다. 그리고 보안 정책 설정에 대한 여러 관점을 제공하여 특정 프로세스가 접근 가능한 자원 목록 보기 기능과 특정자원에 접근이 가능한 프로세스의 목록 보기 기능을 지원한다. 이와 같이 정책 복잡성 감소, 권한집합 축소, 정책 자동생성 그리고 웹 사용자 인터페이스를 제공하고 있는 webmin과 GTK+를 통해 구현된 그래픽 사용자 인터페이스 환경의 관리 도구 등 여러 가지 면에서 사용자의 편의성을 높여 SELinux가 갖는 정책 복잡성을 상당부분 개선하였다. 하지만 SEEdit의 간소화된 정책의 정보흐름 분석 가능성이 불명확하며 고정적으로 권한을 축약하여 세밀한 접근통제가 불가능 하며 간소화된 규칙 언어인 SPDL 자체에 대한 지식이 요구된다[6,7].

3.3 MITRE Corporations의 SLAT and polgen

SLAT과 polgen는 정책 생성과 정책 분석을 수행하는 도구이다. SLAT는 SELinux 보안 정책 설정이 보안 목표와 부합하는지 판단해주는 도구로써 보안 목표를 표현할 수 있는 간단한 문맥을 제공하며, polgen은 시스템에 설치되어 있는 프로그램들에 대해 동작 패턴에 기반 한 정책을 자동으로 생성한다. 그러나 위의 SLAT과 polgen은 각 단계별로 관리자의 부가적 지식을 요구할 뿐만 아니라 타입을 생성하는 typegen 명령어를 수행하기 전에 대상 프로그램이 접근하는 객체들에 대한 정보를 사용자가 직적 입력해야 하는 번거로움이 있다. 그리고 사용자의 실수로 누락된 파일들에 대한 보안 정책을 자동으로 생성해주지 않아 올바른 정책이 생성되지 않을 수 있다[8,9,10].

Ⅳ. SELinux 복잡성 개선을 위한 연구 (SELT : Security-Enhanced Linux Template)

본 논문에서 제안하는 SELinux 복잡성 개선을 위한 연구는 SELinux 구조를 분석하고 추상화하여 상위 언어인 보안 정책 템플릿(SELT)을 개발하고 SELT 기술 언어로 작성된 정책을 SELinux 정책으로 변환하는 방법을 연구한다. 또한 정책을 작성하고자 하는 대상 프로그램의 정보흐름을 자동으로 분석하여 SELT 기술 언어로 자동으로 생성할 수 있는 방법을 연구한다.

4.1 SELinux 추상화를 통한 복잡성 개선

SELinux는 [표 1]과 같이 다양한 객체 타입과 각 객체별 많은 권한을 정의하고 있으며 이를 통해 세분화된 접근 통제를 가능하게 하여 잠재적인 보안 위협으로부터 피해를 최소화할 수 있도록 하였다. 반면에 전문 지식이 없는 일반 리눅스 시스템 이용자에게 있어서 SELinux의 보안 정책을 설정하는 것을 어렵게 하였고 이로 인해 실제로 SELinux가 대중화되지 못하고 있다. 결과적으로 SELinux가 추구되어야할 궁극적인 목표인 SELinux의 대중화를 통해 컴퓨팅 환경의 전체적인 보안성 향상을 위해서 SELinux의 복잡성 개선이 필수적이 되었다. 이를 위한 가장 최선의 방법은 SELinux에 존재하는 다양한 주체와 객체의 클래스를 꼭 필요한 부분만으로 최소화하고 순차관계와 연관성을 갖는 권한들은 통합하여 권한을 단순화하는 것이다. 본 절에서는 객체 클래스 축약과 권한 축약을 위한 방법을 설명한다.

4.1.1 객체 클래스 축약을 통한 복잡성 감소 연구

객체 클래스 축약을 통한 복잡성 감소 연구에서는 SELinux에서 세분화하여 정의한 객체 클래스들과 각

클래스별 권한들의 관계를 분석하여 권한들이 같은 것들을 하나의 객체 클래스로 통합하였다. SELinux는 파일 객체와 관련하여 file, blk_file, chr_file, fifo_file, lnk_file, sock_file 그리고 dir 등 7개의 객체 클래스가 정의 되어 있다. 각 클래스별로 권한들을 정의하고 있으며 이 중 blk_file, fifo_file, lnk_file 그리고 sock_file 객체 클래스는 특정(unique) 권한을 갖지 않고 파일 클래스의 일반(common) 권한들만으로 정의되어있다. 그리고 문자 장치 파일 클래스인 chr_file와 파일 클래스인 file은 동일하게 일반 권한과 execute_no_trans, entrypoint 그리고 execmod 권한을 정의하고 있다. 이와 같이 파일 클래스별 권한들 간에 관계를 분석하여 객체 클래스들을 통합하면 [표 4]와 같이 객체 클래스를 축약할 수 있다. [표 4]에서는 공통 권한은 생략하고 특정 권한만을 나타내고 있다. 공통 권한에는 getattr, read, relabelto, ioctl, append, setattr, swapon, write, lock, create, rename, mounton, quotaon, relabelfrom, link, unlink, execute가 있다.

〔표 4〕 SELinux 파일 클래스 축약

클래스	특정(unique) 권한	SELT
file	execute_no_trans, entrypoint, execmod	file
chr_file	execute_no_trans, entrypoint, execmod	
blk_file	none	special_file
filo_file	none	
lnk_file	none	
sock_file	none	
dir	search, rmdir, getattr, remove_name, reparent, add_name	dir

〔표 5〕 SELinux 파일 클래스 관련 권한 축약

SELT	SELinux	설 명
common	getattr	파일 속성 읽기
read (r)	read	파일을 읽을 수 있음
write (w)	relabelto, ioctl, append, setattr, swapon, write, lock, create, rename, mounton, quotaon, relabelfrom, link	파일에 쓸 수 있음
remove (r)	unlink	파일을 지울 수 있음
execute (x)	execute execute_no_trans, entrypoint, execmod	파일을 실행할 수 있음

4.1.2 권한 축약을 통한 복잡성 감소 연구

SELinux는 각 객체 클래스별로 적게는 1개에서 많게는 31개까지 많은 권한들을 정의하고 있다. 리눅스 시스템에서는 파일 객체인 경우 임의적 접근통제에서의 퍼미션(permission)에서 이용하는 r, w, x 권한들만으로 정의하고 있기 때문에 몇 배나 많은 SELinux의 권한들을 이용하여 보안 정책을 설정하기에는 어려움이 있다. 따라서 이러한 점을 문제로 제기하고 기존 리눅스 시스템과 같은 편리성을 위해 퍼미션인 r, w, x와 대응하는 권한을 정의하고 이렇게 정의된 권한과 SELinux에서의 권한들 간에 관계를 비교하여 권한을 분류하면 [표 5]와 같이 권한을 축약할 수 있다.

4.2 보안 정책 템플릿 기술언어

보안 정책 템플릿 기술언어는 기존의 리눅스 시스템에서 사용하는 접근통제 정책 형태와 유사한 형태로 설계하였다. [그림 1]은 기술언어를 간단하게 나타낸 것이다. subject 항목에는 접근 권한을 허용할 프로그램의 주체 타입 이름을 설정한다. 그리고 object 항목에는 주체가 허용할 객체들과 객체의 타입 이름을 추가한다. 그리고 permission 항목에는 주체와 객체 타입의 권한관계를 설정한다. 예를 들어 passwd 명령어에 대한 보안

```
subject:
    subject_type        [option]
object:
    object_type         { path_name | value }
permission:
    object_alias object_type { operations, ... }
```

〔그림 1〕 보안 정책 템플릿 기술언어

```
subject:
    passwd binary
object:
    passwd_bin { /usr/bin/passwd }
    passwd_file { /etc/passwd*, /etc/shadow* }
permission:
    passwd_bin file { execute }
    passwd_file file { read, write }
```

〔그림 2〕 보안 정책 템플릿 passwd 명령어 예제

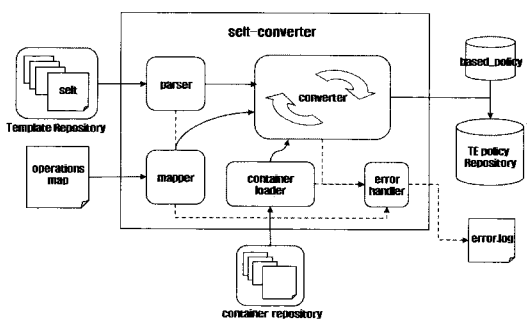
정책을 설정하기 위해서는 [그림 2]와 같이하면 된다. 설정 후 적용한 이후에는 /usr/bin/passwd를 실행할 경우에 passwd_t 주체 타입이 passwd 프로세스에게 부여되고 passwd 프로세스는 지정된 /etc/passwd 파일과/etc/shadow 파일만을 읽고 쓸 수 있게 된다[11,12].

4.3 보안 정책 템플릿 응용 도구

이 절에서는 SELT 언어로 작성된 정책을 SELinux 정책으로 변환하는 보안 정책 템플릿 변환기와 변환된 정책을 SELinux에 적용하는 집행기 그리고 자동으로 대상 프로그램의 정보흐름을 분석하여 자동으로 보안 정책 템플릿을 생성하는 생성기에 대해서 설명한다.

4.3.1 보안 정책 템플릿 변환기

보안 정책 템플릿 변환기(selt-converter)는 [그림 3]과 같이 변환기(converter)와 매퍼(mapper), 파서(parser) 등으로 구성된다. 매퍼는 보안 정책 템플릿의 객체와 권한을 SELinux의 객체 및 권한으로 매핑 시켜 놓은 operations-map을 파싱하여 자료구조화 하고 변환기에 전달하여 주는 기능을 한다. 파서는 기술언어 형태로 정의된 보안 정책 템플릿 파일을 읽어서 자료구조로 만들어 변환기에 전달한다. 변환기는 매퍼와 파서에서 받은 정보와 컨테이너(container) 저장소의 정보 및 기본 정책(based_policy)를 기반으로 하여 SELinux 보안 정책을 생성한다. 각 모듈에서 발생하는 에러는 에러 처리기가 수집하여 로그로 기록하여 디버깅 정보로 활용할 수 있다. [표 6]은 SELT 템플릿 변환기의 주요 구성 요소에 대한 설명이다[13].



[그림 3] 보안 정책 템플릿 변환기 구조

[표 6] 보안 정책 템플릿 변환기 주요 구성 요소

파 일	설 명
based_policy	초기 시스템 상태를 유지하기 위한 최소한의 권한을 포함하고 있는 SELinux 보안 정책
selt	보안 정책 템플릿 파일로 변환될 보안 정책 파일
container	컨테이너의 역할은 규칙 템플릿 변환 과정에서 반복적으로 자주 쓰이거나 그룹화 시킬 필요가 있는 TE 정책을 M4 매크로의 형태로 이용하기 위해서 개발
operations-map	SELinux 권한과 보안 정책 템플릿 권한 매핑 파일

[표 7] 보안 정책 템플릿 변환기 주요 모듈

모 듈	설 명
converter_domain_declaration( )	도메인 생성 함수로서, 해당 도메인을 정의 하고 타입을 결정
converter_role_assignment( )	역할 할당 함수로서 템플릿으로 부터 주체타입을 가져와 기본적으로system_r에 할당
converter_type_declaration( )	타입생성 함수로서 객체 타입과 속성을 가져와 te 형태로 변환
converter_transition_declaration( )	도메인 전이 설정 함수로서 템플릿으로 부터 전이 값을 가져와 te파일로 변환
converter_permission_declaration( )	권한 생성 함수로서 템플릿에 표현된 권한을 바탕으로 te형태로 변환
converter_filecontext_declaration( )	파일 보안 컨텍스트 생성 함수로서템플릿에 표현된 컨텍스트를 fc형태로 변환

SELinux 정책 템플릿 변환기는 SELinux 정책 생성 절차에 따라 크게 6개의 모듈로 구성된다. 각각의 모듈은 템플릿으로 부터 획득한 자료를 바탕으로 SELinux의 '*.te'와 '*.fc' 파일로 생성하는데 주요 역할을 하는 모듈들이다. [표 7]에서 보는 바와 같이 각각의 함수들이 하는 역할들이 정의 되어있다. 해당 함수들에 의해서 SELinux형태로 변환된다.

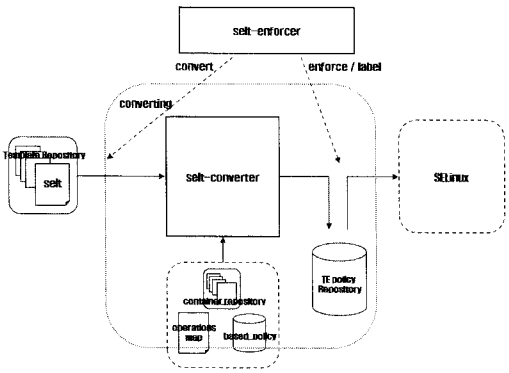
4.3.2 보안 정책 템플릿 집행기

보안 정책 템플릿 집행기(selt-enforcer)는 [그림 4]와 같이 보안 정책 템플릿 도구들을 관리하는 역할과 변환기에 의해 변환되어 생성된 SELinux 보안 정책을 시스템에 적용하는 역할을 수행한다. 보안 정책 관리자가 집행기를 실행시키면 집행기는 변환기는 실행하고 SELinux

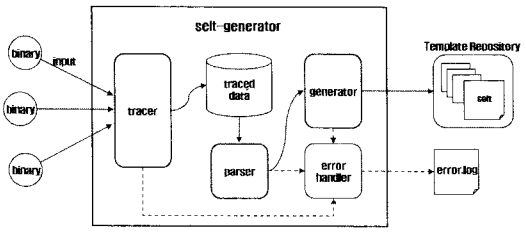
보안 정책을 생성한다. 그리고 SELinux의 유틸리티인 checkpolicy, setfiles 그리고 set · getenforce 도구들을 실행하여 변환된 SELinux 보안 정책을 컴파일하고 컴파일된 바이너리 보안 정책을 적재하여 시스템에 적용 시킨다.

4.3.3 보안 정책 템플릿 생성기

보안 정책 템플릿 생성기(selt-generator)는 프로세스 행위 추적(trace)을 통해 자동적으로 보안 정책 템플릿 파일을 생성하는 것을 목표로 한다. 보안 정책을 설정할 프로그램이 접근하는 객체들을 분석하고 객체별로 권한을 추적하여 얻어진 정보를 정규화하고 최종적으로 보안 정책 템플릿 기술언어 형태의 보안 정책 템플릿 파일을 생성한다[14].



〔그림 4〕 보안 정책 템플릿 집행기 구조



〔그림 5〕 보안 정책 템플릿 생성기 구조

[그림 5]는 보안 정책 템플릿 생성기의 동작 구조이다. seltgen이라는 명령어를 통해서 실행되며 사용자로부터 실행파일을 입력받아 실행하고 추적하여 객체와 권한 데이터를 획득한다. 획득된 데이터는 파싱 과정을 통해 자료구조에 삽입되고 정규화 모델을 통해 정규화한다. 정규화된 자료구조는 보안 정책 템플릿 형태로 작성되어 파일로 저장된다.

4.4 보안 정책 템플릿 정책

이 절에서는 앞 절에서 설명한 보안 정책 템플릿 기술 언어와 응용 도구를 테스트하기 위해 주요 서비스 데몬에 대한 보안 정책 템플릿 중에서 가장 대표적인 서비스인 아파치(apache) 웹 서버 정책과 MySQL 데이터베이스 서버 정책에 대해 설명한다.

4.4.1 아파치 보안 정책 템플릿

아파치 웹 서버는 서비스 제공을 하기 위한 웹 페이지들에 해당하는 다양한 객체에 접근한다. 이것은 일반적인 데몬(daemon) 시스템으로써 시스템 내의 객체들인 /etc/passwd와 같은 객체들에 대해서도 접근한다. 이외에도 네트워크 객체로 80번 포트 또는 8080번 포트를 사용한다. 아파치 웹 서버가 정상적으로 서비스되기 위해서는 [표 8]과 같이 요구되는 모든 객체들에 대한 접근 권한이 설정되어야 한다.

〔표 8〕 아파치 데몬 객체 보안 정책

클래스	권한	객 체
file	execute	/usr/sbin/(httpd. httpd.worker) /etc/init.d/httpd /var/www/cgi-bin/*
	read	/var/www/html/* /usr/lib/httpd/modules/* /etc/httpd/conf/*
	write	/var/log/httpd/* /var/www/html/upload/*
dir	access	/var/www/html/
	view	/var/www/html/ /etc/httpd/conf/ /usr/lib/httpd/modules/
	create	/var/log/httpd/ /var/www/html/upload/
	remove	/var/www/html/upload/
protocol	allow	tcp
port	allow	80, 8080

4.4.2 MySQL 보안 정책 템플릿

MySQL 데이터베이스 서버의 객체 구조는 아파치

웹 서버보다 간단한 객체 구조이다. [표 9]와 같이 데이터 저장 공간과 설정 파일 그리고 3306 포트에 대한 접근 권한을 설정하면 된다.

〔표 9〕 MySQL 데이터베이스 서버의 정책

클래스	권한	객 체
file	execute	/usr/libexec/mysqld /etc/init.d/mysqld /usr/bin/mysqld_safe
	read	/var/www/html/* /usr/lib/httpd/modules/* /etc/my.cnf /var/lib/mysql/mysql.sock
	write	/var/lib/mysql/* /var/log/mysql.* /var/lib/mysql/mysql.sock
	remove	/var/run/mysqld/* /var/lib/mysql/mysql.sock
dir	access	/var/lib/mysql/
	view	/var/lib/mysql/
	create	/var/lib/log/ /var/lib/mysql/
	remove	/var/lib/mysql/
protocol	allow	tcp
port	allow	3306

V. 보안 정책 템플릿 분석 및 평가

이 장에서는 SELinux를 이용하였을 때와 본 논문에서 제안한 보안 정책 템플릿을 이용하였을 때의 복잡성을 분석하고 비교 평가한다.

5.1 실험 환경

본 실험에서는 페도라 리눅스 코어 4 배포판을 설치하고 기본으로 제공하고 있는 SELinux 보안 정책을 설치하고 적용하였다. 그리고 제안하는 도구인 보안 정책 템플릿 정책과 도구를 설치하였다. 실험 항목은 SELinux 기본 보안 정책인 지정 정책(targeted policy)에서 아파치 웹 서버와 MySQL 데이터베이스 서버에 설정한 보안 정책과 보안 정책 템플릿에서 동일한 데몬 들에 대한 보안 정책을 설정한 파일 간에 객체 타입과 권한의 개수를 비교 분석하여 복잡성을 분석하였다.

5.2 SELinux 구조 복잡성 분석

이 절에서는 SELinux와 보안 정책 템플릿에서 정의된 객체 클래스 종류와 권한의 개수를 비교 분석하고 2장에서 설명한 단순 정책 복잡성(SPC)와 전체 정책 복잡성(FPC)를 비교 분석하였다. [표 10]의 결과와 같이 본 논문에서 제안한 보안 정책 템플릿은 SELinux 복잡성을 크게 개선한 것을 볼 수 있다.

〔표 10〕 구조 비교 분석을 통한 SELinux 복잡성 개선 분석

구 분	SELinux 복잡성	SELT 구조 복잡성
객체 클래스 개수	56개	4개
권한 개수	212개	9개
단순 정책 복잡성	11,872	36
전체 정책 복잡성	$2^{212}-1$	511

5.3 SELinux 정책 복잡성 분석

이 절에서는 아파치 웹 서버, MySQL 데이터베이스 서버, named DNS 서버, 그리고 Sendmail 메일 서버에 설정한 SELinux 정책과 보안 정책 템플릿 정책을 비교하여 타입과 규칙 개수를 정량적으로 계산하여 보안 정책 복잡성 개선 정도를 분석하였다. [표 11]과 같이 제안한 보안 정책 템플릿은 SELinux 복잡성을 크게 개선한 것을 볼 수 있다.

〔표 11〕 정책 비교 분석을 통한 SELinux 복잡성 개선 분석

구 분	SELinux 정책		제안 보안 정책 템플릿	
	타입 개수	규칙 개수	타입 개수	규칙 개수
Apache	30개	285개	13개	41개
MySQL	8개	78개	10개	21개
Named	13개	145개	5개	11개
Sendmail	4개	80개	4개	18개

5.4 보안 정책 템플릿 자동 생성 충실도 분석

본 논문에서 제안한 보안 정책 템플릿 자동 생성기의 성능을 실험하기 위해서 수동으로 작성한 아파치 웹 서버와 MySQL 데이터베이스 서버의 보안 정책 템플릿 기준으로 자동으로 생성된 보안 정책 템플릿의 완성도를 분석하였다. 분석 결과 [표 12]와 같이 아파치 웹 서

버의 경우 타입 개수 61.5%와 규칙 개수 41.5%의 충실도를 보였다. 또한 MySQL 데이터베이스 서버의 경우 60%와 42.9%의 충실도를 보였다. 이점은 향후 보다 높은 충실도를 갖도록 추가 연구가 요구된다.

〔표 12〕 보안 정책 템플릿 자동 생성 충실도 분석

구 분	아파치 웹 서버	MySQL 데이터베이스 서버
타입 개수	8/13 (61.5%)	6/10 (60%)
규칙 개수	17/41 (41.5%)	9/21 (42.9%)

Ⅵ. 결 론

SELinux는 이미 많은 리눅스 시스템 사용자들에게 가장 널리 퍼져있는 보안 운영체제가 되었다. 그러나 SELinux가 갖고 있는 복잡성으로 인해 전문 지식이 없는 일반 사용자들에게 대중화되지는 못하고 있다. 이러한 SELinux 복잡성을 해결하기 위한 연구가 미국, 일본에서 진행 되고 있다. 따라서 국내에서도 보다 편리하게 SELinux를 사용할 수 있도록 하는 연구가 요구되고 있다.

본 논문에서 제안한 보안 정책 템플릿과 도구는 SELinux가 정의한 객체들을 추상화하여 최소한의 객체 타입으로 축약 하였다. 그리고 권한 간에 연관성을 분석하여 기존의 리눅스 시스템에 비슷한 형태로 축약하였다. 이처럼 축약된 객체 타입과 권한을 바탕으로 보안 정책 템플릿 기술언어를 개발하여 SELinux 보안 정책의 복잡성을 개선하였다. 또한 보안 정책 템플릿 기술언어로 설정된 보안 정책을 SELinux의 보안 정책으로 변환하는 변환기와 대상 프로그램의 정보흐름을 분석하여 자동으로 보안 정책 템플릿 정책을 생성하는 생성기를 개발하여 SELinux 사용에 있어서 편리성을 높였다.

제안한 보안 정책 템플릿과 SELinux 구조의 복잡도를 계산하고 보안 정책 템플릿의 정책과 SELinux 정책을 분석하여 SELinux 복잡성 개선도를 분석한 결과 SELinux 복잡성이 크게 개선된 것을 확인하였다. 또한 보안 정책 템플릿 자동 생성기를 통해 약 50% 이상의 정책을 자동으로 생성할 수 있었다. 본 연구의 결과로 SELinux 복잡성이 크게 개선됨으로써 향후 국내에서 SELinux 대중성에 기여할 것으로 예상된다. 현재 국내의 가장 대표적인 리눅스 시스템인 부요(booyo)에 적용되어 활용되고 있다.

참고문헌

[1] SELinux-Enhanced Linux, http://www.nsa.gov/research/selinux.

[2] Linux Kernel Archives, http://www.kernel.org.

[3] SELinux Wiki-Tresys Technology, http://www.selinuxproject.org/page/Main_Page.

[4] Daniel J Walsh, "SELinux Targeted vs Strict policy History and Strategy", SELinux Symposium, 2005.

[5] SETools Policy Analysis Suite, http://oss.tresys.com/projects/setools.

[6] SELinux Policy Editor, http://seedit.sourceforge.net.

[7] Yuichi Nakamura, "Simplifying Policy Management with SELinux Policy Editor", SELinux Symposium 2005.

[8] MITRE-Security-Enhanced Linux, http://www.mitre.org/tech/selinux.

[9] Amy L. Herzog, "Policy Analysis and Generation Work at MITRE", SELinux Symposiun, 2005.

[10] David R. Harris The MITRE Corporation "Guided Policy Generation for Application Authors" SELinux Symposium, March 2006.

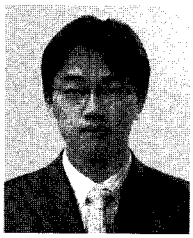
[11] 최민호, 김정순, 김민수, 노봉남, "SELinux 정책 템플릿 기술언어 설계", 정보처리학회 춘계학술발표대회, 2007.

[12] 정종민, 김정순, 김민수, 정성인, 노봉남, "SELinux 정책 복잡성 개선을 위한 보안정책 템플릿", 정보보호학회, 2006.

[13] 박호준, 이재서, 김정순, 김민수, 노봉남, "SELinux 정책 템플릿 변환기", 한국정보처리학회, 제14권 1호, 2007.

[14] 박호준, 정종민, 김정순, 김민수, 정성인, 노봉남, "SELinux를 쉽게 사용하기 위한 정책 자동 생성기에 관한 연구", 한국정보보호학회 학술발표회논문집, 2007.

〈著者紹介〉



이 재 서 (Jaeseo Lee)
학생회원

2004년 2월: 전남대학교 컴퓨터정보학부 졸업
2006년 2월: 전남대학교 정보보호협동과정 석사
2006년 3월~현재: 전남대학교 정보보호협동과정 박사과정

김 민 수 (Minsoo Kim)
종신회원

1993년 2월: 전남대학교 전산통계학과 졸업
1995년 2월: 전남대학교 전산통계학과 석사
2000년 2월: 전남대학교 전산통계학과 박사
2000년~2001년: 한국정보보호진흥원 선임연구원
2001년~2004년: 전남대학교 연구교수
2005년~현재: 목포대학교 정보보호학과 조교수
<관심분야> 침입탐지, 컴퓨터 포렌직스, 보안 운영체제, 데이터마이닝 등

노 봉 남 (Bong-Nam Noh)
종신회원

1978년 2월: 전남대학교 수학교육과 졸업(학사)
1982년 2월: KAIST 전산학과 졸업(석사)
1994년 2월: 전북대학교 대학원 전산과 졸업(박사)
1983년~현재: 전남대학교 전자컴퓨터공학부 교수
2000년~현재: 전남대학교 시스템보안연구센터 소장
<관심분야> 컴퓨터와 네트워크 보안, 개인정보보호, 사이버사회와 윤리