19 - 1 - R01

논리를 기반한 접근제어 시스템

A Short Tutorial on Logical Foundation of Access Control

신승철
동양대학교 컴퓨터학부
shin@dyu.ac.kr

요 약

컴퓨터 보안 메카니즘의 하나인 접근제어시스템을 알고리즘 기반 방식이 아닌 논리 기반 방식으로 다루는 방법에 대하여 설명한다. 분산시스템을 위한 일반화된 접근제어시스템은 매우 복잡한 질의를 처리해야 하기 때문에 알고리즘 기반 시스템으로 구성하기에는 올바름과 완전함을 보장하기 어렵다. 논리 기반 시스템은 대수 시스템과 양상논리를 이용하여 올바르고 완전한 접근제어시스템을 구현하는 데에 유용하다.

1 서 론

보안은 외부의 공격으로부터 개인이나 재산 및 단체를 보호하는 일반적인 용어이지만 컴퓨터 보안은 외부의 공격으로부터 시스템이나 정보 또는 서비스 자체를 보호하는 것이다. 외부의 공격으로부터 일어날 수 있는 피해는 비밀정보의 유출, 중요 정보의 손상, 중대한 서비스의 중단, 물리적인 자원의 부당한 접근, 금전의 도난, 시스템의 위법적인 사용 등이 있으며 이를 막거나 적어도 최소화해야 한다.

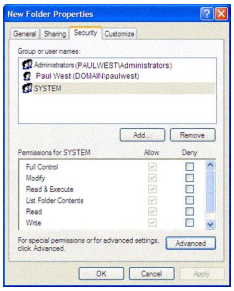
보안 문제를 바라보는 시각은 두 가지로 나누어 생각할 수 있다. 하나는 무엇을 어떤 것으로부터 보호할 것인가이고 또 하나는 어떻게 보호할 것인가이다. 전자를 보안 정책이라고 하고 후자를 보안 메카니즘이라고 한다. 보안 정책은 기밀성(confidentiality), 무결성(integrity), 가용성(availability), 사적 자유(privacy), 익명성(anonymity), 부인 방지(no repudiation) 등의 보안 성질을 표현한 것이다. 특히 기밀성과 무결성, 가용성은 보안 정책의 기본이 되는 성질이다. 기밀성은 비밀 정보에 대한 원하지 않는 유출이 금지되는 성질이고 무결성은 중요한 정보가 외부로부터의 간섭으로 손상되지 않는 것이다. 가용성은 정보에 대한 서비스가 통제된 방법으로 항상 제공되는 것을 의미한다.

보안 성질들이 조합된 보안 정책이 정해지면 이를 보장하는 방법이 필요하다. 컴퓨터 시스템은 공격받기 쉬운 취약점을 가질 수 있기 때문에 보안정책을 보장하기 위한 인증(authentication), 인가(authorization), 감사(auditing)와 같은 보안 메카니즘을 동원해야 한다. 소프트웨어 보안은 컴퓨터 시스템을 보호하기 위해 어떻게 보안 메카니즘을 소프트웨어로 구현해야 하는가 그리고 어떤 소프트웨어가 특정 보안 정책을 위배하는지를 어떻게 알 수 있는가와 같은 문제를 다룬다. 인증은 비밀번호, 생체인식, 인증서 등을 이용하여 해당 요구의 주체가 누구인가를 확인하고 인가(또는 접근제어)는 사용자가 수행하고자 하는 요구가 그 사용자에게 허가되었는지를 확인한다. 감사는 사후 단속을 위해서 누가 언제 무엇을 했는지에 대한 정보를 모으고 분석한다.

여기서는 보안 메카니즘 중에서 접근제어의 기본 개념을 바탕으로 양상 논리가 어떻게 적용될 수 있는가를 설명한다. 논리를 기반으로 하는 접근제어시스템은 양상논리와 주체 언어를 이용함으로써 올바르고(sound) 완전한(complete) 시스템을 구현할 수 있다.

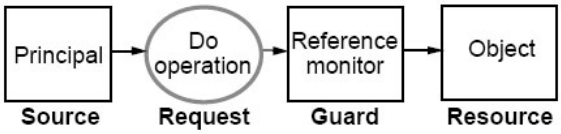
2 접근제어시스템

접근제어시스템는 사용자 P가 행위 A를 수행하도록 인가되었는가를 확인한다. 예를 들면 그림 1에서와 같이 운영체제는 ACL(Access Control List)를 통해 화일 접근 허가를 중재하는 시스템을 가지고 있다.



[그림 1] Windows ACL

일반적인 접근제어 모델은 네 가지 구성요소로 이루어지는데, 접근의 목표가 되는 자원이나 대상(object), 접근 내용인 행위의 요구(request), 행위의 요구를 내는 사용자 주체(principal), 요구에 대한 허용 여부를 결정하는 참조 모니터(reference monitor)이다.



[그림 2] Access control model

접근제어시스템은 인증시스템과 구별되는 데, 접근제어는 주체 P가 요구 R에 대하여 기대할 수 있는가 다시 말해서, 사용자 P가 R을 요구하면 R이 주어져야 하는가에 대한 대답을 제공하고 인증은 누가 R을 요구했는가 또는 사용자 P는 누구의 위임을 받았는가에 대한 대답을 제공한다.

이를 위해서 접근제어 기준이 되는 그림 3과 같은 접근제어 행렬을 이용한다. 접근제어 행렬을 구현하기 위해 현재까지 많이 사용되어 온 두 가지 방법은 ACL과 Capability이다. ACL은 접근제어 행렬의 열들로 이루어지며 주어진 접근 대상에 대하여 각 주체들에 허용되는 요구들을 나타낸다. Capability는 접근제어 행렬의 행들로 구성되며 특정 주체에게 허용되는 각 대상에 대한 요구들을 나타낸다.

접근제어시스템는 완전한 중재 원칙(principle of complete mediation)을 구현해야 한다. 즉, 모든 대상에 대한 모든 접근이 검사되어야 한다. 이 원칙은 여러가지 방법으로 구현될 수 있는데, 사용자의 요구 중에서 일부는 Unix 같은 운영체제가 처리하고 나머지는 하드웨어가 처리하는 식으로 이루어지거나 JVM과 같은 소프트웨어 가상기계가 사용자 요구의 일부를 처리한다.

접근제어는 주체의 결합(conjunction), 그룹, 역할(role), 위임(delegation) 등에 의해서 복잡해질 수 있다. 때로는 여러 주체가 함께 하나의 접근 요구를 내야만 그 요구가 허용되는

objects / principals	file1	file2	file3	file4
user1	rwx	rw	r	x
user2	r	r		x
user3	r	r		x

[그림 3] Access control matrix

경우가 있는데 이런 경우에 주체 결합이라고 한다. 주체 결합은 접근제어 정책 내에 명시적으로 또는 암시적으로 나타날 수 있다. 주체는 그룹을 구성할 수도 있고 특정 역할을 대신할 수도 있다. 그룹과 역할은 간접적인 접근을 나타내는 수단이다.

또한 접근 대상이나 행위 요구도 그룹을 구성할 수도 있다. 회사내의 모든 화일을 나타내는 대상 그룹이나 하나의 대상에 대한 모든 read 행위들을 나타내는 요구 그룹도 생각할 수 있다. 게다가 때로는 요구들은 대상과 결합된 형태로 나타난다. 예들 들면 '환자의 기록 읽기'나 '로그 기록을 쓰기'와 같은 요구들은 접근대상과 애초에 결합되어 나타나기도 한다.

프로그램 자체도 주체가 될 수 있다. 이런 경우에 프로그램 호출 시퀀스를 관리할 필요가 있다. 예를 들면 운영체제가 웹브라우저를 호출하고 웹브라우저는 애플릿을 호출하는 경우에 애플릿이 어떤 대상에 접근할 때 접근 주체는 애플릿이 아니라 호출시퀀스의 모든 프로그램이 주체가 되어야 한다. 게다가 프로그램들과 다른 주체와의 관계를 다룰 수 있어야 한다. 프로그램을 작성한 주체, 이를 제공한 주체, 설치한 주체, 실행시킨 주체가 모두 이 프로그램에 의한 접근 요구와 연관된다.

접근제어는 상당히 넓은 분야에 존재한다. 응용 프로그램은 물론이고 가상기계, 운영체제, 방화벽, 게이트웨이 등과 결합되는 접근제어를 생각할 수 있다. 접근제어를 올바르게 구현하는 것은 결코 쉽지 않으며 분산시스템 등을 고려하면 상황은 훨씬 더 복잡하고 해결하기 어려워진다.

오랫동안 접근제어를 위한 많은 이론과 시스템이 제안되어왔다. 그것들은 논리, 언어, 인프라스트럭처, 아키텍처 등을 기반하는 형태로 등장하였으며 대부분 접근제어를 설명하고 구성하여 통합시킬 목적으로 제안되었다. 이들은 이론적으로 매우 만족스러울 뿐아니라 실제로도 유용한 결과를 가져왔다.

Harrison 등[1]으로부터 시작된 알고리즘 기반 방식의 접근제어시스템은 명령과 권한에 대한 유한집합을 이용하여 접근제어 행렬을 구성한다. 명령은 '조건이 만족되면 연산을 수행하라'와 같은 형태이다. 여기서 조건은 행렬에 대한 술어이고 연산은 권한이나 대상 또는 주체를 추가하거나 삭제한다. 다음은 화일의 소유자가 다른 사용자에게 읽기 권한을 부여하는 연산을 보여준다:

```
COMMAND CONFER_r (owner, friend, file)
    if own in (owner, file)
    then enter r in (friend, file)
END
```

접근제어시스템의 안전성(safety) 보장은 신뢰할 수 없는 주체가 모든 가능한 상태에서 허용되지 않는 대상을 접근할 수 없다는 것을 의미하는데 이를 확인하는 일은 일반적으로 결정불가능하다. 그러나 모든 접근제어 문제가 결정불가능한 것은 아니다. Li 등이 제안한 문제[3]는 '모든 가능한 상태에서 모든 주체가 하나의 성질을 만족하면 다른 성질을 반드시 만족하는가'와 같이 포함관계를 다룬다. 이를 이용하여 특정 대상에 접근하는 주체가 어떤

성질을 항상 만족하는지를 확인할 수 있으며 다양한 형태의 시스템에서 이 문제는 결정가능하다.

정형적으로 검증된 커널 시스템이야말로 적어도 빠른 시간내에 보안 시스템을 구성하는 가장 유망한 기반이 될 것이라고 생각되었다. 그러나 정형화를 통해 제안된 여러가지 시스템과 모델이 여전히 실용적인 측면에서는 어려움과 문제점을 드러내고 있기도 하다. 예를 들어 접근제어 행렬을 논리를 이용해서 표현하는 방식을 생각해 보자. 접근제어 행렬은 삼항 술어 기호 *may-access*로 표현될 수 있다. 즉, Alice가 Foo.txt를 읽을 권한이 있음을 나타내는 may-access(Alice, Foo.txt, Read) 와 같은 공리와 쓰기 권한은 읽기 권한을 포함하고 있음을 표현하는 may-access(p, o, Write) ⇒ may-access(p, o, Read) 형태의 추론규칙을 나타낼 수 있다. 이 방식은 고전논리의 술어논리를 완전하게 재현한다고 할 수 있다. 여기서 이러한 방식이 보안 정책을 표현하고 그 보장을 확인하는 데에 유용한가를 생각해보자. 다양한 보안정책을 표현하기 위해서는 더 많은 술어와 공리가 필요할 수 있다. 예를 들면, may-jointly-access(p_1, p_2, o, r), owns(p, o), ... 유한한 술어와 공리로는 무한히 많은 보안 정책을 표현하는 것이 불가능하다.

게다가 분산시스템에서의 접근제어는 문제를 더욱 어렵게 만드는 요소들이 산재하다. 예를 들면, 시스템의 크기, 시스템 동작의 고장, 이질적인 통신 채널, 자율성으로인한 중앙관리의 부재 등이다.

3 기본양상논리

Abadi 등[5]은 접근제어를 위한 새로운 논리기반 방식을 제안하였는데 (여기서는 이를 접근제어 논리'라고 부르자)기존의 술어논리와는 달리 주체 언어(calculus of principals)를 기반으로 그 위에 접근 제어를 위한 다중 양상논리(multi-modal logic)를 정의함으로써 보다 일반적인 접근제어를 표현하고 검증하는 시스템을 구성하였다. 접근제어 논리의 언어는 두 계층을 갖는다. 하나는 주체를 표현하는 주체식(principal expression)으로 이루어진 주체 언어이고 또 하나는 주체식을 기반하는 다중양상논리식으로 구성되는 양상논리이다.

주체 언어는 주체와 주체의 요구를 나타내기 위한 표기로서 접근대상이나 연산, 신뢰성, 채널 등을 표현할 수 있다. 양상논리는 접근제어에 대한 추리를 담당하는 논리이다. 먼저 양상논리를 형식적으로 설명하고난 후에 주체언어를 포함하도록 확장하기로 하자. 기본 양상 논리식을 생성하는 기호들의 집합은 다음 집합들의 합집합이다:

- 명제 기호의 집합 $= \{p, q, ...\}$
- 명제 논리 연결자의 집합 $= \{\neg, \wedge, \top\}$
- 양상 연산자의 집합 $= \{\Box\}$

명제 논리에서와 같이 $\vee$ 또는 $\rightarrow$, $\leftrightarrow$, $\bot$ 등은 위의 명제 논리 연결자들로부터 정의될 수 있다. 또한 양상 연산자 $\Diamond$도 $\Box$로부터 정의할 수 있으므로 명시적으로 포함시키지 않았다. 기호들의 집합으로부터 얻어지는 모든 문자열 중에서 다음의 규칙을 따르는 것들이 기본 양상 논리의 언어를 구성하는 잘-정의된(well-formed) 논리식들이다:

$$\phi \ ::= \ p \ | \ \top \ | \ \neg\phi \ | \ \phi \wedge \psi \ | \ \Box\phi$$

여기서 ϕ와 ψ는 기본 양상 논리식을 나타내고, 양상 연산자는 단항 연산자로서 $\neg$과 같은 연산 순위를 가진다.

양상 논리식의 의미구조는 S. Kripke가 제안한 가능 세계(possible worlds) 관계 구조(relational structure)가 가장 널리 받아들여지고 있다. 양상 논리의 가능 세계 의미구조는 크게 골조(frame)와 모델(model)로 나누어 설명할 수 있다.

기본 양상 논리를 위한 골조는 $\mathfrak{F} = (W, R)$ 로 나타내어지고 여기서 W는 모든 가능 세계들의 집합으로 공집합이 아니고, R은 $W \times W$의 부분집합인 이항 관계이다. 또한 기본

양상 논리에 대한 모델은 골조 $\mathfrak{F}$를 이용하여 $\mathfrak{M} = (\mathfrak{F}, V)$ 와 같이 나타내어지고, 여기서 $V : \Phi \to \mathcal{P}(W)$는 평가 함수로서 기본 양상 논리식이 주어지면 주어진 논리식이 참인 가능 세계들의 집합을 건네준다.

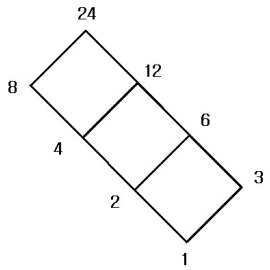
이렇게 기본 양상 논리의 의미구조가 결정되면 만족 관계(satisfaction relation) $\vDash$를 정의할 수 있는데, $\mathfrak{M}, w \vDash \phi$ 는 주어진 모델 $\mathfrak{M}$의 가능 세계 w에서 기본 양상 논리식 ϕ가 참임을 의미한다. 주어진 모델과 가능 세계에서 주어진 기본 양상 논리식이 참이 되는 조건은 다음과 같다.

$$\begin{aligned}
\mathfrak{M}, w \vDash p &\quad \text{if} \quad w \in V(p), \text{ where } p \in \Phi \\
\mathfrak{M}, w \vDash \top &\quad \text{if} \quad \text{always} \\
\mathfrak{M}, w \vDash \neg\phi &\quad \text{if} \quad \mathfrak{M}, w \nvDash \phi \\
\mathfrak{M}, w \vDash \phi \wedge \psi &\quad \text{if} \quad \mathfrak{M}, w \vDash \phi \quad \text{and} \quad \mathfrak{M}, w \vDash \psi \\
\mathfrak{M}, w \vDash \Box\phi &\quad \text{if} \quad \text{for every } v \in W \text{ such that } Rwv, \text{ we have } \mathfrak{M}, v \vDash \phi
\end{aligned}$$

[**예제 1**] 그림 4는 다음의 골조 $\mathfrak{F}$를 Hasse 다이어그램으로 나타낸 것이다.

$$\mathfrak{F} = (\{1, 2, 3, 4, 6, 8, 12, 24\}, R) \quad \text{where } Rxy \text{ iff } x \neq y \text{ and } x \text{ divides } y$$

여기에 평가 함수 $V(p) = \{4, 8, 12, 24\}$, $V(q) = \{6\}$를 제공하자. 그러면 다음이 참임을 알



[그림 4] lattice

수 있다.

(1) $\mathfrak{M}, 2 \nvDash \Box p$

(2) $\mathfrak{M}, 2 \vDash \Diamond(q \wedge \Box p) \wedge \Diamond(\neg q \wedge \Box p)$

기본 양상 논리를 대상으로 하는 증명 체계 **K**의 공리들은 다음으로 구성된다:

- 명제 논리의 모든 항진 논리식
- $\Box(p \to q) \to (\Box p \to \Box q)$

또한 **K**의 증명 규칙은 다음과 같다:

- 긍정 추론(modus ponens): ϕ와 $\phi \to \psi$이 주어지면 ψ는 증명된다.
- 일률적인 치환(uniform substitution): ϕ가 주어지면, ϕ 안의 명제 기호들을 임의의 논리식으로 치환하여 얻어진 θ는 증명된다.
- 필연 추론(necessitation): ϕ가 주어지면 $\Box\phi$는 증명된다.

K-증명은 논리식들의 유한한 시퀀스로 구성되고 각 논리식들은 공리이거나 시퀀스의 앞쪽에 등장한 하나 이상의 논리식으로부터 증명 규칙을 적용하여 얻어낸 논리식이다. 이것은 힐버트 방식의 증명인데, 논리식 ϕ가 어떤 **K**-증명의 맨 마지막 항목으로 나타나면 논리식 ϕ는 **K**-증명 가능하다고 하고 $\vdash_{\mathbf{K}} \phi$와 같이 표시한다.

모든 **K**-증명가능한 논리식들이 모든 골조의 집합에 대하여 타당하면 **K**는 올바르다(sound)고 한다. 또한 모든 골조들의 집합에 대하여 타당한 논리식들이 모두 **K**-증명가능하면 **K**는 완전하다(complete)고 한다.

4 접근제어논리

이제 접근제어시스템에 적용할 다중양상논리를 정의해 보자. 논리식을 생성하는 기호들의 집합은 양상연산자 $\Box$가 다중양상 $\Box_A$로 바뀐 것을 제외하고는 기본 양상논리와 같다. 따라서 잘 정의된 논리식은 다음의 문법을 따른다. ψ :

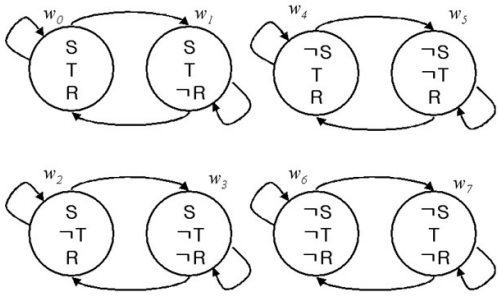
$$p \mid \top \mid \neg\phi \mid \phi \wedge \psi \mid \Box_A \phi$$

다중양상 $\Box_A\phi$ 는 때론 $[A]\phi$와 같이 표시되기도 하는데 'A believes ϕ' 또는 'A says ϕ'(A가 ϕ를 요구하다)를 의미한다. 여기서 A는 어떤 인덱스집합의 원소이다. 앞으로 접근제어를 구체적으로 나타내는 다중양상연산은 $\Box_A\phi$ 대신에 'A says ϕ'를 쓰도록 한다.

다중양상논리를 위한 골조는 다중관계구조로 나타나는데, $\mathfrak{F} = (W, \{R_A\})$ 와 같다. 마찬가지로 W는 가능세계 집합이고 R_A 는 W에서 정의된 이항 관계로서 인덱스집합 P의 원소 A와 결합되어 있다. 다중양상논리의 모델 $\mathfrak{M} = (\mathfrak{F}, V)$ 의 형태이고 V는 평가 함수 $\Phi \to \mathcal{P}(W)$ 이다. 다중양상논리의 만족관계는 다시 다음과 같이 정의된다.

$$\begin{aligned}
\mathfrak{M}, w \vDash p &\quad \text{if} \quad w \in V(p), \text{ where } p \in \Phi \\
\mathfrak{M}, w \vDash \top &\quad \text{if} \quad \text{always} \\
\mathfrak{M}, w \vDash \neg\phi &\quad \text{if} \quad \mathfrak{M}, w \nvDash \phi \\
\mathfrak{M}, w \vDash \phi \wedge \psi &\quad \text{if} \quad \mathfrak{M}, w \vDash \phi \quad \text{and} \quad \mathfrak{M}, w \vDash \psi \\
\mathfrak{M}, w \vDash \Box_A \phi &\quad \text{if} \quad \text{for every } v \in W \text{ such that } R_A w v, \text{ we have } \mathfrak{M}, v \vDash \phi
\end{aligned}$$

[**예제 2**] 그림 5는 8개의 가능세계를 가진 모델을 나타낸다. 각 명제 기호에 대한 문장은 아래에 나타나있다. 여기에는 단 하나의 이항관계만 존재하는데 R_A이다.



[그림 5] 8개의 가능세계 모델

S: Alice is in summer school

¬S: Alice is in Hae-un-dae
T: Talk of summer school is funny
¬T: Talk of summer school is boring
R: It is raining in Hae-un-dae
¬R: It is sunny in Hae-un-dae

이 모델에서 다음의 만족관계가 성립한다.

$$\mathfrak{M}, w_0 \vDash A \text{ says } T$$
$$\mathfrak{M}, w_0 \nvDash A \text{ says } R$$
$$\mathfrak{M}, w_0 \nvDash A \text{ says } \neg R$$

여기서 R_A는 Alice가 알 수 있는 일을 나타내고 있다고 직관적으로 생각할 수 있다. 따라서 Alice가 w_0의 세계에 있다면 R_A에 의해서 Alice가 여름학교에 있고 강의가 재미있는 상황이다. 하지만 해운대에 비가 오는 지 아닌지는 알 수 없는 상황이기 때문에 말할 수 없는 것이다.

지금까지 보여준 다중양상논리에서 다중양상연산자에 결합되는 인덱스가 접근제어의 주체에 대응된다는 것을 쉽게 짐작할 수 있다. 이제 접근제어의 주체가 합성된 형태의 주체가 될 수 있는 경우로 확장해 보자. 그러면 주체는 하나의 사용자만을 지칭하는 것이 아니라 여러 사용자들이 혼합되어 가상의 사용자 주체를 형성하는 경우를 포함할 수 있게 되고 이때 주체는 주체식이 된다. 인덱스 집합의 원소 $A \in P$와 주체식 $\mathcal{A}, \mathcal{B}$에 대하여 주체식은 귀납적으로 다음과 같이 정의된다:

- primitive principals $A \in P$
- $\mathcal{A} + \mathcal{B}$
- $\mathcal{A}|\mathcal{B}$

주체식에 대한 의미구조는 의미 함수 $\mathcal{R} : P \to \mathcal{P}(W \times W)$에 의해 정의되는데 다음과 같다:

- $\mathcal{R}(A) = R_A$
- $\mathcal{R}(\mathcal{A} + \mathcal{B}) = \mathcal{R}(\mathcal{A}) \cup \mathcal{R}(\mathcal{B})$
- $\mathcal{R}(\mathcal{A}|\mathcal{B}) = \mathcal{R}(\mathcal{B}) \circ \mathcal{R}(\mathcal{A})$

주체식을 포함하는 다중양상논리식의 만족관계는 쉽게 유추할 수 있다. $\mathcal{B}|\mathcal{A}$을 포함하는 경우만을 생각해 보자. 만족관계 $\mathfrak{M}, w_0 \vDash \mathcal{B}|\mathcal{A}$ says ϕ 는 다음과 같이 정의된다:

$$\mathfrak{M}, w \vDash \phi \text{ for all } w \text{ such that } w \in \bigcup_{w' \in \mathcal{R}(B)(w_0)} \mathcal{R}(A)(w')$$

위와 같이 정의된 주체식으로 이루어진 주체 언어는 대수 구조상으로 곱셈의 세미래티스 세미그룹(multiplicative semilattice semigroup)이다. 또한 이것은 합집합과 합성관계를 가진 이항관계의 대수로 표현이 가능하고 이를 이용한 의미구조 설명이 가능하다. 하지만 타당한 논리식의 집합이 재귀적이지 않기 때문에 논리식의 타당성 검증은 결정불가능할 수 있으므로 논리식의 형태를 제한함으로서 결정가능하고 다루기 쉬운 정도의 접근제어 논리를 구성할 수 있다. 여기서는 이에 대한 논의를 생략하기로 한다. 관심있는 독자는 [5]를 참조하기 바란다.

다중양상논리를 위한 증명체계 $\mathbf{K}_n$를 생각해 보자. $\mathbf{K}_n$의 공리들은 다음과 같다:

- propositional tautologies
- A says $(\phi \to \psi) \to (A \text{ says } \phi \to A \text{ says } \psi)$

$\mathbf{K}_n$는 다음의 증명 규칙을 가진다:

- 긍정 추론(modus ponens): ϕ와 $\phi \rightarrow \psi$이 주어지면 ψ는 증명된다.
- 일률적인 치환(uniform substitution): ϕ가 주어지면, ϕ 안의 명제 기호들을 임의의 논리식으로 치환하여 얻어진 θ는 증명된다.
- 필연 추론(necessitation): ϕ가 주어지면 $\Box\phi$는 증명된다.

여기에 주체 언어가 포함된다. 다음의 공리를 포함하고
- $(A + B) + C = A + (B + C)$
- $A + B = B + A$
- $A + A = A$

아래의 대수규칙을 포함시키면
- $(A|B)|C = A|(B|C)$
- $A|(B + C) = (A|B) + (A|C)$
- $(A + B)|C = (A|C) + (B|C)$

주체 언어가 가지는 타당한 주체식들을 접근제어 논리에 포함시킬 수 있다. 또한 주체식을 포함하는 다음의 다중양상 논리식을 공리로서 포함시키면 주체언어와 다중양상논리가 연결되어 접근제어를 위한 논리가 만들어진다.
- $(A + B)$ says $\phi \equiv (A$ says $\phi) \wedge (B$ says $\phi)$
- $(B|A)$ says $\phi \equiv B$ says $(A$ says $\phi)$

5 접근제어논리의 확장

주체식의 'speaks for' 관계를 포함하도록 접근제어 논리를 확장하여 보자. 'B speaks for A'는 B가 A를 대신한다는 의미로서 $B \Rightarrow A$ 와 같이 표기하고 $(B \Rightarrow A) \equiv (B = B + A)$ 와 같이 정의된다. 추가적으로 다음의 공리들을 포함한다:
- $(A \Rightarrow B) \rightarrow ((A + C) \Rightarrow (B + C))$
- $(A \Rightarrow B) \rightarrow ((A|C) \Rightarrow (B|C))$
- $(A \Rightarrow B) \rightarrow ((C|A) \Rightarrow (C|B))$

그러면 다음의 정리를 얻을 수 있다:
- $(B \Rightarrow A) \rightarrow ((B$ says $\phi) \rightarrow (A$ says $\phi))$
- $(B \Rightarrow A) \wedge (C \Rightarrow B) \rightarrow C \Rightarrow A$

의미구조 상에서 'speaks for'의 의미를 만족관계를 이용하여 나타내면 다음과 같다.

$$\begin{aligned} & \mathfrak{M}, w \vDash B \Rightarrow A \\ \text{iff} \quad & \mathcal{R}(B) = \mathcal{R}(B + A) = \mathcal{R}(B) \cup \mathcal{R}(A) \\ \text{iff} \quad & \mathcal{R}(A) \subseteq \mathcal{R}(B) \end{aligned}$$

여기서 'speaks for'는 가능세계 w와 무관하게 그 의미를 결정함을 알 수 있다. 따라서 다음이 성립한다.

$$\mathfrak{M} \vDash B \Rightarrow A \quad \text{iff} \quad \mathcal{R}(A) \subseteq \mathcal{R}(B)$$

다음으로는 정의된 접근제어 논리를 이용하여 ACL을 구현하는 방법에 대하여 예제를 통해 설명한다. 이를 위해 새로운 유도형 연결자 'controls'을 다음과 같이 정의한다:

$$A \text{ controls } s \quad \equiv \quad (A \text{ says } s) \rightarrow s$$

이는 A에게는 s에 대한 권한이 있다는 것을 의미한다. 이를 이용하여 ACL을 접근제어 논리식으로 구성하는 것이 가능하다.

[**예제 3**] 다음 사항을 가정하자:
- 명제 s_{write}: 화일 O_1을 쓰는 것이 허용된다.
- $B \Rightarrow A$
- A controls s_{write}

그러면 우리에게 B says s_{write}이 주어지면 접근제어 논리의 추론으로부터 '화일 O_1을 쓰는 것이 허용된다'는 결론을 얻을 수 있다. 화일 O_1에 대한 ACL은 보통 다음과 같이 접근제어 논리식의 집합 형태를 가질 수 있다.

$\text{ACL}(O_1) = \{A \text{ controls } s_{read},\ \ A \text{ controls } s_{write},\ \ B \text{ controls } s_{read}\}$

연결자 $\wedge$와 $|$이 임의의 조합으로 구성된 'speaks for' 논리식이 가정에 존재하면 주체의 접근 권한에 대한 증명을 구성하는 시간이 지수 시간이 될 수 있다. 따라서 접근제어 논리식의 증명이 다루기 쉬운 정도의 복잡도를 가지려면 논리식의 형태를 제한할 필요가 있다. 일반적으로 자주 등장하는 고수준 연산자는 'as', 'for' 등이 있으며 여기서는 'as'를 소개한다. 이것들은 주체 언어의 관용어구적인 사용 패턴을 반영하도록 고안되었기 때문에 저수준의 연산자들이 조합되는 방법이나 결합법칙과 명등식 등의 성질을 제한할 수 있다.

'as' 연산은 역할을 나타내는 방법으로 관리자 모드 R_{admin}로 작업하는 주체 A가 권한 s_1을 가지거나($A|R_{admin}$ controls s_1) 일반 사용자 모드 R_{user}로 작업하는 주체 A가 권한 s_2를 가질 수 있다($A|R_{user}$ controls s_2). 이것은 작업하는 주체가 본래의 권한보다 작은 권한을 가지고 작업하고자 할때 자주 이용된다. 또한 일반 사용자 모드인 주체 A를 대신하는 주체 B를 나타내기도 하고($B \Rightarrow A|R_{user}$) 주체 A가 가장 낮은 권한을 가지고 작업할 수도 있게 한다($A|R_{untrusted}$). 이를 위해서 $R, R' \in Roles$에 대하여 다음의 공리를 포함시킨다:
- $R|R = R$
- $R'|R = R|R'$
- $A \Rightarrow A$ as R

6 결론

컴퓨터 보안은 보안 정책과 보안 메카니즘을 다루며 소프트웨어 보안은 보안 메카니즘을 소프트웨어로 구현하거나 소프트웨어 시스템이 보안 정책을 준수하는 지를 확인하는 방법을 연구하는 분야이다. 접근제어시스템은 분산시스템과 같이 보다 일반화된 환경에서 매우 복잡해질 수 있다. 논리를 기반하는 방식이 최선이 아닐 수도 있지만 적어도 복잡한 접근제어시스템을 다루는 데에 유용하다. 본 논문에서는 양상 논리와 주체 언어로 이루어진 논리 기반 접근제어시스템을 설명하였다. 논리 기반 접근제어시스템은 올바르고 완전한 시스템을 구현하는 데에 유용하다. 또한 이 시스템이 다루기 쉬운 정도의 복잡도를 가지도록 하기 위해서는 접근제어 논리식의 형태를 제한할 수도 있다. 접근제어논리를 기반으로 하거나 유사한 방식을 이용하는 많은 이론과 시스템이 존재한다. 운영체제 보안[6], JVM 보안[7], 웹 접근제어시스템[9], PolicyMaker/KeyNote, SDSI, SPKI[8], D1LP/RT, SD3[11], Binder[10] 등이 있다.

참고문헌

[1] Harrison, M.A, Ruzzo, W.L., and Ullman, J.D., Protection in operating systems, Communications of the ACM, 19(8), 1976

[2] Blackburn, P., M. de Rijke, and Y.Venema, Modal logic, Cambridge University Press, 2001

[3] Li, N., Winsborough,W.H., and Mitchell, J., Design of a role-based trust-management framwork, Beyond proof-of-compliance: safety and availability analysis in trust management, Proceedings of 2003 IEEE Symposium on Security and Privacy, 2003

[4] Lampson, B., Abadi, M., Burrows M., and Wobber, E., Authentication in distributed systems: theory and practice, ACM Transaction on Computer Systems, 10(4), 1992

[5] Abadi, M., Burrows M., Lampson B., and Plotkin G., A calculus for access control in distributed systems, ACM Transaction on Programming Languages and Systems, 15(3), 1993

[6] Wobber E., Abadi, M, Burrows M., and Lampson, B., Authentication in the Taos operating system, ACM Transactions on Computer Systems, 12(1), 1994

[7] Wallach, Dan S., Appel, A.W., and Felton, E.W., SAFKASI: a security mechanism for language-based systems, ACM Transactions on Software Engineering and Methodology, 9(4), 2000

[8] Howell, J. and Kotz, D., A formal semantics for SPKI, Proceedings of 6th European Symposium on Research in Computer Security, LNCS 1895, Springer-verlag, 2000

[9] Bauer, L., Schneider, M.A., and Felten, E.W., A general and flexible access-control system for the Web, Proceedings of 11th USENIX Security Symposium, 2002

[10] DeTreville, J., Binder, a logic-based security language, Proceedings of 2002 IEEE Symposium on Security and Privacy, 2002

[11] Jim, T., SD3: a trust management system with certified evaluation, Proceedings of 2001 IEEE Symposium on Security and Privacy, 2001

신승철

– 1992-1996 인하대 전자계산학과 박사
– 1999-2000 캔사스주립대 연구원
– 1996-현재 동양대 컴퓨터학부 부교수

<관심분야> 프로그래밍 언어, 소프트웨어 보안, 프로그램 분석 및 검증, 수리 논리