# TTS기반에서 디지털 서명의 실행 인증을 통한 에이전트의 무결성 보장 기법

정회원 정 창 렬*, 윤 홍 상**

# Integrity Guarantee Scheme of Mobile Agents through Authentication of Digital Signature with TTS

Chang-ryul Jung*, Hong-sang Yoon** ***Regular Members***

요 약

여기는 본 논문은 이동 에이전트의 안전한 수행을 보장하기 위한 TTS기반의 디지털 서명 실행 인증 기법을 제안한다. 즉 기존의 연구의 문제점인 처리과정에서 발생하는 시스템의 처리속도와 네트워크의 트래픽을 개선한다. 또한 효율적이고 안전한 이동 에이전트의 실행과 무결성을 보장하기 위해 디지털 서명을 이용한다. 디지털 서명은 합성함수와 공개키 기반의 암호화 알고리듬 그리고 해시함수를 이용한 인증서 체인을 한다. 그리고 디지털 서명의 인증서 체인을 이용할 경우, 공격자에 의해서 체인을 끊고 새로운 인증서를 생성하여 삽입하는 공격으로부터 안전하게 보호한다. 또한 공격자에 의해 정직한 호스트를 악의적으로 이용될 수 있는 위협으로부터 보호한다. 그리고 컴퓨터 실험을 통해 인증서 처리에 대한 시스템의 처리속도와 실시간 처리를 분석한다. 이러한 분석을 통해서 시스템의 오버헤드와 네트워크의 트래픽에 대한 효율성을 증명한다.

**Key Words** : Agent, Digital Signature, Authentication, Integrity

ABSTRACT

This paper propose the technique for the execution authentication of digital signature with TTS(traceable trust server) to guarantee the safe execution of mobile agents. That is to say, it is focused on improving the processing speed of systems and the traffic of network, which are problems in the existing studies. The digital signature is used to guarantee the efficient and safe execution and the integrity of mobile agents. The certificate of it is chained with synthesis function, cryptographic algorithm based on public key, and hash function. And white hosts can be protected against the threat of being used maliciously. Then, we prove the efficiency of system overhead and the traffic of network by the analysis. In case the certificate chain of a digital signature is used, the safe execution of mobile agents can be protected against attackers that wish to insert a newly created certificate after cutting off the chain after striking space key 2 times.

## Ⅰ. 서 론

이동 에이전트는 실행 코드에 의해 실행되며, 이질적인 통신망에서 프로그램의 상황에 따라 동적으로 대처하는 유연성을 지니고 있다. 뿐만 아니라 에이전트로 생성이 되어 에이전트의 코드와 상태 그리고 데이터로 이루어져 임무가 완료되면 소멸된다. 호스트는 수행 될 이동 에이전트가 호스트에 방문

* 순천대학교 컴퓨터과학과 DB&EC 연구실 (chari7@suchon.ac.kr), ** 광주대학교 컴퓨터정보통신공학부 (hsyoon@gwangju.ac.kr)
논문번호 : KICS2005-08-336, 접수일자 : 2005년 8월 16일, 최종논문접수일자 : 2006년 5월 18일

하면 에이전트 확인을 위해 인증서를 검증하여 상호 신뢰성을 확보한다. 그 후 에이전트가 실행할 수 있도록 에이전트 실행 환경을 제공한다. 이러한 과정은 이동 에이전트의 이동 계획에 의해 임무가 완료되면 소멸한다. 에이전트의 구성은 코드와 에이전트 실행 결과가 있는 데이터로 되어 있다. 데이터는 항상 가변적이기 때문에 에이전트의 실행 결과의 안전성을 보장받기 위해서는 보안이 필요하다.

또한 다양한 네트워크에서 실행됨으로 통신상의 보안 문제와 에이전트가 호스트와 상호 작용 과정에서 악의적인 호스트에 의해 공격 받을 수 있기 때문에 안전한 실행을 보장받아야 한다. 이처럼 에이전트의 안전한 수행과 무결성 보장을 위해 디지털 서명과 에이전트의 실행 추적 등이 논의되고 있다. 디지털 서명은 사이버 공간에서 데이터의 위·변조에 대한 증명과 인증, 무결성, 부인 봉쇄를 할 수 있다. 때문에 에이전트의 무결성을 보장하기 위해 쌍 유리 함수[1], 임계치 서명[2], 컨테이너에 서명 삽입[4], 해시 함수와 연결고리[9], 암호화 함수[3] 등이 제안되었다.

그러나 에이전트의 무결성 및 에이전트 코드 수정과 저장된 데이터의 안전성의 문제로 인하여 쌍 유리 함수와 임계치 서명 방법은 서명의 비분성에 대한 키의 길이와 서명에 대한 검증 문제가 있다. 컨테이너에 서명 삽입과 해시함수와 서명 연결고리 방법은 서명과 체크 섬에 의존하기 때문에 검증이 쉽지 않아 위조의 가능성이 있다. 암호화 함수는 암호화 함수를 부호화하는 과정에서 모순된 제약이 발생하여 서명과 키 지배력 높게 하고, 키 크기의 증가로 시스템 오버헤드와 시간 제약의 요구조건을 충족해야 하는 문제가 발생한다. 디지털 서명뿐만 아니라 에이전트의 보안 요구사항까지 만족시키기 위해서는 실행 추적 인증 메커니즘을 통해서 에이전트의 안전성과 무결성을 보장하거나[5, 7], 특정 플랫폼을 두어서 무결성 체크하는 방법[6]이 있다. 또 임계값에 의한 서명 방법[8]이 있다. 이 방법들은 견고성과 무결성 강화로 인해서 수행 경로가 많아질수록 로그 사이즈와 로그 개수도 증가한다. 그 때문에 계산량의 증가로 인한 트래픽 증가로 효율성이 떨어진다. 그 밖에 검출기를 통해서만 검증이 이루어지는 한계도 있다. 에이전트는 안전하고 견고한 실행 보장뿐만 아니라 시스템 처리 속도와 네트워크의 트래픽, 시스템에 대한 오버헤드를 최소화해야 한다. 그리고 에이전트의 무결성도 보장되어야 한다. 또한 정당한 호스트가 악의적인 위협에 의해서 악의적으로 남용되지 않아야 한다. 이를 위해 본 논문에서는 TTS기반의 디지털 서명의 실행 인증을 위해 실행 추적을 통해서 이동 에이전트의 보장과 안전한 실행이 이루어질 수 있는 기법을 제안한다.

본 논문에서 2장은 안전한 디지털 서명 생성과 인증서를 TTS를 기반으로 생성하고, 3장은 디지털 서명의 실행 인증 처리의 프로토콜을 제시한다. 4장은 제안한 기법에 대한 기존 연구와 비교 분석 및 평가를 하고, 5장에서 결론을 제시한다.

## Ⅱ. 안전한 디지털 서명 생성과 TTS

이동 에이전트는 자율적으로 수행이 가능한 능동적인 객체로 호스트에 의해서 에이전트의 코드 수정과 데이터의 간섭을 받을 수 있기 때문에 무결성 보장을 위해 디지털 서명을 이용한다.

그러므로 디지털 서명의 인증서는 간결하면서 안전하고 견고해야 한다. 그렇기 위해서는 표준적인 보안 요구를 입증해야 한다. 이를 위해 공개키 기반의 정책 정보를 포함한 임계값을 매개 변수($Pk$)로 사용하여 디지털 서명의 인증서를 생성한다. 생성된 인증서는 이동되는 멀티 홉 간에 체인 관계를 형성하여 TTS(traceable trust server)의 디지털 서명 모듈에 의해 에이전트 서버에서 암호화 함수에 의해서 생성한다. 이는 다양한 서명 체제의 응용이 가능한 공개키 기반에서 $n$개의 호스트를 이동(migration)하면서 이루어지기 때문에 악의적인 위협 요소들로부터 무결성이 이루어진다.

디지털 서명을 TTS에 의해 생성된 디지털 서명의 인증서는 디렉터리에 저장하고 $AS_i$(에이전트 서버)에 의해서 인증서 체인 관계를 형성한다. 인증서는 디지털 서명의 견고성을 위해 임계값에 의해서 RSA를 기반으로 한 합성 함수 서명을 한다. $n$개 요소들의 집단이 어떤 매개변수 $Pk_i(1 \le i \le n)$에 대하여 어떤 $i$번째 요소의 메시지에 대한 유효한 디지털 서명이 이루어지도록 하지만 $i-1$에 대한 어떠한 집합도 일괄적으로 디지털 서명이 이루어질 수 없다는 것에 근거한다. $AS_i$는 매개변수 $Pk_i$를 체인 관계를 형성하여 트랜잭션이 발생할 때 TTS에 의해 처리되며, 이 과정에서 위·변조로부터 에이전트를 보호하여 기밀성과 무결성을 보장한다.

에이전트의 인증서는 그림 1과 같이 수행 처리 과정을 거쳐 디지털 서명의 인증서를 생성한다. 그림 2는 이러한 디지털 서명의 인증서 생성 및 처리 알고리듬을 나타내고 있다. 수행 절차는 매개 변수 $Pk_i$

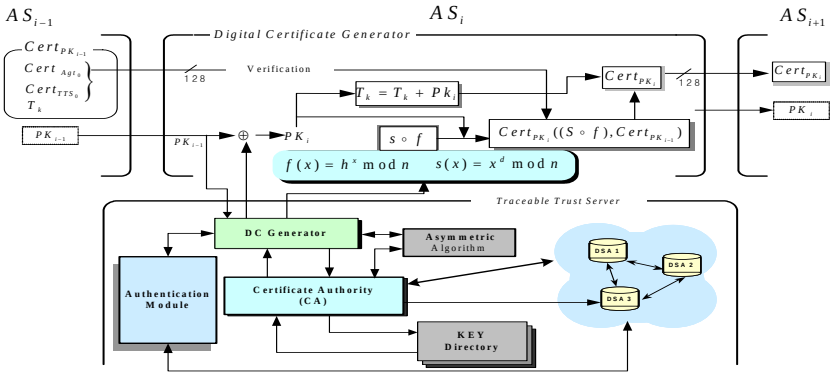


그림 1. 디지털 서명 생성과 인증서 처리

에 의한 인증서 체인 관계를 형성한다. 현재 에이전트 서버는 이전 호스트 플랫폼으로부터 인증서 $Cert_{Pk_{i-1}}$을 수신한다. 수신되면 TTS의 CA에 의해서 인증서 검사한 후 인증 모듈에 의해 인증을 한다. 이때 매개 변수 $Pk_{i-1}$을 TTS의 CA의 디지털 서명 생성 모듈에 의해 새로운 $PK_i$가 생성된다. 이때 $PK_i$는 $AS_i$에 의해서 다음 호스트로 전달되면서 $PK_i$는 $AS_{i+1}$의 $PK_{i+1}$와 체인관계 형성을 한다.

$AS_i$는 디지털 서명 함수의 쌍으로 이루어진 합성함수 $S \circ f$가 처리되어 디지털 서명된다. 이렇게 생성된 디지털 서명의 인증서는 $Cert_{Pk_i}$이다. 이 인증서는 $AS_{i+1}$와 체인 관계에 의한 인증서의 상호 인증이 이루어진다. 이 과정에서 $Cert_{Pk_i}$가 CA의 공개키를 갖지 못한다면, 인증서 체인을 통해서 유도될 수 있도록 공개키 기반의 혼합형 인증 경로에 의한 상호 인증이 이루어지도록 한다. 이렇게 생성된 $Pk_i$는 $T_k$에 저장되며, $T_k$는 $T_k = k_i + k_{i+1} + \cdots + k_n$이다.

$i(1 \le i \le n)$는 현재 실행중인 $AS_i$를 의미 한다. 또한 $T_k$는 인증서 $Cert_{Pk_i}$에 포함된다.

### 2.1 디지털 서명과 인증서 발행

견고함과 무결성 보장을 위해 에이전트를 생성한 소유자에 의해 TTS의 디지털 서명 모듈 생성자와 XOR하여 $Pk_i$를 생성한 후 체인 관계를 이룬다.

이러한 서명의 시퀀스는 $AS_{i-1}$에서 전달된 $Cert_{k_{i-1}}(T_k, Cert_{Agt_0}, Cert_{TTS_0}, Cert_{k_{i-2}}, Cert_{TTS_{i-2}})$이다. $T_k$는 서명에 이용되는 매개 변수 $Pk_i$의 저장된 합을 의미하고, $Cert_{Agt_0}$는 에이전트 소유자에 의해 디지털 서명이 이루어진 에이전트 서명이다.

$Cert_{TTS_0}$는 에이전트 소유자의 AS와 연결된 TTS에 의한 최초의 디지털 서명이고, 디지털 서명에 사용되는 변수는 $d, h, r, T_k, PK_i$이다. 전달 받은 $Pk_i$는 업데

$n$ : combined numeric
d : secret key
$h$ : hash function
$r$ : constraints of the customer
$T_k$ : threshold parameter's sum
$PK_i$ : $i$ threshold parameter value

Receive : $Cert_{Pk_{i-1}}$ , $PK_{i-1}$
$Cert_{k_{i-1}}(T_k, Cert_{Agt_0}, Cert_{TTS_0}, Cert_{k_{i-2}}, Cert_{TTS_{i-2}})$

Parameter Value Update:
$PK_i = PK_{i-1} \oplus TTS_i$
$T_k = T_k + PK_i$

Parameter Value Chain Relation :
transfer : $PK_i \rightarrow AS_{i+1}$

Digital Signature :
$h = hash(c, r)$ : one-way hash function
$t = h^d \bmod n$ : RSA signature of $h$
$s(x) = x^d \bmod n$: RSA signature function
$f(x) = h^x \bmod n$, $f_{signed}(x) = t^x \bmod n$
: digital signature function pair
$(s \circ f)(x) = s(f(x)) = s(h^x) = (h^x)^d$
$= (h^d)^x = t^x = f_{signed}(x)$

Certificate Creation
$Cert_{PK_i} = Cert_{Pk_i}((s \circ f), Cert_{PK_{i-1}})$

Digital Signature Certificate Chain Relation :
transfer
: $Cert_{PK_i}(AS_i)$ : $AS_i \rightarrow AS_{i+1}$ : $Cert_{PK_i}(AS_i)$

그림 2. 디지털 서명 알고리듬

이트를 위해 TTS의 디지털 서명 모듈에 의해서 이루어진다. $PK_i = PK_{i-1} \oplus TTS_i$. 이때 생성된 매개 변수 $PK_i$는 $T_k = T_k + PK_i$를 하여 누적시키고 매개 변수의 값 $PK_i$를 체인 관계를 형성하기 위해서 $AS_{i+1}$로 전송된다. 합성 함수와 해시 함수 $h$에 의해 서명된다. 즉, $h$의 RSA 서명 $t$, RSA서명 함수 $S(x)$, 그리고 디지털 서명 함수의 쌍인 $f(x)$와 $f_{signed}(x)$을 이용한다. 즉 $(s \circ f)(x) = s(f(x)) = s(h^x) = (h^x)^d = (h^d)^x$이다. $Cert_{PK_i}$는 $Cert_{PK_i} = Cert_{PK_i}((s \circ f), Cert_{PK_{i-1}})$가된다. 이로써 함수 쌍이 생성되면 디지털 서명은 $PK_i$에 의해 다시 서명이 이루어진다. 그러므로 디지털 서명 함수의 쌍을 공격하여 서명이 해독 되어도 임계값 $PK_i$을 알아야 완전한 해독이 된다. 그러므로 $PK_i$는 에이전트 소유자에 의해 생성됨으로 해독을 위해서는 많은 호스트 플랫폼들과의 공모가 이루어지지 않고서는 해독이 어렵게 되어 디지털 서명은 안전하다.

## Ⅲ. 디지털 서명의 실행 인증 처리 프로토콜

### 3.1 디지털 서명의 실행 인증

호스트의 TTS는 에이전트가 수신되어 실행됨과 동시에 추적과 에이전트의 현재 상태를 검증하고 인증된다. 때문에 TTS의 인증 모듈은 에이전트 인증서 발행과 에이전트의 결과 데이터를 암호화할 수 있도록 암호화 알고리듬 지원과 에이전트가 실행되는 동안 DoS (denial of service)공격으로 부터 보호한다. 또한 에이전트의 재실행(replay) 공격으로 부터 보호하기 위해 암호화 키는 단 방향 해시 함수를 이용하며, 에이전트 실행 추적 인증은 그림 3과 같이 처리된다. 즉, TTS는 에이전트가 호스트에서 안전하게 수행될 수 있도록 호스트 간 이동 중에 인증하여 에이전트의 무결성을 보장한다. 또한 TTS는 인증서를 발행하는 인증기관과 이동 에이전트가 호스트에 수신되어 실행되는 동안 에이전트의 실행 추적도 병행한다.

이 과정은 호스트와 이동 에이전트 간 신뢰와 안전한 실행을 보장한다. 그림 3의 호스트 간 상호 인증은 이동하는 에이전트를 인증하고 신뢰하는 과정이며, 실행 인증은 이동 에이전트에 대한 에이전트 서버와 TTS 간의 정당한 처리가 이루어질 수 있도록 하는 절차이다. AS(에이전트 서버)와 TTS간 실행 추적 과정은 에이전트를 검증하고, 에이전트 안전한 실행을 보장하고, 악의적인 요소들로부터 에이전트를 보호할 뿐만 아니라 에이전트에 대한 신뢰

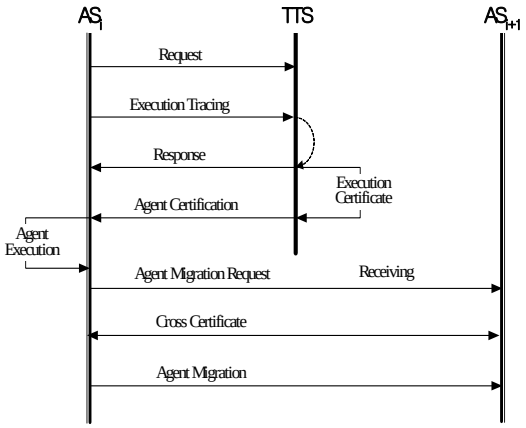


그림 3. AS와 TTS간 실행 추적 과정

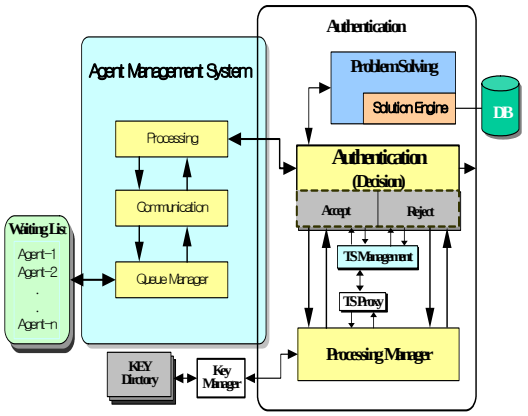


그림 4. TTS의 인증 처리 과정

성과 에이전트 데이터에 대한 무결성을 보장하기 위한 인증 절차이다. TTS의 인증 처리는 통신 채널을 통해서 TTS에 수신된 에이전트를 순차적으로 처리 모듈에 의해 이루어진다. TTS의 인증 처리는 통신 채널을 통해서 TTS에 수신된 에이전트를 순차적으로 처리 모듈에 의해 이루어진다.

### 3.2 디지털 서명 처리 프로토콜

디지털 서명을 통해서 에이전트의 무결성 보장받는다. 이때 생성된 디지털 서명의 인증서는 체인관계를 이루고 있다.

그러나 공격자가 인증서 체인을 끊고 새로운 인증서 체인을 생성하는 공격을 할 때 수신된 호스트에서는 이러한 공격 사실을 알지 못 하기 때문에 정상적인 호스트가 악의적인 요소들로 이용하는 오라클 공격에 의해 악의적으로 남용될 수 있다. 때문에 에이전트를 처리하기 위한 인증 모듈의 TS(trac-

ing service) 프락시와 TS관리 모듈에 의해서 검색과 갱신되어 처리된다. TS의 처리와 추적되는 과정의 프로토콜은 다음과 같이 이루어진다.

$$i_{n-1} \rightarrow i_n : C_{n-1},$$
$$i_n \rightarrow TS : i_n, C_n, C_{n-1},$$
$$TS \rightarrow i_n : M \in \{ok, error\}, \quad i_n \rightarrow i_{n-1} : C_n$$

이로써 $i_n$이 $i_{n-1}$으로부터 에이전트가 수신되면 TS에 의해서 추적하여 $i_{n-1}$의 에이전트 서버로부터 수신되었다는 것을 식별을 한다.

식별되면 수신된 체인 관계 $C_{n-1}$을 인증한다. 여기서 $C$ 는 그림 1에서와 같이 체인 관계로 이루어진 암호화된 인증서이다. 인증이 이루어지면, 이동 계획에 의해 보내기 위해서 인증 모듈을 통해서 에이전트 서버 $AS_{i_n}$에게 권한을 위임 한다. 이로써 권한이 없는 악의적인 호스트는 에이전트에 접근할 수 없게 됨으로써 가로채기 등의 공격을 할 수 없다. 그러나 악의적인 호스트와 같은 공격자는 현재 호스트에서 DoS공격을 시도할 수 있음으로 공개키 기반에서 서명 함수를 이용한 암호화와 실행 추적을 한다. 때문에, 이처럼 비정상적인 처리는 실시간으로 검출된다. 그리고 정상적인 호스트가 악의적으로 이용될 경우, 호스트 플랫폼에서 TTS는 에이전트가 이동되어 온 에이전트를 항상 체크한다. 그러나 만약 $Cert_{PK_{i-1}}$가 공격자에 의해 해독 되었다고 가정한다면, $Cert_{PK_{i-1}}$는 정당한 호스트로 위장을 하기 위해서 다음과 같은 수행 과정을 하게 될 것이다.

먼저 TTS와 공모를 시도한다. TTS공모에 응하면 공격자는 송신자에 의해서 $S \circ f$의 함수의 공개키 암호 기법을 암호화되었기 때문에 $AS_{i-1}$의 비밀 키 $d$을 추출해야만 새로운 인증서를 만들 수 있다. 그러나 공격자가 $f^{-1}_{signed}(x)$를 알기 위해서는 송신자의 비밀 키를 해독하기 위해 $p$와 $q$를 알려고 할 것이다. 그러나 이 또한 실현 불가능하다. 이는 송신자만이 $p$와 $q$알고 있기 때문에 소수 $p$와 $q$를 곱하여 비밀 키를 생성할 수 있기 때문이다. 그리고 인증서는$(s \circ f)(x) = s(f(x)) = s(h^x) = (h^x)^d = (h^d)^x = t^x = f_{signed}(x)$로 암호화되고, 단 방향 해시 함수에 의한 인증서 체인이 이루어졌기 때문에 이 또한 해독이 불가능함으로 에이전트 의 안전함과 무결성이 보장된다.

## Ⅳ. 비교 분석 및 평가

디지털 서명이 안전성과 신뢰성을 보장을 위해서는 기본적인 보안 요구사항이 만족되어야 한다. 디지털 서명은 사이버 공간에서 송·수신자간의 신원과 송신된 메시지의 위·변조 방지 및 수신 사실의 부인 봉쇄를 하기 위해 디지털 서명의 인증서를 사용한다. 본 논문에서 디지털 서명의 실행 인증을 통한 무결성을 보장하기 위해 제안한 기법을 기존 연구의 방법들과 비교한다. 또한 디지털 서명의 인증서 처리에 대한 성능 평가도 병행한다.

서명의 크기의 경우 임계서명은 $AS$(에이전트 서버)의 인증서, $k$개의 인증서, 그리고 $k$개 서명이 공유되어 있어 서명의 크기가 $2k+1$이 된다. 그러나 이 경우 서명 공유가 이루어 져 있기 때문에 에이전트가 처리해야 하는 계산량이 증가하게 된다. 서명을 검증할 때도 서명의 인증을 수신하면 $k+1$(에이전트 소유자의 인증서+$AS$의 인증서들)개와 $k$개의 서명들을 각각 검증해야 한다. 무결성의 강도는 RSA기반의 임계 서명을 하기 때문에 키의 사용이 엄격하여 무결성 보장이 이루어진다. 그러나 컨테이너 삽입 서명은 컨테이너를 보호하기 위해 체크 합에 의존하기 때문에 이동될 때 다른 한 개의 에이전트 서버와 공모를 한다면 체크 합을 성공적으로 해독하고, 서명을 추출하여 컨테이너에 다시 넣을 수 있는 문제가 있다.

이러한 인증서를 추출하기 위해 다른 에이전트 서버와 공모를 하더라도 제안한 기법에서 추출이되지 않는다.다만 추출하기 위해서는 전체의 에이전트 서버와 공모를 하지 않는 한 추출은 불가능하다. 무결성의 검사의 경우 임계 서명은 발생된 인증서 전체를 검증해야 한다.  이는 컨테이너 삽입 서명도 마찬가지이다. 그러나 제안한 방법에서 현재의 $AS$에서 서명을 추출하여 TTS에서 검증하고 인증한다. 제안한 기법은 인증서를 상호 인증과정에서 인증서에 대한 실행 추적이 가능함으로 효과적이다. 마지막으로 이동 속도에서도 에이전트 서버에서 서명을 검증하고, 인증서를 발행하는 다른 방법과 달리 제안한 방법은 인증서에 안전성은 TTS에 의해서 서를 발행하는 다른 방법들과 달리 제안하는 방법은 TTS에서 검증되고 실행만 에이전트 서버에서 이루어지기 때문에 처리 속도가 기존 방법에 비해 현저히 빠르다. 그러나 제안한 방법의 시스템은 다른 보안 기법들과 통합 구성되어야 한다는 한계가 있다. 비교 평가 결과는 다음 표 1과 같다.

디지털 서명의 인증서 처리에 대한 성능의 분석은 윈도우즈 2000서버, CPU P-Ⅲ1600MHz, RAM512MB, JDK1.3에서 시뮬레이션 하였다. 인증

표 1. 기존 연구들의 서명 방법과 비교

| | 임계 서명[3] | 컨테이너 삽입서명[5] | 제안한 서명기법 |
|---|---|---|---|
| 서명의크기 | 2k+1 | k+1 | k+1 |
| 서명검증 | 2k+1 개 | k 개 | k 개 |
| 무결성강도 | 보통 | 낮다 | 높다 |
| 무결성검사 | 전체 | 전체 | 부분적 |
| 추적성 | 없다 | 없다 | 있다 |
| 서명기반 | 공개키 | 공개키 | 공개키 |
| 이동속도 | 보통 | 느림 | 빠름 |

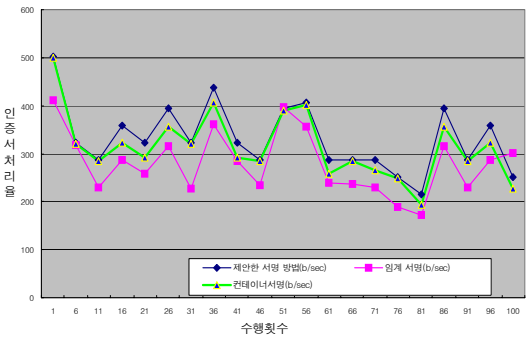


그림 5. 인증서 처리율에 대한 비교 분석

서 사이즈와 최대 처리 요구율과 매우 넓은 영역의 네트워크 최대 네트워크의 로드율, 최대 처리 로드율, 최대 지연율 그리고 인증서 확인율을 계산할 수 있도록 인증서 사이즈와 인증서 확인율 등을 증가시키면서 인증서의 정당성을 분석하였다. 인증서의 사이즈는 100, 처리 가능율과 기간을 100건/일, 1분으로 하며, 인증서 취소율은 연 인증서의 10%라고 가정한다. 또한 시뮬레이션을 위한 최대 네트워크 로드의 대여폭은 10Mbit이고, 최대 처리 가용량은 $100ns$이다. 인증서의 사이즈가 100B이고 일일 처리되는 평균은 100이고, 취소율 10%일 때 타당한 처리 기간을 60분으로 하여 100회 실시한 결과는 그림 5와 같다.

인증서 처리 요구에 대한 처리(*nanosecond*)는 요구에 따른 바로 처리가 이루어져 최대 $1.2ms$의 지연되었으며, 인증서 처리가 이루어지는 처리율의 차이는 제안하는 방법이 기존 방법에 비해 효과적임을 알 수 있다.

## V. 결 론

에이전트의 안전한 수행을 위해 TTS 기반의 디지털 서명의 실행 인증 통한 악의적인 요소들로부터 에이전트의 안전한 수행을 지원할 수 있는 기법을 제안하였다. 제안된 기법은 보안 정책과 서명에 대한 유효 기간을 포함하는 임계치의 값인 매개 변수를 사용한 디지털 서명이다. 서명된 인증서는 호스트 간 신뢰 확인과 안전한 에이전트 수행을 보장한다. 뿐만 아니라 에이전트의 무결성 또한 보장한다. 제안된 기법의 성능을 분석하기 위해 디지털 서명의 인증서 처리에 대한 분석을 하였다. 에이전트와 함께 이동되어 온 인증서를 TTS에 의해서 실시간으로 검색하여 인증서의 상태를 검증하는 최대 요구에 대한 최대 처리의 속도가 빨라 실시간으로 안정적인 처리가 이루어졌다.

## 참 고 문 헌


〔1〕 T. Sander, and C. Tschudin, "Protecting Mobile Agents Against Malicious Hosts," in G. Vigna(Ed.), Mobile Agents and Security, LNCS1419, pp.44-60, Springer-verlag, 1998.

〔2〕 V. Shoup, "Practical threshold signa- tures," In B. Preneel(Ed.), Advances in Cryptology - EUROCRYPT 2000, LNCS 1807, pp.207-220. Springer -Verlag, 2000.

〔3〕 P. Kotzanikolaou, M. Burmester, and V. Chrissikpoulos, "Secure Transactions with Mobile Agents in Hostile Environments," in E. Dawson et al. (Ed.s), Information Security and Privacy, in Proceedings of the 5th Australasian Conference ACISP 2000, LNCS 1841, pp.289-297, Springer-verlag, 2000.

〔4〕 N. M. Karnik, A. R. Tripathi, "Security in the Ajanta Mobile Agent System," Technical Report TR-5-99, University of Minnesota, Minneapolis, MN 55455, U.S.A., May, 1999.

〔5〕 C. R. Jung, "A Framework of Agent Protection Protocol for Secure Execution of Mobile Agent," in Journal of KIMICS, Vol. 8, No. 2, pp. 371-378, 2004.

〔6〕 H. K. Tan and L. Moreau, "Certificates for Mobile Code Security," in Proceedings of the 17th ACM Symposium on Applied Computing, March 2002.

〔7〕 H. K. Tan, and L. Moreau, “Extending Execution Tracing for Mobile Code Security,” K. Fischer, D. Hutter(Eds), in Proceeding of the 2nd International Workshop on Security in Mobile Multi-Agent Systems(SEMAS’ 2002), DFKI Research Report(RR-02-03), pp.51-59, 2002.

〔8〕 N. Borselius, C. J. Mitchell, and A. Wilson, “On the Value of Threshold Signature,” in Proceeding ACM SIGOPS Operating Systems Review, Vol. 34, No. 4, pp.30-35, 2002.

〔9〕 박종열, 이동익, 이형효, 박중길, “이동 에이전트의 데이터 보호를 위한 일회용 에이전트 키 생성 시스템”, 한국정보과학회논문지:정보통신 제28권, 제3호, pp.309-320, 2001.

**정 창 렬** (Chang-ryul Jung) 정회원

1995년 2월 광주대학교 컴퓨터공학과 졸업

1999년 8월 순천대학교 컴퓨터교육과 석사

2005년 2월 순천대학교 컴퓨터과학과 박사

2003년 6월 Alberta State University, Canada, Visiting Researcher.

<관심분야> 정보보안, 전자상거래, 이동에이전트, 데이터베이스

**윤 홍 상** (Hong-sang Yoon) 정회원

1987년 2월 Western Illinois Univ. 수학과 졸업

1990년 2월 Western Illinois Univ. 컴퓨터과학과 석사

2000년 2월 순천대학교 컴퓨터과학과 박사

2000년 8월~현재 광주대학교 컴퓨터정보통신공학부 교수

<관심분야> 네트워크보안, 이동에이전트, 데이터통신, 의료영상보안