VOD에 적용 가능한 인증 방식 실현

염 흥 열*, 김 상 필*, 이 종 형**

순천향대학교 전자공학과*, 한국전자통신연구소**

Implementation of Authentication Method for VOD

Heung-Youl,Youm*, Sang-Pil,Kim*, Jong-Hyeong,Lee**

Dept. of Electronic Eng., Soonchunhyang Univ.*,
Electronics and Telecommunications Research Institute**

-요약-

VOD(Video on Demand) 에서의 인증 시스템은 기본적으로 공개키 증명서 방식과 공개키 암호 메카니듬을 이용한 인증 프로토콜에 바탕을 두고 실현될 수 있다. 본 논문에서는 VOD 에서의 인증 시스템을 실현하기 위한 인증 프로토콜 스택, 인증 프로토콜, 그리고 적용 가능한 인증 메카니듬을 분석한다. 그리고 VOD 인증 시스템을 실현하기 위해 요구되는 서버 및 사용자 측면에서의 인증 관련 변수 및 타이머를 정의하고, 각 개체에서의 인증 절차를 규정한다. 또한 국내 VOD 시스템에 적용될 수 있는 인증 방식의 기본 원칙을 제시한다. 그리고 인증을 위한 공개키 증명서의 형태를 정의하고, RSA 서명 메카니듬을 이용하여 이를 구현하며, 인증 프로토콜을 768 비트 RSA 서명 메카니듬 및 MD5 해쉬 함수를 이용하여 C 언어로 실현한 시뮬레이션한 결과를 제시한다. 본 연구의 논문의 결과는 VOD CA(Certificate Authority) 구축시 및 VOD 인증 시스템 실현시 유용하게 적용될 수 있다.

제1장 서론

광대역 ISDN 이 구축되면서, VOD 서비스에 대한 요구는 급증할 추세이다. 따라서 음성, 영상, 그리고 비디오 등의 멀티미디어 서비스가 광대역 ISDN 을 통해 활발히 유통될 것이다. 현재 DAVIC(Digital Audio-Visual Council) 에서는 양방향 비디오 서비스에 대한 국제 표준화에 대한 연구를 수행하고 있다. VOD 에서의 핵심 기능중의 하나는 비인가자에 의한 비디오 신호의 액세스를 제한하는 액세스 제어 기술이다[1]. 액세스 제어 기술은 인증 기술로 실현될 수 있다. 액세스 제어는 가입자측 장비인 STU(Set Top Unit), 각 가입자의 액세스 제어 기능을 수행하는 스마트 카드 형태, 그리고 서비스 제공자의 자격 관리 및 제어 기능을 담당하는 보안 모듈 등을 이용하여 실현될 것이다. VOD 서비스 제공은 사용자 인증이 완료된 후 수행되어야 한다. DAVIC 에서 스마트 카드 인터페이스 표준은 ISO 7816 권고안을 따른 CA1 인터페이스와 별도의 CA0 인터페이스를 권고하고 있다. 인증은 다른 개체에 한 개체가 신분을 확인 받는 과정이다. 인증 방식은 크게 패스워드와 사용자 ID 에 기반을 둔 기본 인증 방식, 사용자 ID, 계수기, 그리고 인증 부호를 이용한 일회용 인증 방식, 그리고 challenge and response 에 기반을 둔 대화용 인증 방식으로 구분된다. 대화용 인증 방식은 대칭형 암호 메카니듬과 공개키 암호 메카니듬을 이용하는 방식이 있다. 또한 대화용 인증 방식은 공개키 증명서에 바탕을 둔 인증 방식과 영지식 증명 방식으로 분류될 수 있다. 인증 프로토콜을 수행함으로서 부차적으로 추후 암호 통신의 기밀성 및 무결성 서비스를 위한 세션키들을 교환할 수 있다.

본 논문의 2장에서는 DAVIC 에서 권고중인 인증 프로토콜 스택, 인증 프로토콜, 그리고 적용 가능한 인증 메카니듬을 분석한다. 그리고 VOD 인증 시스템을 실현하기 위해 요구되는 서버 및 사용자 측에서의 인증 관련 변수 및

타이머를 정의하고, 공개키 증명서의 형태를 제시한다. 또한 국내 VOD 인증 방식 원칙을 제상한다. 그리고 3장에서는 768 비트 크기의 RSA 서명 알고리듬을 이용하여 공개키 증명서를 구현하며, 구체적인 인증 프로토콜을 RSA 서명 메카니듬 및 MD5 해쉬 함수를 이용하여 C 언어로 실현한다. 본 논문의 결과는 VOD CA 구축시 및 VOD 인증 시스템 실현시 유용하게 활용될 수 있다.

제2장 공개키 증명서 및 인증 프로토콜

2.1 인증 관련 정보 흐름 및 프로토콜 스택

DAVIC 에서의 S1 정보 흐름은 내용(Content) 정보 흐름으로서 서버로 부터 사용자로의 정보 흐름이다. S1 정보 흐름은 인증이 완료된 후, 사용자로 전송되어야 한다. S1 정보 흐름은 ATM 기본 전송 시스템에 바탕을 두는 경우, 그림 2.1과 같은 같은 프로토콜 스택을 갖는다.

<table>
<tr><td>Other data</td><td>MPEG2 비디오
기본 스트림</td><td>MPEG2 오디오
기본 스트림</td><td></td></tr>
<tr><td>MPEG2 private section</td><td colspan="2">MPEG-2 Packetized Elementary Stream</td><td>MPEG2 Program
Specific Information</td></tr>
<tr><td colspan="4">MPEG2 Transport Stream</td></tr>
<tr><td colspan="4">AAL-5</td></tr>
<tr><td colspan="4">ATM</td></tr>
<tr><td colspan="4">Lower Laver</td></tr>
</table>

그림 2.1 S1 정보흐름을 위한 프로토콜 스택

내용 정보에는 MPEG 비디오 및 오디오 정보를 포함한다. 기타 데이타는 파일, DSM-CC DownloadDataBlock, DVB 서비스 정보(Service Information), 영상, 그리고 실시간 그래픽 정보를 포함한다. MPEG2 PSI(Program Specific Information) 에는 PAT(Program Association Table), PMT(Program Map Table), CAT(Conditional Access Table), 그리고 NIT(Network Information Table) 등을 포함한다.

표 2.1 PSI(Program Specific Information)

MPEG2 PSI	스트림	PID 번호	내용
PAT	MPEG	0x00	프로그램 번호와 Program Map Table 의 PID 를 전달함
PMT	MPEG	PAT_PID 로 할당	하나 이상의 프로그램 요소에 대한 PID 값 규정
CAT	MPEG	0x01	하나 이상의 EMM 에 특정의 PID 값 할당
NIT	private	network PID 로 할당	FDM 주파수와 수신기의 개수 등의 물리적 망 변수

PSI(Program Specific Information) 는 디코더에서 프로그램의 역다중을 가능케 하는 정보이다. 프로그램은 하나 이상의 기본 스트림들로 구성되며, 각각의 기본 스트림은 고유의 PID 를 갖는다. 프로그램과 기본 스트림은 한정 액세스되어야 한다. PSI 정보는 스크램블링되지 않는 형태로 전달된다. PAT(Program Association Table) 는 프로그램 번호와 프로그램 정보를 전달하는 Program Map Table 의 PID 값을 전달하며, PAT 의 PID 값은 "0x00" 이다. 프로그램 번호 "0x0000" 은 network PID 용으로 설정되었다. PMT(Program Map Table) 은 프로그램과 이를 구성하는 기본 스트림간의 관계를 나타내며, 주로 기본 스트림들의 PID 을 전달한다. 이는 Program Map Table Section 으로 전달된다. CAT는 하나 이상의 CA 시스템과 CA 시스템의 EMM 채널을 전달하는 트랜스포트 패킷의 CA_PID 를 전달한다.

2.2 인증되어야 할 개체

VOD 에서 인증이 요구되는 개체(Entity) 들은 스마트 카드, 디스크램블러 등의 보안 장치, NIU(Network Interface Unit), STU(Set Top Unit), 서비스 제공자(Service Provider), 내용 제공자(Content Provider), 전달 시스템(Delivery System) 에서의 서비스 관련 제어 개체 등이다. 이들 개체는 완전히 구별 가능해야 하고, 계층적 구조로 구성되며, 사용자와 친숙한 ID 를 가져야 한다. 망 계층에서의 인증은 STU 와 SRM 간 인증을 수행하는 망차원 인증과 서버와 STU 내의 스마트 카드간에 인증되는 종점간 인증 방식이 있다.

2.3 STU 인터페이스

STU 는 디스크램블러(Descrambler), MPEG 복호기, 역다중화기, 그리고 보안 관리 기능(Security Management Functions) 으로 구성되어 있으며, NIU 에서는 망 관련 보안 기능을 수행한다. DAVIC 에서는 보안 장치는 암호 장치의 설치 및 수출에 엄격한 제한이 있고, 보안 위협 요소는 너무 다양하여 다양한 공격 방법으로 위협받으며, 보안 장치가 안전하다고 확신하기가 매우 어렵고, 실제로 완전히 안전한 보안 장치를 실현하는 것은 불가능하다는 이유로 세계 표준 단일 보안장치로 표준화되지 않을 전망이다. DAVIC STU 는 그림 2.2와 같이 수신 신호를 검출하는 수신기 및 복조기, 스마트 카드로의 CA 메세지을 전달하기 위한 필터부, 암호화되어 수신된 MPEG-2 스트림을 복호하기 위한 디스크램블러, 수신된 MPEG-2 신호를 역다중하기 위한 역다중화기, 그리고 MPEG 복호기 등으로 구성되어 있다. 따라서 STU 는 보안 기능을 수행하기 위하여 하나 이상의 부가적인 보안 모듈(Detachable Security Module) 과 인터페이스한다.

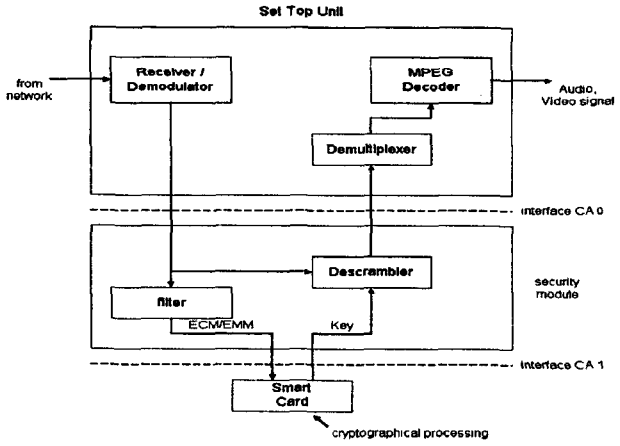

그림 2.2 STU 기능 블럭도

사용자 댁내 장치(Customer Premises Equipment) 는 STU, 그리고 스마트 카드와 디스크램블러로 구성되는 부가적인 보안 장치로 구성될 수 있다. 보안 모듈은 암호 알고리듬에 대한 공격이 성공했을 경우 쉽게 대체될 수 있고, STU 에 가입자 관련 정보를 제거하여 STU 의 일반화가 가능하며, STU 의 쉬운 변경 및 재사용이 가능케 한다. STU 와 보안 모듈간의 인터페이스는 CA0 인터페이스로, 보안 모듈과 스마트 카드간의 인터페이스는 CA1 인터페이스로 정의되어 있다. CA0 는 STU 와 고속의 디스크램블링 기능 및 필터 기능 차원에서 인터페이스하며, 저속의 키 관리 정보 역시 송수신할 수 있어야 한다. CA1 은 저속이며, CA0 인터페이스는 고속이다. CA1 인터페이스는 저속의 보안 관리 기능만을 수행하는 저속 스마트 카드와 인터페이스한다. CA1 인터페이스 호환성 여부는 STU 와 서비스 제공자간에 사용되는 스크램블링 알고리듬의 종류 및 공통의 필터의 적용 여부에 좌우된다. CA1 인터페이스는 다음과 같은 특성이 있다. 디스크램블러, 역다중화기, 복호기를 하나의 칩으로 집적화할 수 있으므로 스마트

카드의 복잡도를 감소시킬 수 있다. 또한 보안 알고리듬의 공격 성공, 새로운 서비스의 도입, 새로운 서비스 제공자로의 액세스 등을 가능케 한다. 그리고 디지틀 정보에 대한 보호가 용이하다는 것이다. CA0 인터페이스는 인터페이스를 통하여 복구된 디지틀 스트림이 전달되므로 도청이 가능하다. 그러나 CA1 인터페이스에서는 복구된 디지틀 스트림이 STU 내에 포함되므로 사용자가 도청하는 것을 어렵게 한다. 만약 디스크램블링 기능과 복호가 하나의 칩에서 수행된다면 디지틀 스트림에 대한 보호는 더욱 용이해진다.

2.4 개체들간의 요구사항

각 개체간에 설정된 정보에 대한 보안 요구 사항은 매우 중요하다. DAVIC 에서의 개체들은 그림 2.3과 같이 서비스 제공자(Service Provider), 논리적 망(Logical Network), 그리고 서비스 사용자(Service User) 로 구성된다. 서비스 제공자의 정보원과 사용자의 정보싱크간의 S1 정보 흐름에 대한 보안 요구 사항(R1) 은 제 삼자에 의한 서비스 제공자 또는 서비스 사용자를 가장하는 것을 방지해야 하고, 내용 정보 및 키 정보에 대한 도청을 예방해야 하며, 전달된 스트림의 비인가된 액세스에 대해 보호해야 하고, 다운로드된 SW 의 비인가된 변경에 대한 보호 및 내용의 비인가된 복사에 대한 보호, 그리고 서비스 제공자에 의한 거부에 대한 대책이 있어야 한다.

서비스 제공자의 종점간 제어 개체와 서비스 사용자의 종점간 제어 개체간의 S2 정보 흐름에 대한 보안 요구 사항(R2) 은 서비스 제공자나 서비스 사용자의 가장에 대한 보호, 민감한 서비스 제공자와 사용자간의 대화에 대한 도청 예방, 서버 개체의 비인가된 액세스에 대한 보호, 다운로드된 SW 의 비인가된 변경에 대한 보호, 위조에 대한 보호, 그리고 서비스 사용자에 의한 거부에 대한 대책이 있어야 한다.

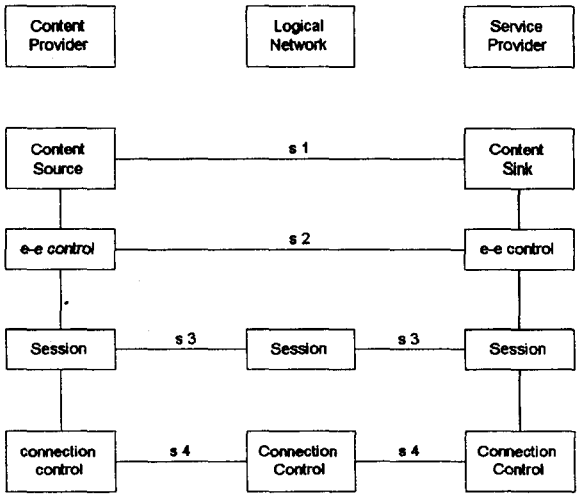

그림 2.3 DAVIC 개체들

2.5 S2 또는 S3 상에서의 보안 메카니듬

S2 정보 흐름상에서의 통신을 보호하기 위한 보안 메카니듬은 인증 메카니듬, 무결성, 그리고 암호 메카니듬이 있다. 인증 방식은 ITU X.509 에서 표준화된 삼방향 인증 메카니듬을 이용한다. 이는 상대 개체를 서로 인증하는 양방향 인증 방식이며, 동기화된 클럭을 제거할 수 있으며, 인증 프로토콜 수행과 동시에 추후의 무결성 및 암호 통신을 위한 세션키를 공유할 수 있는 방식이다.[2,3,4,5,6]

인증 방식에서 이용되는 정보 채널의 구조는 DSM-CC 사용자간 메시지에서의 AuthRequest_T 구조를 사용한다. 인증 과정의 개시는 사용자에 의해 수행된다. 이에 응하여 서버는 인증이 필요치 않을 경우, 사용자로 No-Auth exception 정보를 이용한다. 이는 서버에서 사용자로 향하는 No-Auth exception 이며, 이의 파라메타에는 일방향, 이방향, 그리고 삼방향 인증을 나타내는 인증 방식의 종류 정보와, RSA, ElGamal, FS, 그리고 GQ 등의 채용한 서

명 기법을 나타내는 서명 방식의 종류 정보가 포함된다.

2.6 S2 또는 S3 상에서의 인증 프로토콜과 변수

DAVIC 인증을 위한 참조 모델은 그림 2.4와 같다. 인증 정보 교환을 위한 흐름은 DSM-CC U-U 정보를 이용하여 실현된다. 여기서 이용될 수 있는 알고리듬은 RSA 서명 알고리듬, GQ의 서명 알고리듬, 또는 그외의 영지식 서명 알고리듬이 있다.

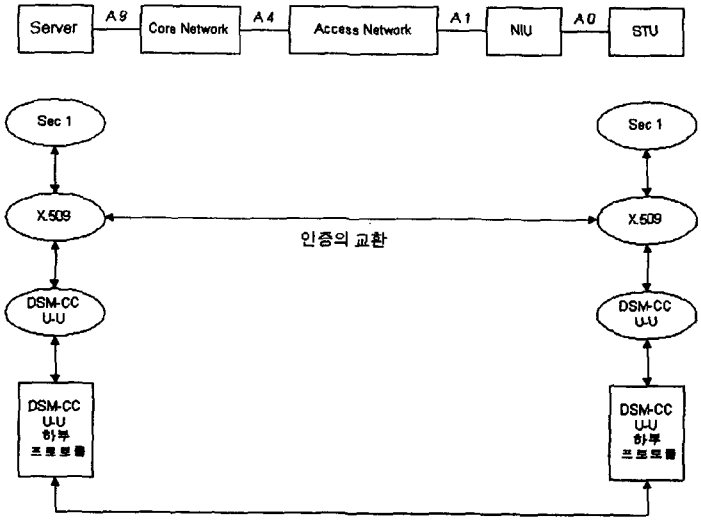

그림 2.4 인증 참조 모델

S2 상에서의 ITU X.509 인증 과정을 이용한 인증 프로토콜은 그림 2.5와 같다. 인증 프로토콜은 넌스(Nonce) 와 공개키 암호 시스템에 바탕을 두고 있다.

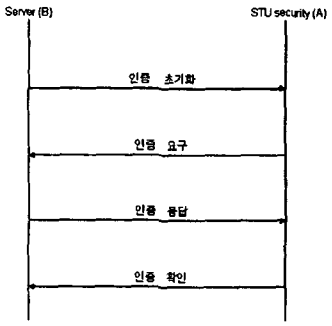

그림 2.5 인증을 위한 정보 흐름

인증 프로토콜은 인증을 받는 사용자 또는 증명자(Prover) 와 이를 검증하는 서버 또는 검증자(Verifier) 간에 수행된다. 인증 프로토콜에서 이용되는 인증 정보는 그림 2.5와 같이 크게 인증 초기화(Authentication Initialization), 인증 요구(ARQ : Authentication Request), 인증 응답(ARS : Authentication Response), 인증 확인(AC : Authentication Confirm), 그리고 선택적으로 인증 성공 확신(CONFIRM-AP) 으로 구성된다. 이외 장애 상태의 발

생을 상대국에 전달하기 위한 FAULT 정보가 있다. 인증 초기화 정보는 인증 과정을 개시하기 위하여 서버에서 사용자로 전달되는 인증 정보이다. 인증 요구 정보는 사용자가 서버로 사용자의 이름 A, 사용자의 공개키 증명서 C_A, 추후 암호 통신에 이용될 무결성 알고리듬의 종류를 나타내는 IntAlgs 와 암호 알고리듬의 종류를 나타내는 ConfAlgs, 서버의 이름 B, 선택적으로 타임 스템프 T_A, 그리고 사용자 넌스 N_A 를 사용자의 비밀키로 암호화한 파라메타를 전달한다. 인증 요구 정보는 식 (2.1)과 같다.

$$ARQ : A, D_{Sa}[B, N_A, T_A(Opt.), IntAlgs, ConfAlgs], C_A \quad (2.1)$$

ARQ 를 수신한 서버는 사용자의 C_A 의 유효성을 확인하고 이로 부터 서명용 공개키를 복구한 후, 서명용 공개키를 이용하여 D_{Sa}[B, N_A, T_A(Opt.), IntAlgs, ConfAlgs] 를 복호하여 B, N_A, T_A(Opt.), IntAlgs, ConfAlgs 를 복구한다. 그리고 자신으로 향하는 정보인가를 B 로 확인하고, 넌스 N_A 를 추후의 ARS 를 위해 저장한다.

인증 응답 정보는 서버에서 사용자로 향하는 정보 흐름이다. ARS 는 서버의 이름 B과 서버의 공개키 증명서 C_B, 그리고 사용자로 부터 제안된 내용을 바탕으로 선택된 추후의 무결성 알고리듬인 IntAlgs 과 암호 알고리듬인 ConfAlgs, 사용자의 이름 A, 선택적으로 타임 스템프 T_B, 자신의 기밀성용 공개키로 추후 무결성 서비스를 위한 세션키 K^I_{AB}를 암호화한 $E_{Pa}[K^I_{AB}]$, 자신의 기밀성용 공개키로 추후 기밀성 서비스를 위한 세션키 K^C_{AB} 를 암호화한 $E_{Pa}[K^C_{AB}]$, 그리고 서버의 Nonce N_B 를 서버의 서명용 비밀키로 서명한 서명문을 묶어서 전달한다. 인증 응답 정보는 식 (2.2)와 같다.

$$ARS : B, D_{Sb}[A, N_A, N_B, T_B(Opt), IntAlgs, ConfAlgs, E_{Pa}[K^I_{AB}], E_{Pa}[K^C_{AB}]], C_B \quad (2.2)$$

ARS 를 수신한 사용자는 C_B 의 유효성을 검증하고 서버의 서명용 공개키를 구한 후, 이를 이용하여 D_{Sb}[A,N_A,N_B, T_B(Opt),IntAlgs,ConfAlgs,$E_{Pa}[K^I_{AB}]$,$E_{Pa}[K^C_{AB}]$] 를 복호하여, A,N_A,N_B,T_B(Opt),IntAlgs,ConfAlgs, $E_{Pa}[K^I_{AB}]$, $E_{Pa}[K^C_{AB}]$ 를 복구한다. 복구된 A 를 이용하여 자신으로 향한 정보인가를 확인하고, 수신된 N_A 와 송신된 N_A 가 동일한가를 확인함으로서 서버의 정체를 인증하고, 협상된 무결성 알고리듬과 기밀성 알고리듬의 종류를 확인한다. 그리고 $E_{Pa}[K^I_{AB}]$ 와 $E_{Pa}[K^C_{AB}]$ 를 자신의 비밀키를 이용하여 서버에서 전송된 무결성 서비스용 키 K^I_{AB} 와 기밀성용 키 K^C_{AB} 를 복구한다.

인증 확인 정보는 사용자에서 서버로 향하는 정보로서, 사용자의 이름 A 와 서버의 이름인 B 및 서버로 부터 수신된 넌스 N_B 를 사용자의 서명용 비밀키로 서명한 결과를 전달한다. 상기 과정들을 수행함으로서 양방 인증이 수행될 수 있다. 인증 확인 정보는 식 (2.3)과 같다.

$$AC : A, D_{Sa}[B, N_B] \quad (2.3)$$

표 2.2 개체가 유지해야 할 인증 관련 타이머

타이머	유지 객체 및 값	용도
T100	증명자, 3초	. 증명자가 ARQ 를 보내고 ARS 를 수신할때 까지 기다리는 시간 . 개시자는 타이머가 해제되면 I-ARQ-Retry-Count 값이 I-Max-ARQ-Retry 를 넘지 않는 다면 ARQ 를 재전송
T101	증명자, 9초	. 증명자가 ARS 수신하고, AC 를 전송했으나, 응답자가 AC 를 수신하지 못해, 계속 ARS 를 송신하고 있는 상태 . 개시자는 T101 타이머가 해제되기 이전에 ARS 를 수신하면 반복적으로 AC 를 송신해야 함
T102	검증자, 3초	. 검증자가 ARS 를 보내고 AC 를 수신할때 까지 기다리는 시간 . 응답자는 타이머가 해제되면 R-ARS-Retry-Count 값이 R-Max-ARS-Retry 를 넘지 않는 다면 ARS 를 재전송

AC 를 수신한 서버는 B 를 확인함으로서 자신으로 향하는 정보인가를 판단하고, 복구된 N_B 와 송신된 넌스 N_B 가 같은가를 확인함으로서 사용자를 인증한다. 그리고 인증 과정을 신속히 수행하기 위하여 서버는 사용자로 CONFIRM-AP(Authentication Protocol) 정보를 전송한다. 이는 필수 인증 과정은 아니나 인증 과정을 빨리 수행하는 것을 가능케 한다. FAULT 정보는 장애를 검출한 객체가 장애의 원인을 상대방에 전달하기 위한 정보이다. 인증 과정을 수행하기 위해 요구되는 타이머 관련 변수는 표 2.2와 같은 3가지 종류가 있다. 또한 인증과 관련해 유지해야 할 계수기와 관련 변수는 표 2.3과 같다.

표 2.3 인증 관련 변수

인증 관련 변수	유지 객체 및 값	용도
I-ARQ-Retry-Count	증명자	. T101 과 연동됨 . ARQ 가 보내진 개수 . 최대값 : I-Max-ARQ-Retry
R-ARS-Retry-Count	검증자	. T102 과 연동됨 . ARS 가 보내진 개수 . 최대값 : R-Max-ARS-Retry
I-Max-ARQ-RETRY	증명자, 2	. ARQ 가 송신되는 최대 횟수
R-Max-ARS-RETRY	검증자, 2	. ARQ 가 송신되는 최대 횟수

증명자측에서 인증 절차는 그림 2.6과 같고 검증자측에서 인증 절차는 그림 2.7과 같다.

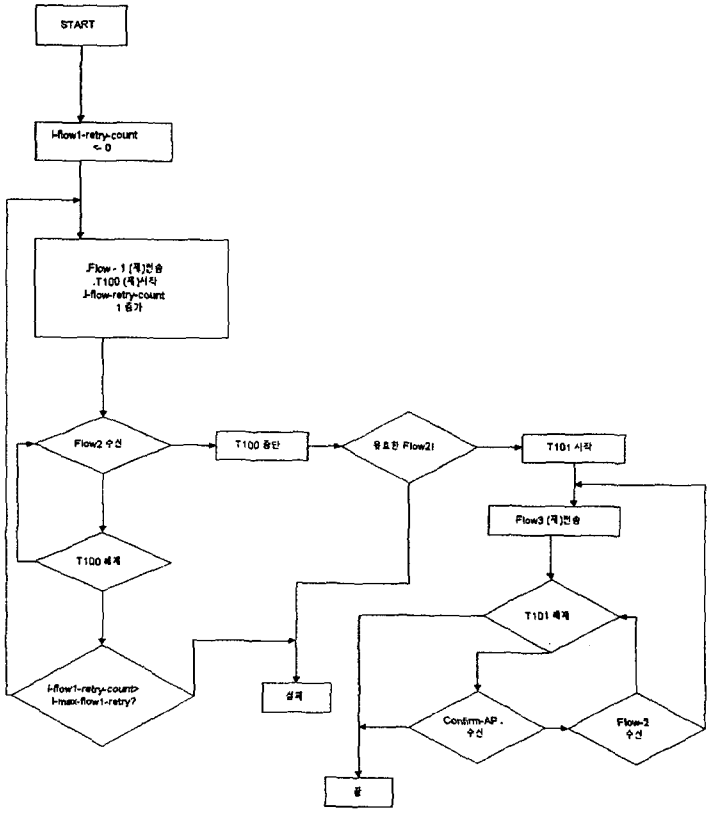

그림 2.6 증명자측에서 인증 절차

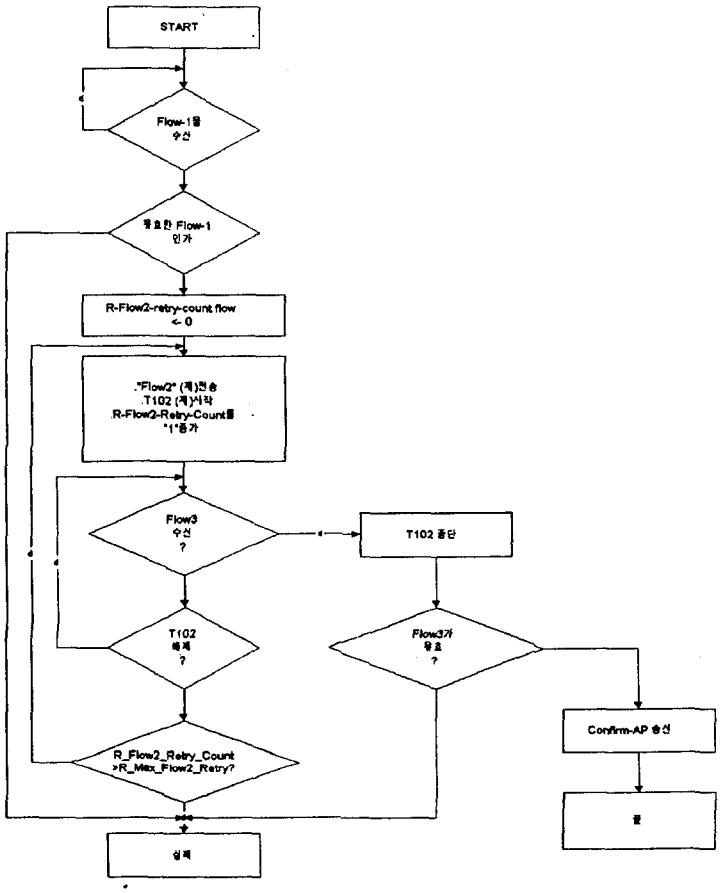

그림 2.7 검증자측에서 인증 절차

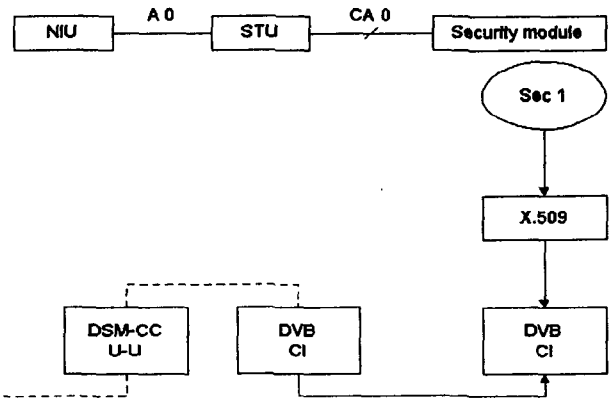

그림 2.8 CA0 인터페이스를 갖는 STU 와 보안 모듈간의 프로토콜 스택

그림 2.8은 CA0 인터페이스를 갖는 STU 와 보안 모듈간의 프로토콜 스택을 나타내고 있으며, 그림 2.9는 CA1 인터페이스를 갖는 STU 와 스마트 카드간의 프로토콜 스택을 나타내고 있다. 기본적으로 국내에서는 CA1 인터페이스를 갖는 인증 참조 모델을 채용하는 것이 현실적이라 판단된다.

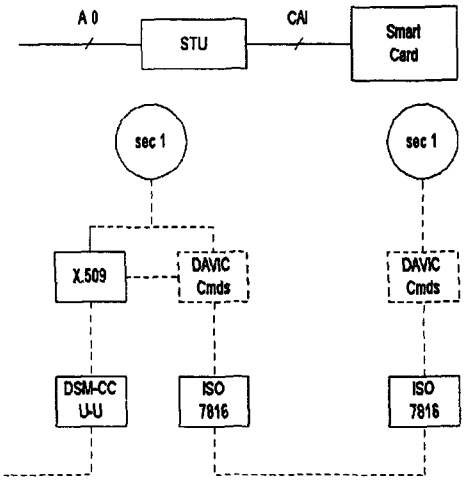

그림 2.9 CA1 인터페이스를 갖는 STU 와 스마트 카드간의 프로토콜 스택

2.7 VOD 를 위한 인증 프로토콜 설정 및 스택

VOD 서비스는 네트워크 계층에서 수행되는 S3 상에서의 인증과 서버와 STU 간의 종점간에 수행되는 S2 상에서의 인증을 포함해야 한다. S3 또는 S4 정보 흐름을 이용한 사용자(Client) 과 SRM(Session and Resource Manager) 간의 인증을 위해 개입되는 프로토콜 스택은 그림 2.10과 같다. 인증은 UDP(User Datagram Protocol) 상위 계층에서 실현된다.

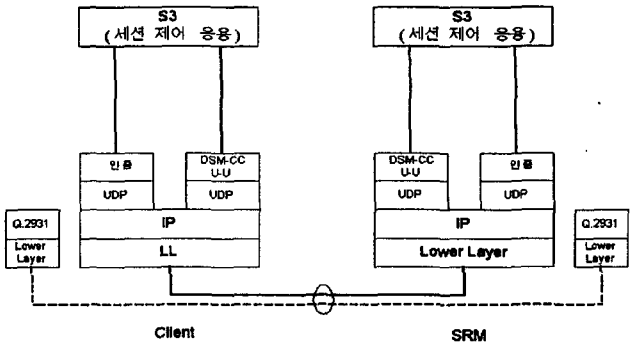

그림 2.10 고객과 SRM 간의 인증을 위한 프로토콜 스택

또한 SRM 과 서버간의 인증을 위해 개입되는 프로토콜 스택은 그림 2.11과 같다. 이는 TCP 상위 계층에서 수행된다.

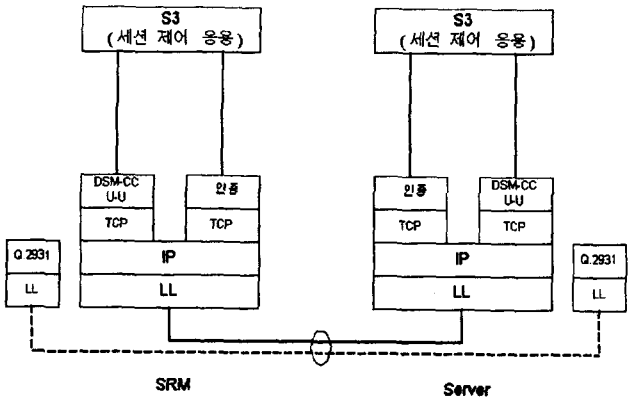

그림 2.11 SRM과 서버간의 인증을 위한 프로토콜 스택

또한 고객측에서의 인증을 위한 프로토콜 스택은 그림 2.12와 같다. 그림 2.10 에서 그림 2.12 에서 알 수 있듯이 인증은 TCP 또는 UDP 상위 계층에서 수행됨을 알 수 있다. 따라서 TCP/UDP 계층과의 인증 정보 교환은 기존의 프리미티브를 이용하여 실현될 수 있다.

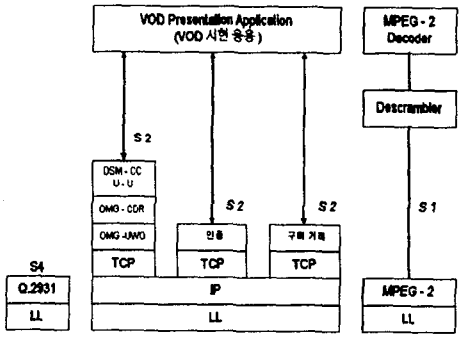

그림 2.12 고객측에서의 인증을 위한 프로토콜 스택

VOD 세션 서비스를 위한 전체 과정은 그림 2.13과 같다. 그림 2.13의 첫 단계에서는 고객과 SRM 간의 세션 설정을 위한 과정이다, 고객은 SRM 으로 UNConfigRequest 정보를 전송하고, SRM 은 고객으로 UNConfigConfirm 를 전송함으로서, 고객과 SRM 은 서로의 NSAP(Network Service Access Point) 와 IP 연결을 확인한다. 이후, UDP 를 통한 인증 정보를 교환함으로서, 고객과 SRM 간의 인증 기능을 수행한다. UDP 는 인증 정보 교환을 위한 DSM-CC UN 정보의 전달 메카니듬을 제공한다. 그리고 SRM 과 고객은 서비스를 선택하기 위한 다운로드 과정과 협상 과정을 수행한다. 이를 완수하면 고객은 서비스를 선택하고 SRM 을 통하여 서버로 연결되기 위한 CientSessionSetupRequest 을 SRM 에 전달한다. 이를 수신한 SRM 은 서버와 인증 정보를 교환하여 서로의 신분을 확인한 후, 세션 협상 과정을 수행한다. 이를 위하여 SRM 은 서버로 ServerSessionSetupIndication 정보를 전송하고, 서버는 SRM 으로 ServerAddResourceRequest 를 전송하며, SRM 은 서버로 ServerAddResourceConfirm 을 전송하며, 서버는 SRM 으로 ServerSessionSetupResponse 을 전송한다. 이를 수신한 SRM 은 고객으로 ClientSessionSetupConfirm 을 전송함으로서, 서버와 고객은 SRM 의 중재로 상대방의 NSAP 을 알고 서로의 IP 연결을 설정한다. 여기서 여러개의 TCP 연결이 인증, 구매 요구, DSM-CC UU 정보 교환을 위하여 설정되며, S1

정보 교환을 위한 MPEG-2 TS 가 설정된다. 이후 고객과 서버는 서로의 신분을 확인하고 추후의 암호 통신을 위한 세션키 구축을 위한 세션키 공유를 위한 인증 과정을 수행하며, VOD 서비스가 최종적으로 제공하기 이전에 수행되어야 할 구매 대화, 그리고 S1 정보를 통한 영화 내용 교환, 그리고 DSM-CC U-U 구조를 이용한 비디오 및 오디오 제어 신호의 교환 과정 등을 수행한다. VOD 서비스가 완료되면 고객은 SRM 으로 ClientReleaseRequest 를 전송하고, 이를 수신한 SRM 은 ServerReleaseIndication 을 서버로 전달하며, 서버는 ServerReleaseResponse 을 SRM 으로 전달하며, 마지막으로 SRM은 고객에 ClientReleaseConfirm 을 전달한다.

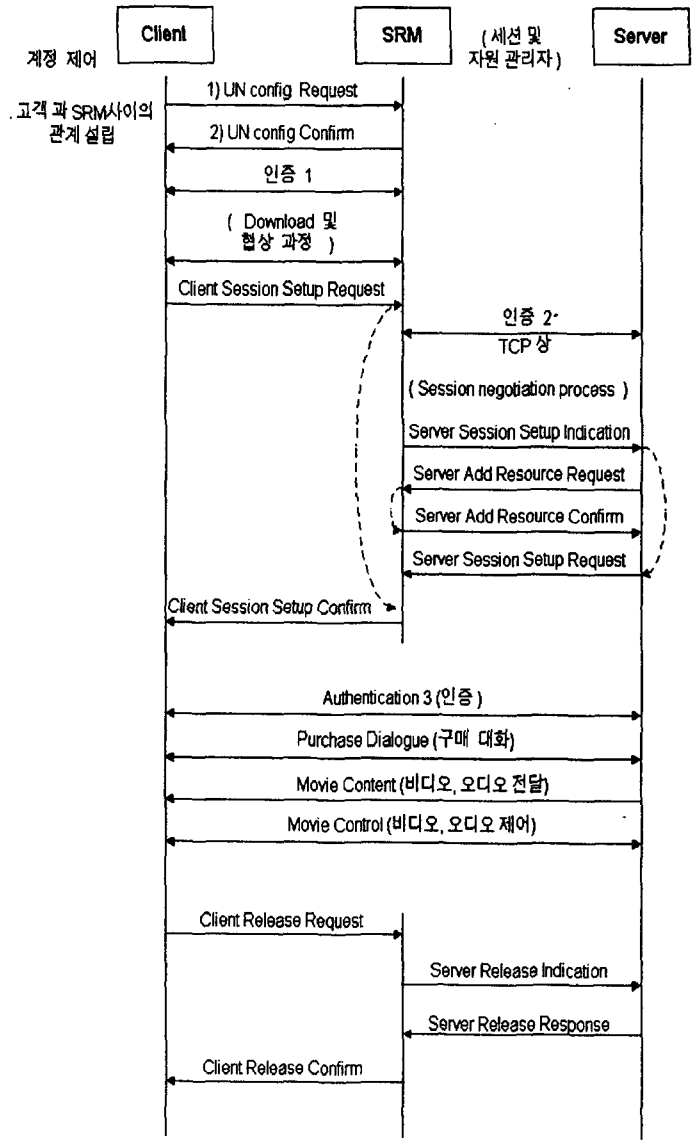

그림 2.13 VOD 세션에 응용된 예

2.8 공개키 증명서

개체 A 의 공개키 증명서는 모든 객체들이 믿을 수 있는 인증센타 (CA:Certification Authority) 가 발행하며, CA 는 모든 개체가 신뢰할 수 있는 개체(Entity) 이다. 공개키 증명서의 구성 정보는 사용자의 공개키와 사용자의 구별 가능한 이름, 공개키 증명서의 유효 기간 등이며, 공개키 증명서는 기본적으로 ID 와 PKA 를 CA 의 비밀키로 서명한 D_{CAs}(IDA,PKA) 이다. 공개키 증명서의 검증은 CA 의 서명용 공개 정보를 이용하여 수행되며, CA 의 공개 정보는 각 사용자가 변경 불가능한 영역에 보관해야 한다. CA의 공개 정보의 누출은 CA 에 의해 검증된 사용자간의 정보의 무결성에 심각한 영향을 미치기 때문이다. 따라서 CA 의 공개 정보 저장 매체는 스마트 카드가 바람직한다. 공개키 증명서의 일반적 구조는 다음과 같다.

```
DAVIC-Certificate ::= SEQUENCE
    { to be signed SEQUENCE {
                version                 INTEGER         DEFAULT 1,
                certificateNumber       PrintableString,
                namingAuthority         Name,
                signatureAlgorithm      AlgorithmIdentifier,
                issuer                  Name,
                timeOfIssue             UTCTime,
                validity                Validity,
                subject                 Name,
                firstSubjectKeyInfo     PublicKeyInfo,
                secondSubjectKeyInfo    PublicKeyInfo   OPTIONAL,
                                }
                signature               OCTET STRING
    }

Validity ::= SEQUENCE {
                notBefore       UTCTim
                notAfter        UTCTim
                        }

PubKeyInfo ::= SEQUENCE {
                subjectAlgorithm        AlgorithmIdentifier,
                subjectPublicKey        OCTET STRING
                                }
```

버전은 DAVIC 공개키 증명서 버전의 번호이고, certificateNumber 는 알파벳을 허용하기 위한 printablestring 이며, 공개키 증명서를 유일하게 확인하기 위한 식별자이다. namingAuthority 는 공개키 증명서의 소유자가 속해있는 보안 영역을 책임지는 기관의 이름이다. signatureAlgorithm 은 CA 가 서명문 생성을 위해 사용한 알고리듬이다. issuer 는 공개키 증명서 발행자의 유일한 이름이다. timeOfIssue 는 공개키 증명서가 발행되는 시간을 의미한다. validity 는 공개키 증명서의 유효 기간을 나타내며, 유효 시작 일시와 유효 종료 일시를 포함한다. subject는 공개키 증명서 소유주의 DistinguishedName 이다. firstSubjectKeyInfo 와 secondSubjectKeyInfo 는 이용되는 보안 알고리듬의 종류와 각 보안 서비스를 위한 공개키 값을 포함하고 있다. signature 는 상기 구성 정보를 CA 의 서명용 비밀키로 서명한 결과이다.

개체 A 가 믿는 CA 를 CA_A 라 하고, 개체 B 가 믿는 CA 를 CA_B 라 하면, 두 개체에 대한 인증 센타들이 서로 다를 경우, 각 CA 가 공통적으로 믿을 수 있는 상위의 CA 를 구한 후, 각 개체는 공통의 CA 을 통해 상대방의 공개키 증명서를 얻는다. 공개키 증명서는 위조를 불가능하게 하기 위해 인증 센타 CA 의 비밀 서명키로 서명한 개체의 ID 와 공개키를 의미하며 그 형식은 식 (2.4) 와 같다.

$$Cert(A) = \{I, CAS[h(I)]\} \quad (2.4)$$

여기서, I 는 공개키 증명서의 구성 정보이고, h(I) 는 정보 I 에 대한 해쉬 결과 값이며, CAS[h(I)] 는 데이타 블럭 h(I) 를 CA 의 비밀키로 서명된 내용, h(I) 는 데이타 블럭 I 를 일방향 해쉬 함수에 대입하여 얻은 결과이다. X.509 공개키 증명서는 공개키 증명서의 유효성을 확인하기 위해 식 (2.5) 의 만족 여부를 확인하고 validity 가 최종 시간 내에 있는가를 확인한 후 검증된다.

$$h(I) \quad =? \ CAP\{CAS[h(I)]\} \quad (2.5)$$

식 (2.5) 의 좌측은 CA(A) 의 좌측에 있는 I 를 일방향 해쉬함수 h 에 직접 대입하여 얻은 결과이고, 우측은 CA(A) 의 우측에 있는 CAS[h(I)] 를 CA 의 공개키로 복호한 결과이다.

2.9 VOD 인증 방식 선정

국내 VOD 시스템에서의 인증 방식은 다음과 같은 원칙으로 실현되어야 한다.

첫째, 인증 프로토콜은 ITU X.509 의 인증 방식을 이용한다. 즉 공개키 증명서와 공개키 서명 메카니듬을 이용한 삼방향 인증 방식을 채용함으로서, 정교한 망 클럭의 필요성을 제거한다. 둘째, 인증용 서명 알고리듬은 특정의 알고리듬을 지정하여 사용치 않고 인증 객체간에 협상에 의해 수행되도록 한다. 이는 인증 정보인 인증 초기화 정보를 통해 서버에서 고객으로 전달된다. 셋째, 인증 프로토콜을 수행함으로서, VOD 세션을 위한 기밀성용 세션키와 무결성용 세션키를 얻을 수 있는 인증 정보를 이용토록 한다. 넷째, 고객은 서버의 공개키 증명서를 반드시 스마트 카드에 저장해야 한다. 다섯째, 인증은 종점간 인증으로 스마트 카드와 서버간에 수행되도록 하며, 망차원의 인증 역시 선택적으로 실현되도록 한다. 여섯째, 인증 정보는 기본적으로 DSM-CC UU 정보를 통해 전달되며, TCP 또는 IP 상위 계층에서 수행되도록 한다. 일곱째, 공개키 증명서의 서명 알고리듬은 기본적으로 RSA 서명 알고리듬을 이용토록 하며, 이 경우 최소 512 비트 이상의 비트 길이를 갖도록 한다. 여덟째, 인증 과정은 관련 타이머와 계수기를 유지함으로서 안전한 인증 정보 교환을 가능케 해야 한다. 아홉째, 고객측 인증 정보는 기본적으로 스마트 카드에서 생성되어야 하며, CA1 인터페이스의 귀환 채널을 이용한다.

제3장 공개키 증명서와 인증 프로토콜의 실현

3.1 공개키 증명서 방식의 실현

인증 프로토콜에서 이용되는 각 사용자의 공개키 증명서를 C 언어를 이용하여 실현하였다. 이를 위해 구현된 C 루틴은 512 비트 이상 1024 비트 이하의 모듈러 지수 연산 루틴, 128 비트의 MD5 해쉬 루틴, 그리고 안전한 RSA 키 생성 루틴이다. 또한 공개키 증명서 생성 루틴과 공개키 검증 루틴을 관련 보안 메카니듬을 이용하여 실현하였다. 공개키 증명서용 보안 메카니듬은 서명용으로 RSA 서명 메카니듬을, 해쉬용으로 MD5 해쉬 함수를 이용하였다. 곱셈 연산은 속도가 빠른 고전적인 알고리듬을 이용하였으며, 지수 연산 알고리듬은 768 비트에 대해 윈도우의 크기가 6인 윈도우 방식을 이용하였다. 지수 연산을 위한 지수 연산 알고리듬은 IBM PC 586 100MHz에서 약 0.7 초 이내로 구현되었다. 이는 실제 응용 프로그램에 적용 가능한 결과이다. 공개키 증명서에 이용될 수 있는 서명 알고리듬은 표 3.1과 같다.[7,8,9]

표 3.1 공개키 증명서용 서명 알고리듬과 무결성 알고리듬

코 드	의 미	설 명	코 드	의 미	설 명
SA	-	서명 알고리듬	HF	-	해쉬 함수
11	RSA	RSA 서명 알고리듬	51	HDS1	n-비트 블럭 암호, single length
12	DSS	DSS 서명 알고리듬	52	HDS2	n-비트 블럭 암호,double length
13	FS	FS 영지식 서명 알고리듬	53	SQM	RSA의 Square mod n 해쉬
14	GQ	GQ 영지식 서명 알고리듬	54	MDC2	수정 감시 코드, IBM 시스템
15	BM	BM 영지식 서명 알고리듬	57	BGC7	BGC-7.1 해쉬 기능
			58	MD4	MD-4 RSA (1990)
			59	MD5	MD-5 RSA (1991)
			60	SHA	Secure Hashing Algorithm

공개키 증명서를 생성하는 CA 의 RSA 서명용 공개키는 식 (3.1) 과 같고, 비밀키는 식 (3.2) 와 같다.

n = d3f4 84b8 23ce 7035 e7ab 3230 4313 ffff 1b26 c2f8 7fe6 e50f 229a ed80 13d8 1d95 b430 87f0
b5ab 50de b1d8 82b2 42a9 6733 d9e5 c2b1 2a0d 4258 9463 0b11 0ea0 5596 b1cf c07f 1497 46bc
4413 4b5e 4ade 46ce ebc8 b635 8a66 9b33 c223 75f5 c869 7965

e = 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001 (3.1)

d = 9340 ceb0 0b98 5196 f1b3 9b73 b236 83e4 dd1d 29d9 712a 9dba cd24 eb99 ceaf 97a6 3900 7a81
3e05 9e72 8906 4d8e c107 663d 78e9 677d abe5 e8f3 4c70 4e5c fefb 1525 8f37 54ed da01 8ec8
defe 60f5 a1bd e86c 4fc5 7161 a66f a755 c929 d8ca 8bf1 c585 (3.2)

두 개체를 A, B 라고 가정하면 사용자 A 의 공개키 증명서 생성을 위한 공개 정보는 식 (3.3) 과 같다. CA 에 의해 생성되는 디지틀 서명문이다. Naming 타입은 현재 구체적으로 정의되어 있지 않고 있으므로 인터넷 IP 주소를 이용하였다. 공개키 증명서를 실현한 예는 다음과 같다.

```
DAVIC-Certificate  ::= SEQUENCE
{     to be signed SEQUENCE {
         version                          1,
         certificateNumber                1000,
         namingAuthority                  etri.re.kr,
         signatureAlgorithm               RSAwithMD5,
         issuer                           sch.ac.kr,
         timeOfIssue                      199610010000,
         validity  ::= SEQUENCE {
                 notBefore                199610010000,
                 notAfter                 200001010000
                              }
         subject                          sangpil@asan.sch.ac.kr,
         firstPubKeyInfo ::= SEQUENCE {
```

```
            subjectAlgorithm        ConfAlgwithRSA,
            subjectPublicKey        {n,e}
                                    }
    secondPubKeyInfo ::= SEQUENCE {
            subjectAlgorithm        AuthAlgwithRSA,
            subjectPublicKey        {n,e}
                                            }
                        }
    signature       signature value
}
```

(3.3)

이를 MD5 해쉬한 값은 식 (3.4) 와 같으며, 이 결과를 식 (3.2) 와 같은 CA 의 서명용 비밀키로 서명한 결과는 식 (35) 와 같다.

h(I) = 32cb 05e8 22ee d912 6f8f 378b 3f75 34b3 (3.4)

$D_S(h(I))$ = 7887 13d7 d13d 8a05 1437 5ac1 44e2 70ce 9888 5c87 1842 6ee4 7b0c 75eb d63f 3371
36d8 20c8 36cb 041f ebeb fa4c aac2 5ee3 aead 2a6e c1b6 93f2 3cb9 31bc ca4c bd06
1fc3 f80d 5ee8 8f7e e9bf 0bc5 0d71 d5e7 d884 c1bd 9def 3323 52b9 5cfb 2827 8a76
(3.5)

사용자 B 의 공개키 증명서도 같은 방법으로 생성되었다.

3.2 인증 프로토콜의 실현

인증 방식은 ITU X.509의 삼방향 인증 프로토콜과 RSA 서명 알고리듬에 바탕을 둔다. 인증 프로토콜을 위하여 사용자 A 는 고객 또는 증명자(Prover) 로, 사용자 B 는 서버 또는 검증자(Verifier) 로 가정한다. 시뮬레이션은 필수 구성 요소에 대해서만 수행하였다. 실현된 인증 프로토콜은 공개키 증명서 CA, 서명용 RSA 공개키와 비밀키, 각 사용자의 인증용 RSA 공개키와 비밀키에 바탕을 두고 있다. 고객이 서버로 전송하는 인증 요구 정보 (ARQ) 는 식 (3.6) 과 같다.

A = sangpil.@.asan.sch.ac.kr

$D_{Sa}[B,N_A]$ = 4bb7 9e40 d6c7 1cc5 f8c7 3197 9e53 dd80 1b2c edd0 f762 99af 519f bd58 b003 9459
2789 be84 eb25 2a6f 2645 80b6 3113 9ea9 bcad 3016 4af9 a416 356b 0c7d 1e87 e632
27dd 3d01 b295 45a5 d638 0906 20ec 80a4 91b3 7380 9537 0da4 e7bf db2d 8d63 1001

C_A = 1, 1000, etri.re.kr, RSAwithMD5, sch.ac.kr, 199611010000, 199611010000, 200001010000,
sangpil@asan.sch.ac.kr,
ConfAlgwithRSA
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001

d3f4 84b8 23ce 7035 e7ab 3230 4313 ffff 1b26 c2f8 7fe6 e50f 229a ed80 13d8 1d95 b430
87f0 b5ab 50de b1d8 82b2 42a9 6733 d9e5 c2b1 2a0d 4258 9463 0b11 0ea0 5596 b1cf c07f
1497 46bc 4413 4b5e 4ade 46ce ebc8 b635 8a66 9b33 c223 75f5 c869 7965

AuthAlgwithRSA
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001

d3f4 84b8 23ce 7035 e7ab 3230 4313 ffff 1b26 c2f8 7fe6 e50f 229a ed80 13d8 1d95 b430
87f0 b5ab 50de b1d8 82b2 42a9 6733 d9e5 c2b1 2a0d 4258 9463 0b11 0ea0 5596 b1cf c07f
1497 46bc 4413 4b5e 4ade 46ce ebc8 b635 8a66 9b33 c223 75f5 c869 7965

7887 13d7 d13d 8a05 1437 5ac1 44e2 70ce 9888 5c87 1842 6ee4 7b0c 75eb d63f 3371 36d8
20c8 36cb 041f ebeb fa4c aac2 5ee3 aead 2a6e c1b6 93f2 3cb9 31bc ca4c bd06 1fc3 f80d
5ee8 8f7e e9bf 0bc5 0d71 d5e7 d884 c1bd 9def 3323 52b9 5cfb 2827 8a76 (3.6)

서버가 고객으로 수신한 인증 요구에 응하여 전송하는 인증 응답(ARS) 은 식 (3.7) 과 같다.

B = etri.re.kr

$D_{Sb}[A, N_A, N_B]$ = d75a 2ca2 436c fc70 bc8f 4efa 1641 db0d 178b ef95 722c a97f c6e1 4bf4 e0c8 f2dc
dfc9 d4e4 b3ca 89e9 56c0 90f6 1393 4417 5bb8 2076 d35e f228 9ca9 d657 4d35 a8d5 f7af f431
33ca 894a 25b1 3aa8 4dcd 4dcb e108 ac7b 4036 f7fa 4752 2907 7b4e 4090

C_B = 1, 1001, etri.re.kr, RSAwithMD5, sch.ac.kr, 199611010000, 199611010000, 200001010000, etri.re.kr
ConfAlgwithRSA
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001
dde8 2a39 3748 ee8b d079 5144 35f5 6c6d d26c e98a cd9f 894f 094a 1d09 3684 d6be 6da0 9058
9041 8dc7 f402 44e8 d988 43ef 9c99 edd0 9c76 0a27 9fc7 c283 4e2a d8d3 6513 bc4f 5ab2 9d0a
c184 6303 dc51 04f1 3cd8 c95a f7ef f058 48e2 c59f 4e01 71a5

AuthAlgwithRSA
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0001

dde8 2a39 3748 ee8b d079 5144 35f5 6c6d d26c e98a cd9f 894f 094a 1d09 3684 d6be 6da0 9058
9041 8dc7 f402 44e8 d988 43ef 9c99 edd0 9c76 0a27 9fc7 c283 4e2a d8d3 6513 bc4f 5ab2 9d0a
c184 6303 dc51 04f1 3cd8 c95a f7ef f058 48e2 c59f 4e01 71a5

8fff 18dd 1cc7 42d7 1f01 e850 dd9d ec3c 6d79 04af 54c9 0ebf eeb0 e8d0 9da4 00c2 2935 7869
0fd2 ea9a 9ffa 85af e714 4bed 6199 81ef fc38 9ea5 3567 795c 0431 bb2e 759f 81b6 e3d1 54ba

fb67 0153 182e c2e2 1349 7a24 b538 d177 7006 70b0 f902 b5cb

(3.7)

고객은 식 (3.8) 과 같은 인증 확신 정보를 전송한다.

A = sangpil.@.asan.sch.ac.kr

$D_{SA}[B, N_B]$ = c533 2980 d794 4434 ffdf 76e2 b783 c11b 54f1 ece8 d532 63a5 cc31 3eae 0fdb 0719
a1c5 ebb4 3431 e2cb b09d b192 af14 d610 3133 f649 6387 e40d 7079 7c02 a382 4d9b
daf1 2046 d4a9 be8b 4d7d 2c42 f452 c47e 6739 0d57 963f 863b 32eb 8536 f7f9 d049

(3.8)

제4장 결론

VOD 에서 사용자 서비스 제공은 사용자의 정체 확인한 후 제공되야 한다. 본 논문에서는 VOD 인증 시스템을 실현하기 위하여 DAVIC 에서 권고중인 사용자와 SRM, SRM 과 서버, 그리고 서버와 사용자간 인증 프로토콜 스택을 분석하였다. 인증 프로토콜은 하위 기반 계층으로 TCP/IP 계층을 이용하며, VOD 서비스를 위한 정보 흐름 및 여기서의 인증 과정의 위치를 확인하였다. 그리고 VOD 인증 시스템을 실현하기 위해 요구되는 서버 및 고객 측에서의 인증 절차를 제시하였고, 이를 위한 여러 타이머들(T101,T102,T201) 이 필요함을 확인하였으며, 관련 변수인 각 인증 정보의 최대 재전송 회수 및 이를 위한 계수기의 필요성도 확인하였다. 또한 국내 VOD 시스템에 적용될 수 있는 인증 방식의 기본 원칙을 제시하였다. 그리고 VOD 인증 시스템을 위한 공개키 증명서를 768 비트 크기의 RSA 서명 알고리듬과 128 비트 MD5 해쉬 함수를 이용하여 C 언어로 구현하였다. 또한 VOD 를 위한 공개키 증명서의 구성 정보의 일반적 형태를 제시하였고, 실현된 공개키 증명서가 정상적으로 검증됨을 확인하였다. 또한 ITU X.509 인증 프로토콜을 768 비트 크기의 RSA 서명 메카니듬을 이용하여 실현하였다. 또한 공개키 암호 시스템의 속도에 상당한 영향을 주는 고속 지수 연산 알고리듬을 소프트웨어로 실현하였으며, 이는 약 0.7초 이내에 동작됨을 확인하였다. 결과적으로 VOD 를 위한 공개키 증명서와 인증 프로토콜은 여타의 VOD 시스템과 연동되어 동작될 수 있음을 확인하였다. 본 논문의 결과는 공개키 증명서 생성 시스템 구성시 및 국내 VOD 인증 시스템 실현시 유용하게 활용될 수 있다.

- 참 고 문 헌 -

[1] DAVIC, DAVIC 1.2 Baseline Documents, 1996. 6., Newyork

[2] Man Young Rhee, Cryptography and Secure Communications, Mcgraw-Hill, 1992.

[3] ISO/IEC 13818-2, Information Technology - General Coding of Moving Pictures and Association Audio, Committee Draft, ISO/IEC JTC1/SC29, 1993., Seoul

[4] ISO/IEC IS 9798-3, Entity Authentication Mechanisms-Part 3 : Entity Authentication Using a Public-key Algorithm, ISO, Geneva, Switzerland, 1993.

[5] ITU Rec. X.509, The Directory-Authentication Framework, ITU, Geneva, Switzerland, 1993.

[6] NBS, Data Encryption Standard, FIPS Pub-46, 1977.

[7] A. Fiat, A. Shamir, "How to prove Yourself : Practical Solutions to Identifications and Signature Problems," in Advances in Cryptology Crypto'86, Proceedings, Springer-Verlag, pp.186-194, 1987.

[8] T.Beth, "Efficient Zero Knowledge Identification Scheme for Smart Cards," Proc. Eurocrypt'88, pp.77-84, 1988.

[9] J.Simmons, "An Impersonation-proof Identity Verification Scheme," Advances in Cryptology : Proceedings of Crypto'87, Springer-Verlag, pp.211-215, 1988.

본 연구는 한국전자통신연구소 초고속 정보통신 서비스 기반 기술 과제의 위탁 연구 결과입니다.