通信情報保護學會誌
第5卷 第3號, 1995. 9

X.435 EDI 정보보호 서비스 데이타 구조 분석

Analysis of Data Structure for Secure X.435 EDI System

이 정 현*, 윤 이 중*, 김 대 호*, 이 대 기*

요 약

ITU-T X.435 EDI 시스템에서의 정보보호 서비스는 크게 MHS 정보보호 서비스와 Pedi 정보보호 서비스로 나눌 수 있다. 본 논문에서는 X.435 EDI 정보보호 서비스의 종류를 살펴보고, 이들의 데이타 구조의 분석 뿐만 아니라 정보보호 서비스를 제공하기 위해 사용되는 각 필드들이 의미하는 바를 분석, 정리하였다.

1. 서 론

메세지 처리 시스템(MHS:Message Handling System)의 서비스는 기본적으로 UA(User Agent) 서비스, MS(Message Store) 서비스, MTA(Message Transfer Agent) 서비스, PD(Physical Delivery) 서비스로 분류된다.[1][3] 그중에서 UA가 처리하는 메세지의 종류에 따라 IPM(Interpersonal messaging) 서비스, EDI(Electronic Data Interchange) 서비스 등의 특성에 따른 요구 서비스로 분류된다.[2][7] 따라서 본 논문에서는 안전한 EDI 시스템 설계를 위하여 EDI 시스템의 골격을 이루는 MHS 시스템에서의 정보보호 서비스와 UA가 처리하는 메세지 특성에 따른 EDI 정보보호 서비스를 분석 대상으로 하였고, 이들 서비스를 제공하기 위해 정의된 MHS 프로토콜 상의 데이타 구조와 각 필드들의 기능에 대하여 분석하였다.

본 논문의 구성은 2장에서 EDI 시스템의 기반이 되는 MHS 시스템에서의 정보보호 서비스 데이타 구조를 분석하고, 3장에서는 EDI 시스템에서의 Pedi 정보보호 서비스의 데이타 구조를 분석하였으며, 4장에서 결론을 맺는다.

2. MHS 정보보호 서비스 분석

본 장에서는 EDI 시스템의 기반이 되는 메세지 처리 시스템에 관련한 X.435 EDI 서비스들의 데이타 구조에 대해 분석한다. 이에 해당하는 서비스는 크게 발신처 인증(Origin Authentication) 서비스, 안전한 액세스 관리(Secure Access Management) 서비스, 데이타

* 한국전자통신연구소

본 논문은 한국전기통신공사의 출연금에 의하여 수행한 연구결과입니다.

비밀성(Data Confidentiality) 서비스, 데이타 무결성(Data Integrity) 서비스, 부인봉쇄(Non-Repudiation) 서비스, 메세지 보안 레이블링(Message Security Labelling), 정보보호 관리(Security Management) 서비스를 들 수 있다.[3]

2.1 발신처 인증 서비스

2.1.1 메세지 발신처 인증(Message Origin Authentication)

이 서비스는 메세지의 수신자 또는 메세지를 전달하는 MTA에게 메세지의 발신처를 확인하게 해 주는 두가지 목적으로 사용된다. 메세지의 수신자에게만 즉 end-to-end 사이의 서비스를 제공하기 위해서는 per-recipient의 항목인 Message Argument Integrity Security Element를 사용하고, MTS-user 또는 MTA들에게 서비스를 제공하기 위해서는 per-message 항목인 MessageOriginAuthenticationCheck를 사용한다. 어떤 것을 선택하냐는 보안 정책에 따른다.[4]

- MessageOriginAuthenticationCheck

MessageOriginAuthenticationCheck는 per-message의 필드로서 메세지의 수신자와 메세지의 전달을 담당하는 MTA에게 메세지의 발신처를 인증할 수 있는 수단을 제공하며 이 필드는 메세지의 발신처에 의해서 생성된다.

이 필드는 메세지의 발신처 증명과 메세지의 내용물(content)이 변경되지 않았다는 보장(content integrity)과 메세지와 메세지의 보안 레이블 사이의 관련성을 증명하는데 쓰여진다.

이 필드의 데이타 구조는 다음과 같으며, 이의 기능상 구조를 나타낸 것이 그림 1이다.

- 메세지 토큰(Message Token)

메세지 토큰의 기본 구조[8]는 그림 2와 같다.

메세지 토큰은 per-recipient 필드로, signature algorithm identifier, recipient name, time, signed data, encryption algorithm identifier, encrypted data, signature로 이루어져 있으며, 이는 보안관련 정보를 전달해 주기 위한 것이다. '메세지 토큰은 공개키 알고리즘을 사용하고

```
MessageOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
    algotithm-identifier MessageOriginAuthenticationAlgorithmIdentifier,
    content Content,
    content-identifier ContentIdentifier OPTIONAL
    message-security-label MessageSecurityLabel OPTIONAL }

    MessageOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier(X.509)
    Content ::= OCTET STRING
    ContentIdentifier ::= [APPLICATION 10] PrintableString(SIZE(1..ub-content-id-length))
    message-security-label EXTENSION
            MessageSecurityLabel
            CRITICAL FOR DELIVERY ::= 20
```

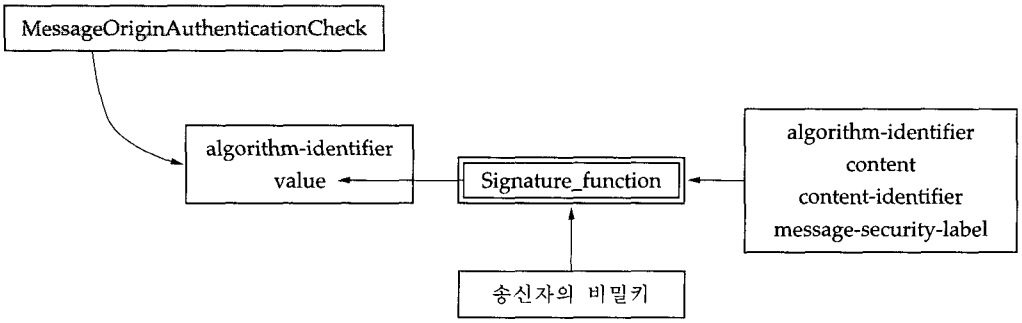


그림 1. MessageOriginAuthenticationCheck

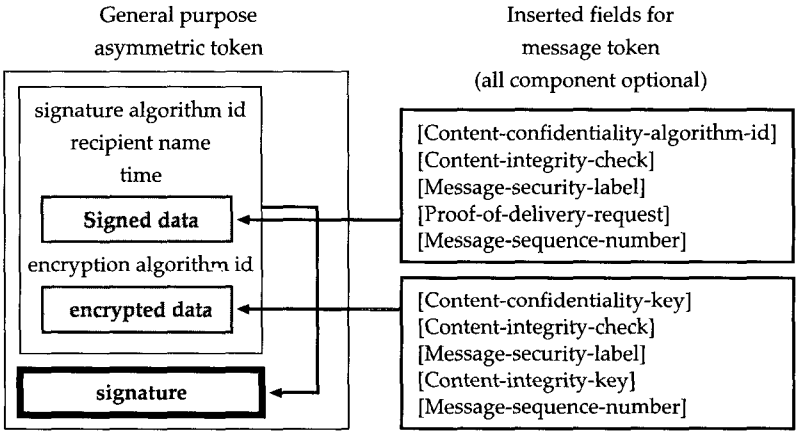


그림 2. 메세지 토큰의 기본 구조

signed data와 encrypted data는 서로 다른 알고리즘을 사용할 수 있다.[4]

메세지 토큰의 signed data에 Content-IntegrityCheck 필드가 포함되어 있다면 발신처 부인봉쇄 서비스를 위해서 third party에게 제공될 수 있다.

메세지 토큰의 데이타 구조는 다음과 같다.

```
AsymmetricToken ::= SIGNED SEQUENCE {
    signature-algorithm-identifier AlgorithmIdentifier,
    name CHOICE {
            recipient-name RecipientName,
            [3] SEQUENCE {
                    global-domain-identifier GlobalDomainIden, fier OPTIONAL,
                    mta-name MTAName }},
    time Time,
    signed-data [0] TokenData OPTIONAL,
```

```
    encryption-algorithm-identifier[1] AlgorithmIdentifier OPTIONAL,
    encryption-data[2] IMPLICIT ENCRYPTED TokenData OPTIONAL }

TokenData ::= SEQUENCE {
    type[0] IMPLICIT TOKEN-DATA,
    value[1] ANY DEFINED BY type }

TOKEN-DATA MACRO ::=
BEGIN
TYPE NOTATION ::= type | empty
VALUE NOTATION ::= value(VALUE INTEGER)
END
```

2.1.2 Probe 발신처 인증(Probe Origin Authentication)

ProbeOriginAuthenticationCheck는 per-message의 필드로 MTA에게 probe의 발신처를 인증할 수 있는 수단을 제공한다. 이 값은 probe의 발신처에 의해서 생성된다. 이 필드는 probe의 발신처를 인증할 수 있는 증거를 제공하고, probe의 message-security-label 필드와 content-identifier 필드 사이의 연계에 대한 증명을 제공한다.

이 필드는 ProbeOriginAuthenticationAlgorithmIdentifier와 SEQUENCE로 정의된 데이타의 asymmetric-encrypted-hashed version으로 구성된다. MTA에서의 검증은 originator-certificate를 통해서 전달된 subject-public-key 필드값을 이용하여 수행한다.

이 필드의 데이타 구조는 다음과 같고, 이의 기능상 구조를 나타낸 것이 그림 3 이다.

```
ProbeOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
    algotithm-identifier ProbeOriginAuthenticationAlgorithmIdentifier,
    content-identifier ContentIdentifier OPRIONAL
    message-security-label MessageSecurityLabel OPTIONAL }

ProbeOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier(X.509)
```

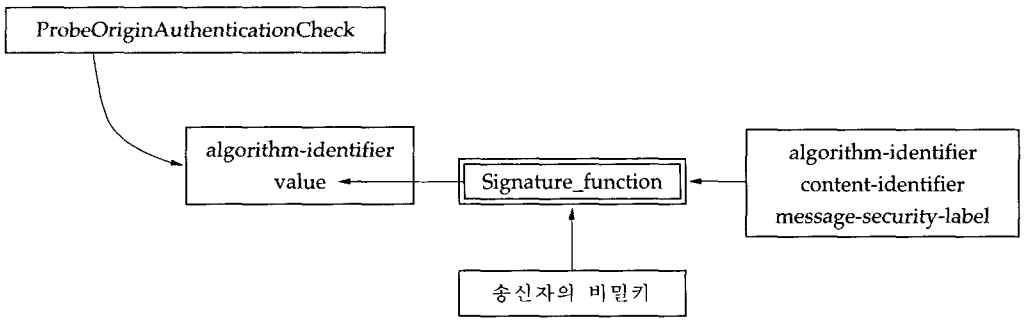


그림 3. ProbeOriginAuthenticationCheck

2.1.3 Report 발신처 인증(Report Origin Authentication)

ReportOriginAuthenticationCheck는 per-message의 필드로 메세지 또는 probe의 발신처와 메세지가 전달되는 MTA에게 report의 발신처를 인증할 수 있게 해준다. 이 값은 reporting-MTA에 의해서 생성된다.

이 필드는 report의 발신처에 대한 증명과 report와 message-security-label 사이의 관계에 대한 증명을 제공한다.

이 필드는 ReportOriginAuthenticationAlgorithmIdentifier와 SEQUENCE로 정의된 데이타의 asymmetric-encrypted-hashed version으로 구성된다.

이 필드의 데이타 구조는 다음과 같고, 이의 기능상 구조를 나타낸 것이 그림 4이다.

2.1.4 제출 증거(Proof of Submission)

이 서비스를 이용하여, 메세지를 제출할 때 제출 사실에 대한 증명을 요구할 수 있다. 이를 위해서 메세지의 제출시에 per-message 필드인 ProofofSubmissionRequest를 이용하여 메세지를 수신한 MTA에게 수신 사실 확인을 요구해야 하고 그 증거를 제출 결과 반환 필드 중의 하나인 ProofofSubmission으로 받는다.

```
ReportOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
    algotithm-identifier ReportOriginAuthenticationAlgorithmIdentifier,
    content-identifier ContentIdentifier OPTIONAL
    message-security-label MessageSecurityLabel OPTIONAL
    per-recipient SEQUENCE SIZE(1..ub-recipient) OF PerRecipientReportFields }

ReportOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier(X.509)

PerRecipientReportFields ::= SEQUENCE {
    actual-recipient-name ActualRecipientName,
    originally-intended-recipient-name OriginallyIntendedRecipientName OPTIONAL
    CHOICE {
            delivery[0] PerRecipientDeliveryReportFields,
            non-delivery[1] PerRecipientNonDeliveryReportFields}}

PerRecipientDeliveryReportFields ::= SEQUENCE {
            message-delivery-time MessageDeliveryTime,
            type-of-MTS-user TypeOfMTSUser,
            recipient-certificate[0] RecipientCertificate OPTIONAL
            proof-delivery[1] IMPLICIT ProofOfDelivery OPTIONAL}

PerRecipientNonDeliveryReportFields ::= SEQUENCE {
            non-delivery-reason-code NonDeliveryReasonCode,
            non-delivery-diagnostic-code NonDeliveryDiagnosticCode OPTIONAL}
```

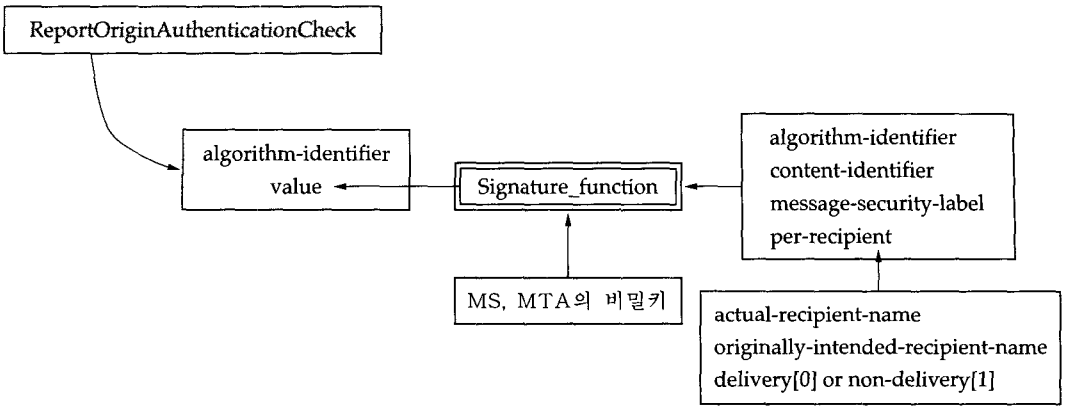


그림 4. ReportOriginAuthenticationCheck

- ProofofSubmissionRequest

ProofofSubmissionRequest는 per-message 필드이며, 이의 데이타 구조는 다음과 같다.

ProofofSubmissionRequest ::= ENUMERATED {
proof-of-submission-not-requested(0),
proof-of-submission-requested(1) }

- ProofofSubmission

이 서비스는 메세지 발신자가 Proofof-SubmissionRequest를 통해서 요구하고 이를 MTA가 받아서 그 값을 생성한 후 UA, MS에게 전달한다. 이 결과는 메세지의 발신처가 MTS에게 메세지를 제출하였다는 증거를 발신처에게 제공한다. 사용되는 알고리즘은 보안 정책에 따른다.[4] 이 값은 originating-MTA가 생성한다.

비대칭 알고리즘(asymmetric algorithm)을 사용한 경우 기본적인 인증과정을 거치며, MTA의 공개키는 orginating-MTA-certificate를 통해서 전달된다. 이 경우 제출 부인봉쇄 서비스를 위해 사용된다.

대칭 알고리즘(symmetric algorithm)을 사용하는 경우는 사용하고자 하는 키를 association이 성립될 때 bind-token을 이용하든지 아니면 별도의 키 전달 방법이 있어야 한다. 이 경우 부인봉쇄 서비스는 보안 정책에 의해서 third party가 존재할 때만 지원된다.

이 필드의 데이타 구조는 다음과 같고, 기능상의 구조를 나타낸 것이 그림 5이다.

ProofofSubmission ::= SIGNATURE SEQUENCE {
algorithm-identifier ProofOfSubmissionAlgorithmIdentifier,
message-submission-envelope MessageSubmissionEnvelope,
content Content,
message-submission-identifier MessageSubmissionIdentifier,
message-submission-time Message-submissionTime}

ProofOfSubmissionAlgorithmIdentifier ::= AlgorithmIdentifier

```
MessageSubmissionIdentifier ::= MTSIdentifier

MTSIdentifier ::= [APPLICATION 4] SEQUENCE {
        global-domain-identifier GlobalDomainIdentifier,
        local-identifier LocalIdentifier }

GlobalDomainIdentifier ::= {APPLICATION 3] SEQUENCE {
        country-name CountryName,
        administration-domain-name AdminstrationDomainName,
        private-domain-identifier PrivateDomainIdentifier OPTIONAL }

PrivateDomainIdentifier ::= CHOICE {
        numeric NumericString(SIZE(1..ub-domain-name-length)),
        printable PrintableString(SIZE(1..ub-domain-name-length)) }

LocalIdentifier ::= IA5String(SIZE(1..ub-local-id-length))

Message-submissionTime ::= Time
```

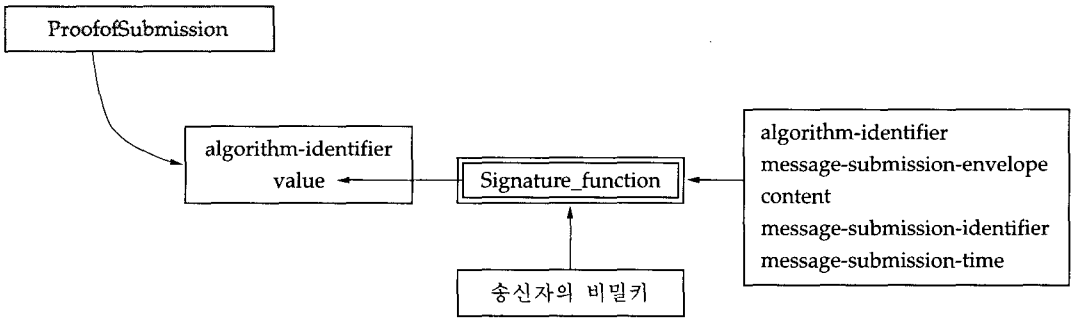


그림 5. ProofofSubmission

2.1.5 배달 증거(Proof of Delivery)

이 서비스를 이용하여, 메세지를 배달할 때 배달 사실에 대한 증명을 요구할 수 있다. 이를 위해서 메세지의 배달시에 per-recipient 필드인 ProofofDeliveryRequest를 이용하여 메세지를 수신한 UA 또는 MS에게 배달 사실 확인을 요구해야 하고 그 증거를 per-message 필드인 ProofofDelivery로 수신한다.

- ProofofDeliveryRequest

ProofofDeliveryRequest는 per-recipient 필드에 속한다. 이 서비스는 발신자가 여러 명의 수신자에게 요구할 수 있으며 그 방법은 ProofofDeliveryRequest를 통해서 요구한다. 이 요구를 수신한 UA, MS는 ProofofDelivery 값을 생성하여 delivering-

MTA에게 전달하고 MTS는 delivery report를 이용해서 그 값을 전달한다.

이 필드의 데이타 구조는 다음과 같다.

```
ProofofDeliveryRequest ::= ENUMERATED {
        proof-of-delivery-not-requested(0),
        proof-of-delivery-requested(1) }
```

- ProofofDelivery

ProofofDelivery는 per-message 필드에 속하며, 이의 대상은 평문 또는 암호문이 될 수 있지만 부인봉쇄의 목적으로 사용되는 경우는 평문을 사용해야 한다. 그 이유 중의 하나는 ProofofDelivery의 값은 MTS-user가 생성하는데 이 MTS-user가 MS인 경우 MS는 비밀성 키(confidentiality key)를 알 수 없다는 것이다.[3][8][9]

이 필드는 메세지의 발신자에게 메세지가 수신자에게 전달되었음을 증명해 준다. 사용 알고리즘은 보안 정책에 따르고, 필드의 값은 수신자가 생성한다.

이 필드의 데이타 구조는 다음과 같고, 기능상의 구조를 나타낸 것이 그림 6이다.

```
ProofofDelivery ::= SIGNATURE SEQUENCE {
    algorithm-identifier ProofOfDeliveryAlgorithmIdentifier,
    delivery-time MessageDeliveryTime,
    this-recipient-name ThisRecipientName,
    orginally-intended-recipient-name OriginallyIntendedRecipientName OPTIONAL,
    content Content,
    content-identifier ContentIdentifier OPTIONAL,
    message-security-label MessageSecurityLabel OPTIONAL }

ProofOfDeliveryAlgorithmIdentifier ::= AlgorithmIdentifier

AlgorithmIdentifier ::= SEQUENCE {
        algorithm OBJECT IDENTIFIER
        parameters ANY DEFINED BY algorithm OPTIONAL }
```

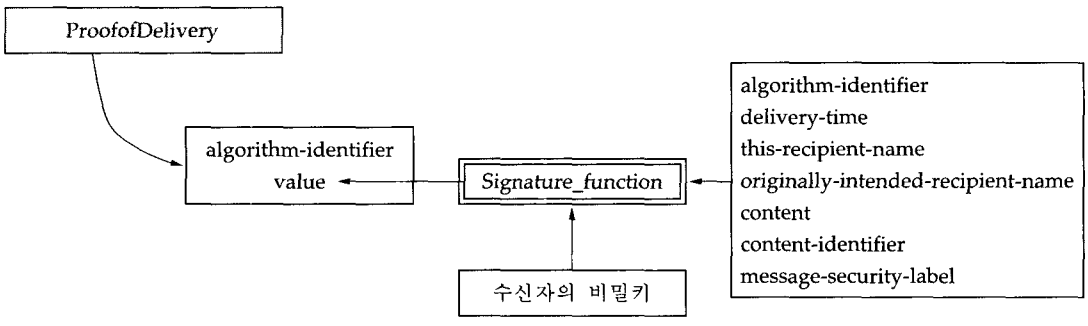


그림 6. ProofofDelivery

2.2 안전한 액세스 관리

2.2.1 상대 실체 인증(Peer Entity Authentication)

이 서비스는 MHS 구성요소 사이에서 application-association을 설정할 때 제공된다. 상호 인증(mutual authentication)은 bind 인수(argument)와 결과(result)로 전달되는 InitiatorCredentials 필드와 ResponderCredentials 필드를 통해 전달되는 양방향 인증 교환(two-way authentication exchange)을 이용하여 수행한다.[4]

인증 교환은 단순 패스워드 또는 X.509에 정의된 양방향 암호 기반 교환을 사용한다.[4]

이 서비스를 지원하기 위해서 패스워드, 비대칭 메세지 토큰, 대칭 메세지 토큰 등을 사용할 수 있다.

- InitiatorCredentials

  이는 MTS-bind의 인수이고, 이의 데이타 구조는 다음과 같다.

```
InitiatorCredentials ::= CHOICE {
        simple Passwd,
        strong [0] StrongCredentials
                (WITH COMPONENT {
                ...,
                bind-token PRESENT})}
```

- ResponderCredentials

  이는 MTS-bind의 결과이고, 이의 데이타 구조는 다음과 같다.

```
ResponderCredentials ::= CHOICE {
        simple Passwd,
        strong [0] StrongCredentials
                (WITH COMPONENT {
                ...,
                bind-token PRESENT})}
```

2.2.2 보안 문맥(Security Context)

이 서비스는 MTS-user, MTA들간에 전달될 수 있는 보안 레이블을 설정하는데 사용된다.

이 서비스는 MHS 구성 요소들간에 passing 할 수 있는 메세지의 보안 레이블을 정하기 위해 다른 구성요소와의 application-association 초기화 기능을 제공한다.

이의 데이타 구조는 다음과 같다.

```
SecurityContext ::= SET SIZE(1..ub-security-
                label) OF SecurityLabel

SecurityLabel ::= SET {
        security-policy-identifier
                SecurityPolicyIdentifier OPTIONAL,
        security-classification
                SecurityClassification   OPTIONAL,
        privacy-mark            PrivacyMark
                OPTIONAL,
        security-categories     SecurityCategories
                OPTIONAL }
```

2.3 데이타 비밀성

2.3.1 연결 비밀성(Connection Confidentiality)

MHS 표준에서 정의하지 않고 있으며, TLSP(Transport Layer Security Protocol), NLSP(Network Layer Security Protocol) 등 하위 레이어(lower layer)에서 지원받도록 권고하고 있다.[3][4][6]

2.3.2 내용 비밀성(Content Confidentiality)

이 서비스는 per-message 필드의 Content-ConfidentialityAlgorithmIdentifier를 사용하는 경우와 per-recipient 필드의 메세지 토큰 데이타들을 사용하는 두가지 방법으로 제공된다.

이 서비스는 ContentConfidentialityAlgorithmIdentifier와 메세지 비밀성 정보보호 서비스의 조합으로 제공되고, 메세지 비밀성 정보보호 서비스는 메세지 내용을 암호화하는데 사용한 키를 전송하기 위해 필요하다.

공개키 알고리즘이 사용되면 하나의 수신자에게만 사용될 수 있다.

- ContentConfidentialityAlgorithmIdentifier

  이 필드는 per-message의 필드이고, 이의 데이타 구조는 다음과 같다.

```
ContentConfidentialityAlgorithmIdentifier ::=
                                AlgorithmIdentifier

AlgorithmIdentifier ::= SEQUENCE {
        algorithm OBJECT IDENTIFIER
        parameters ANY DEFINED BY
                        algorithm OPTIONAL }
```

- 메세지 토큰

  2.1.1절의 메세지 토큰 참조

2.3.3 메세지 흐름 비밀성(Message Flow Confidentiality)

이 서비스는 메세지 흐름의 관찰을 통해서 발생할 수 있는 위협을 막기 위한 것이다. 이를 위해서 Double Enveloping 기법이 사용될 수 있다. 즉, 보내고자 하는 메세지 전체를 하나의 내용으로 내용물화하여 outer envelop를 만들고 수신했을 때 inner envelope를 추출하는 방법이다. 이 방법은 ContentType을 사용한다.

ContentType의 데이타 구조는 다음과 같다.

```
ContentType ::= CHOIE {
    built-in BuiltInContentType,
    extended ExtendedContentType }

BuiltInContentType ::= [APPLICATION 6]
                                INTEGER {
        unidentified(0),
        external(1),
        interpersonal-messaging-1984(2),
        interpersonal-messaging-1988(22),
        edi-messaging(35),
        voice-messaging(40) } (0..ub-built-in-
                                content-type)
ExtendedContentType ::= OBJECT IDENTIFIER
```

2.4 데이타 무결성 서비스

2.4.1 연결 무결성(Connection Integrity)

MHS 표준에서 정의하지 않고 있으며, TLSP, NLSP 등 하위 레이어에서 지원받도록 권고하고 있다.[3][4][6]

2.4.2 내용 무결성(Content Integrity)

이 서비스는 단대단 서비스로 메세지 내용물의 해쉬 값에 공개키 알고리즘을 사용한다.

이 서비스는 per-message의 MessageOrigin-

AuthenticationCheck 필드를 이용하는 방법과 per-recipient의 ContentIntegrityCheck 필드를 사용하는 방법이 있다. 여기서 후자의 경우는 다시 메세지 토큰을 이용하는 방법과 이용하지 않는 방법으로 분류된다. 토큰을 이용하는 방법은 다시 signed data 필드와 encrypted data 필드를 사용하는 두가지 경우로 나누어진다.

ContentIntegrityCheck 필드를 이용하면 여러명의 수신자에게 무결성 서비스를 제공할 수 있다.

또한 이 필드는 발신처 부인봉쇄, 메세지 보안 레이블과 메세지 내용물 사이의 연관성을 증명하기 위해 메세지 토큰의 signed data 또는 encrypted data에 포함된다.

이 필드는 평문을 대상으로 하고, 공개키 알고리즘을 사용한 경우 originator-certificate를 통해 전달되는 송신자의 공개키를 사용하여 검증한다.

- ContentIntegrityCheck

  이 필드는 per-recipient의 필드에 속하며, 이의 데이타 구조는 다음과 같다.

- 메세지 토큰

  2.1.1절 참조

- MessageOriginAuthenticationCheck

  2.1.1절 참조

2.4.3 메세지 순번 무결성(Message Sequence Integrity)

이 서비스는 메세지 인수 무결성 서비스와 메세지 인수 비밀성 서비를 이용하여 제공된다.

```
ContentIntegrityCheck ::= SIGNATURE SEQUENCE {
        algorithm-identifier ContentIntegrityAlgorithmIdentifier,
        content Content }

ContentIntegrityAlgorithmIdentifier ::= AlgorithmIdentifier
```

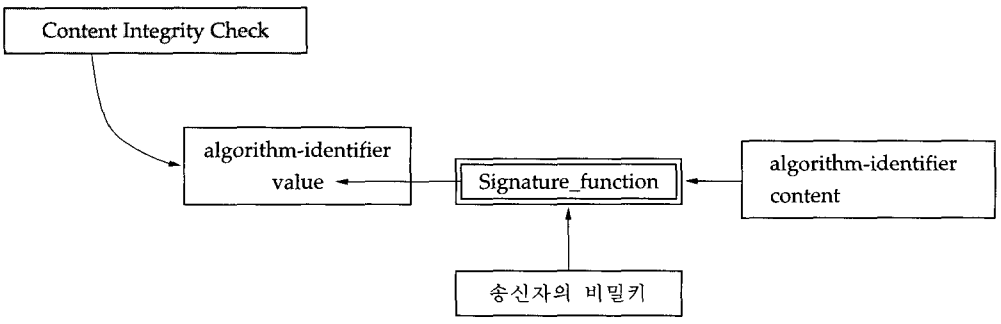


그림 7. ContentIntegrityCheck

2.5 부인 봉쇄 서비스

2.5.1 발신처 부인 봉쇄(Non-repudiation of Origin)

부인 봉쇄 서비스를 위해서 별도의 프로토콜을 정의하지는 않았다. 즉, 다른 보안 요소의 조합을 이용해서 제공한다. 부인 봉쇄 서비스는 메세지의 제출, 전달, 수신이 발생한 후에 문제가 발생했을 때 이들 행위에 대한 증거를 third party에게 제공하는 것이다. 이 경우 비대칭 알고리즘이 사용되었다면 보안 정책은 비대칭 키에 대한 관리 방법을 명확하게 제시하여야 한다.

- ContentIntegrityCheck

  이 필드는 per-recipient의 필드에 속하며, MHS 기능 모델을 나타내는 송신 UA와 수신 UA 사이에만 적용된다.

- 메세지 토큰

  이 필드 역시 송신 UA와 수신 UA 사이에만 적용된다.

- MessageOriginAuthenticationCheck

  이 필드는 per-message 필드에 속하고, 송신 UA와 수신 UA 사이의 모든 구성요소에 적용되며, 내용 비밀성 서비스가 요구되지 않았을 때 사용된다.

2.5.2 제출 부인 봉쇄(Non-repudiation of Submission)

제출 증거(2.1.4절)와 동일하다.

2.5.3 배달 부인 봉쇄(Non-repudiation of Delivery)

배달 증거(2.1.5절)와 동일하다.

2.6 메세지 보안 레이블링

메세지 보안 레이블은 originator와 recipient가 공유하는 보안 정책을 지원하기 위해서 필요하다.(multilevel security의 경우)

메세지 보안 레이블은 per-message 필드와 메세지 토큰의 signed data 또는 encrypted data 필드에 의해 전달되며, originator가 임의로 또는 강제로 주어질 수 있다.

이의 데이타 구조는 다음과 같다.

```
SecurityLabel ::= SET {
    security-policy-identifier  SecurityPolicyIdentifier OPTIONAL,
    security-classification     Security Classification  OPTIONAL,
    privacy-mark                PrivacyMark              OPTIONAL,
    security-categories         SecurityCategories       OPTIONAL }

SecurityPolicyIdentifier ::= OBJECT IDENTIFIER

SecurityClassification ::= INTEGER {
    unmarked(0),
    unclassified(1),
```

```
        restricted(2),
        confidential(3),
        secret(4),
        top-secret(5) } (0..ub-integer-optional)

    PrivacyMark ::= PrintableString(SIZE(1..ub-privacy-mark-length))

    SecurityCategories ::= SET SIZE(1..ub-security-categories) OF SecurityCategory

    SecurityCategory ::= SEQUENCE {
        type[0] IMPLICIT SECURITY-CATEGORY,
        value[1] ANY DEFINED BY type }

    SECURITY-CATEGORY MACRO ::=
    BEGIN
    TYPE NOTATION ::= type | empty
    VALUE NOTATION ::= value(VALUE OBJECT IDENTIFIER)
    END
```

2.7 정보보호 관리 서비스

2.7.1 Credential 변경(Change Credentials)

이 서비스는 MTS-user가 MTS가 가지고 있는 MTS-user의 credentials을 변경하게 하고, MTA가 MTS-user가 가지고 있는 MTA의 credentials을 변경할 수 있게 해준다.

여기서의 credentials는 MTA와 MTS-user 사이의 상호 인증 설정시에 즉, bind시에 서로 교환된다.

이의 데이타 구조는 다음과 같다.

```
ChangeCredentials ::= ABSTRACT-
                              OPERATION
    ARGUMENT SET {
    old-credentials[0] Credentials,
    new-credentials[1] Credentials -- same
                CHOICE as for old-credentials -- }

    RESULT NULL
    ERRORS {
            NewCredentialsUnacceptable,
            OldCredentialsIncorrectlySpecified }
```

2.7.2 등록(Register)

이 서비스는 MTA에 MTS-user에게 허용되는 보안 레이블을 설정하게 한다.

이의 데이타 구조는 다음과 같다.

```
Register ::= ABSTRACT-OPERATION
            ARGUMENT SET {
            user-name UserName OPTIONAL,
            user-address [0] UserAddress
                                OPTIONAL,
            deliverable-encoded-information-
                type EncodedInformationTypes
                                OPTIONAL,
            deliverable-maximum-content-
                        length [1] EXPLICT
                    ContentLength OPTIONAL,
            default-delivery-control [2]
```

```
                EXPLICT DefaultDeliveryControls
                                    OPTIONAL,
            deliverable-content-type [3] SET
                    SIZE(1..ub-content-type) OF
                        ContentType OPTIONAL,
            labels-and-redirections [4] SET
            SIZE(1..ub-labels-and-redirections) OF
              LabelAndRedirection OPTIONAL }

            RESULT NULL

            ERRORS { RegisterRejected }
```

2.7.3 MS 등록(MS-Register)

MS-register는 MS에 MS-user에게 허용되는 보안 레이블을 설정하게 한다.

이의 데이타 구조는 다음과 같다.

```
Register-MS ::= ABSTRACT-OPERATION
    ARGUMENT    Register-MSArgument
    RESULT      Register-MSResult
    ERRORS {
            AttributeError,
            AutoActionRequestError,
            InvalidRarameterError,
            SecurityError,
            ServiceError }

Register-MSArgument ::= SET {
    auto-action-registrations    [0] SET
            SIZE(1..ub-auto-registrations) OF
            AutoActionRegistration OPTIONAL,
    auto-action-deregistratons   [1] SET
            SIZE(1..ub-auto-registrations) OF
            AutoActionDeregistration
            OPTIONAL,
    list-attribute-defaults      [2] SET
            SIZE(1..ub-default-registrations) OF
            AttributeType OPTIONAL,
    fetch-attribute-defaults     [3] SET
            SIZE(1..ub-default-registrations) OF
            AttributeType OPTIONAL,
    change-credentials           [4] SEQUENCE {
            old-credentials      [0] Credentials,
            new-credentials      [1] Credentials }
                                    OPTIONAL

            -- same CHOICE as for old credentials --,
    user-security-labels         [5] SET
                SIZE(1..ub-labels-and-redirections)
                OF SecurityLabel OPTIONAL }
```

3. Pedi 정보보호 서비스 분석

EDI 시스템은 메세지 처리 시스템의 UA가 처리하는 메세지의 종류가 EDI인 시스템으로서 이에 해당하는 서비스는 크게 EDIM 책임 인증(EDIM Responsibility Authentication) 서비스와 EDIM 책임 부인 봉쇄(Non-Repudiation of EDIM Responsibility)로 나눌 수 있는데[2][7], 이들 두 서비스의 차이점은 third party의 존재 유무에 따른 분류이다. 따라서 데이타 구조 측면에서 살펴볼 때 두 서비스는 기능상으로 동일한 서비스이다.

3.1 EDI Notification 증거/부인 봉쇄 (Proof/Non-repudiation of EDI Notification)

이 서비스는 EDI-UA와 EDI-UA 사이의 문제이고, request는 EDINotificationRequests-Field의 EDINotificationSecurity 필드를 사용하고, 결과는 ContentIntegrityCheck, Message-OriginAuthenticationCheck 필드를 사용한다.

EDINotificationRequestsField의 데이타 구조

는 다음과 같다.

```
EDINotificationRequestsField ::= SEQUENCE {
    edi-notification-requests
        [0] DEINotificationRequests DEFAULT {},
    edi-notification-security
        [1] DEINotificationSecurity DEFAULT {},
    edi-reception-security
        [2] DEIReceptionSecurity DEFAULT {} }

EDINotificationRequests ::= BIT STRING {
    pn(0),
    nn(1),
    fn(2) }(SIZE(0..ub-bit-options))

EDINotificationSecurity ::= BIT STRING {
    proof(0),
    non-repudiation(1) }(SIZE(0..ub-bit-options))

EDIReceptionSecurity ::= BIT STRING {
    proof(0),
    non-repudiation(1) }(SIZE(0..ub-bit-options))
```

3.2 검색 증거/부인 봉쇄(Proof/Non-repudiation of Retrieval)

이 서비스는 UA와 MS 사이의 문제이고, EDI-MS에서 EDIMG user actions을 기록하기 위한 Secure EDI-MS audit trail을 제공한다.[1]

3.3 전달 증거/부인 봉쇄(Proof/Non-repudiation of Transfer)

이 서비스는 MTA management domain 사이의 문제이고, 메세지 세부사항과 추적 정보를 기록하기 위한 Secure MT audit trail을 제공한다.[1]

3.4 내용 증거/부인 봉쇄(Proof/Non-repudiation of Content)

이 서비스는 다음과 같은 세가지 제공 방법이 있다.

1) originating EDI-UA에 의한 발신처의 부인 봉쇄일 경우 발신처 부인 봉쇄 정보보호 서비스를 이용하는 방법
2) recipient EDI-UA에 의한 내용 부인 봉쇄일 경우 EDI notification에 content를 포함하거나, 발신처 부인 봉쇄를 이용하여 MTS에게 EDI notification을 제출하는 방법
3) trusted third party를 이용하는 방법

request는 EDINotificationRequestsField의 EDIReceptionSecurity 필드를 사용하고, 결과는 ContentIntegrityCheck, MessageOriginAuthenticationCheck를 사용한다.

4. 결 론

X.435 EDI 시스템이 제공하는 서비스들 가운데서 정보보호 서비스와 연관된 MHS 서비스와 Pedi 서비스를 우선 분석하였다. MHS 정보보호 서비스의 종류는 20가지가 있고, Pedi 정보보호 서비스는 8가지로 분류되는데, 구조 및 기능상으로는 크게 4가지로 구분될 수 있다.

또한 안전한 EDI 시스템을 구현하기 위해서 이러한 서비스에 사용되는 데이타 구조를 분석하였을 뿐만 아니라 정보보호 서비스를 제공하기 위해 사용되는 각 필드들의 기능을 파악하는데 중점을 두었다.

분석한 데이타 구조들은 안전한 EDI 시스템을 구현하는데 직접적으로 사용될 수 있을 것이다.

참 고 문 헌

[1] ITU-T F.400/X.400, Message handing services : Message handling system and Service overview, 1993.

[2] ITU-T F.435, Message handling systems : Electronic data interchange messaging service, 1991.

[3] ITU-T X.402, Message handling systems : Overall architecture, 1992.

[4] ITU-T X.411, Message handling systems - Message transfer system : Abstract service definition and procedures, 1992.

[5] ITU-T X.413, Message handling systems - Message store : Abstract service definition and procedures, 1992.

[6] ITU-T X.419, Message handling systems - Protocol specification, 1992.

[7] ITU-T X.435, Message handling systems : Electronic data interchange messaging system, 1992.

[8] Warwick Ford, *Computer Communications Security - Principles, standard protocols and techniques*, PTR Prentice Hall, 1994.

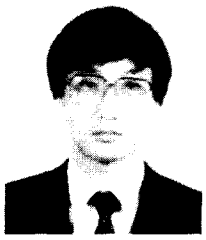
[9] Adrian Tang, Sophia Scoggins, *Open Networking with OSI*, Prentice Hall, 1992.

□ 著者紹介

이 정 현

1993년 숭실대학교 전자계산학과(학사)
1995년 숭실대학교 전자계산학과(석사)
1995년 ~ 현재 한국전자통신연구소 연구원

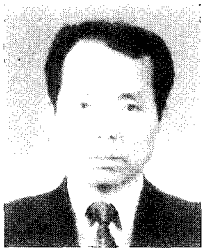
※ 주관심분야 : 컴퓨터/네트워크 보안, ATM 트래픽 제어



윤 이 중

1988년 인하대학교 전산학과(학사)
1990년 인하대학교 전산학과(석사)
1990년 ~ 현재 한국전자통신연구소 선임연구원

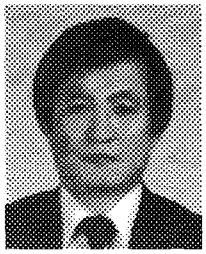
※ 주관심분야 : 컴퓨터/네트워크 보안, DBMS



김 대 호

1977년 한양대학교 전자공학과(학사)
1984년 한양대학교 산업대학원 전자공학과(석사)
1993년 Univ. of Maryland at College Park
Dept. of Computer Science Visiting Scholar
1977년 ~ 현재 한국전자통신연구소 책임연구원

※ 주관심분야 : 전송분야, 통신 및 컴퓨터 보안



이 대 기

1966년 한양대학교 전자공학과(학사)
1987년 한양대학교 전자공학과(석사)
1980년 ~ 1992년 한국전자통신연구소 산업기술개발부장, 지상시스템연구부장
1992년 ~ 현재 한국전자통신연구소 책임기술원
한국통신정보보호학회 산학이사