TTA Standard

정보통신단체표준(국문표준)

TTAS.IT-X1141_2

개정일:2006 년 12 월 27 일

SAML 2.0 바인딩

Bindings for SAML 2.0

한국정보통신기술협회
Telecommunications Technology Association

정보통신단체표준(국문표준)
TTAS.IT-X1141_2 개정일: 2006 년 12 월 27 일

SAML 2.0 바인딩

Bindings for SAML 2.0

한국정보통신기술협회
Telecommunications Technology Association

서 문

1. 표준의 목적

SAML(Security Assertion Markup Language) 2.0 은 분산된 환경에서 인증, 인가 및 속성 정보를 통신하기 위한 XML-기반 프레임워크이다. 이름에서 나타나듯이, SAML 은 비즈니스 엔티티들이 어떤 주체의 신원, 속성 그리고 권한부여에 대한 주장을 파트너 회사 또는 다른 엔터프라이즈 응용 등과 같은 다른 엔티티들에게 주장하는 것을 허용한다. 이 표준은 통신 프로토콜과 프레임워크에서 SAML 주장과 요청-응답 메시지들의 사용에 대한 프로토콜 바인딩을 정의한다.

이 표준은 ITU-T X.1141 “Security Assertion Markup Language (SAML 2.0)”을 근거로 한 국내 표준으로 원문의 다음 내용을 포함하고 있다.

CL 1. 범위

CL 2. 참고문헌

CL 3. 용어정의

CL 4. 약어

CL 5. 관례

CL 7. 공통 데이터 타입

CL 10. SAML 바인딩

2. 주요 내용 요약

이 표준은 추가적인 프로토콜 바인딩을 명기하는 것에 대한 가이드라인을 기술한다. 이 표준은 SAML SOAP 바인딩, Reverse SOAP Binding, HTTP Redirect Binding, HTTP Post Binding, HTTP Artifact Binding, SAML URI Binding 을 정의한다.

3. 표준 적용 산업 분야 및 산업에 미치는 영향

본 표준은 웹 싱글사인온, 속성-정보 기반 인가와 웹 서비스 보호에서 사용될 수 있다. 따라서, 본 표준은 ID 관리 분야와 웹 서비스 정보보호 분야에 직접적으로 적용되며, 정보보호 산업의 핵심 요소로 활용될 수 있다. 또한, ID 연계의 핵심 기술을 제공함으로써, 기업간 협업을 용이하게 함으로써 새로운 서비스를 창출하고 시장을 활성화할 수 있다.

4. 참조 표준(권고)

4.1. 국외 표준(권고)

- ITU-T X.1141, 'Security Assertion Markup Language (SAML 2.0)', 2006.06.

4.2. 국내 표준

- TTAS.OT-10.0041, 'SAML 바인딩과 프로파일', 2005.12.

5. 참조 표준(권고)과의 비교

5.1. 참조 표준(권고)과의 관련성

본 ITU-T X.1141, 'Security Assertion Markup Language (SAML 2.0)',을 근거로 한 국내표준임. TTAS.OT-10.0041 는 OASIS 의 SAMLv1.0 을 기준으로 개발된 표준이였으나 현재 SAMLv2.0 표준이 ITU-T 표준으로 개발되어 이를 개정함.

5.2. 참조한 표준(권고)과 본 표준의 비교표

상기 국제 권고에 대한 추가사항은 없으며 장 구성은 다음과 같음.

ITU-T X.1141	TTAS.IT-X1141_2	비고
1. 범위	1.1. 범위	동일(번역)
2. 참고문헌	1.2. 참고문헌	동일(번역)
3. 용어정의	1.3. 용어정의	동일(번역)
4. 약어	1.4. 약어	동일(번역)
5. 관례	1.5. 관례	동일(번역)
7. 공통 데이터 타입	1.6. 공통 데이터 타입	동일(번역)
10. SAML 바인딩	2~3. SAML 바인딩	동일(번역)

6. 지식 재산권 관련 사항

본 표준의 '지식 재산권 확약서' 제출 현황은 TTA 웹사이트에서 확인할 수 있다.

※본 표준을 이용하는 자는 이용함에 있어 지식 재산권이 포함되어 있을 수 있으므로, 확인 후 이용한다.

※본 표준과 관련하여 접수된 확약서 이외에도 지식 재산권이 존재할 수 있다.

7. 시험 인증 관련 사항

7.1. 시험 인증 대상 여부

- 해당 사항 없음.

7.2. 시험 표준 제정 현황

- 해당 사항 없음.

8. 표준의 이력 정보

8.1. 표준의 이력

판수	제정·개정일	제정·개정 내역
제 1 판	2005. 12. 21.	제정 TTAS.IT-X1141
제 2 판	2006. 12. 27.	개정 TTAS.IT-X1141_2

8.2. 주요 개정 사항

- 해당 사항 없음.

Preface

1. Purpose of Standard

SAML(Security Assertion Markup Language) is an XML-based framework for communicating user authentication, entitlement, and attribute information among disparate Web access management and security products. As its name suggests, SAML allows business entities to make assertions regarding the identity, attributes, and entitlements of a subject (an entity that is often a human user) to other entities, such as a partner company or another enterprise application. This standard defines protocol bindings for the use of SAML assertions and request-response messages in communications protocols and frameworks.

This standard is a domestic standard based on ITU-T X.1141 “Security Assertion Markup Language (SAML 2.0)” and contains the following contents of the original standard.

CL 1. Scope

CL 2. References

CL 3. Definitions

CL 4. Abbreviations

CL 5. Conventions

CL 7. Common data types

CL 10. Bindings for SAML

2. Summary of Contents

The standard specifies guidelines for specifying additional protocol bindings. It defines SAML SOAP Binding, Reverse SOAP Binding, HTTP Redirect Binding, HTTP Post Binding, HTTP Artifact Binding and SAML URI Binding.

3. Applicable Fields of Industry and its Effect

This standard can be used in as web single-sign on, attribute information based authorization and web service security. Therefore, it is directly applicable to security areas such as ID Management and web service security. It is also applicable to other information security industry as essential component. In addition, it provides essential technology for ID federation, which makes companies' collaboration easy and so creates new service and revitalize IT market.

4. Reference Standards (Recommendations)

4.1. International Standards (Recommendations)

- ITU-T X.1141, "Security Assertion Markup Language (SAML 2.0)," 2006.06.

4.2. Domestic Standards

- TTAS.OT-10.0041, "SAML Binding and Profile", 2005.12.

5. Relationship to Reference Standards(Recommendations)

5.1. Relationship of Reference Standards(Recommendations)

This standard is a domestic standard based on ITU-T X.1141, “Security Assertion Markup Language (SAML 2.0)”. This standard updated the TTAS.OT-10.0041 based on OASIS standard as SAMLv1.0.

5.2. Differences between Reference Standard(Recommendation) and this Standard

This standard has no additional contents as to the international recommendations. The differences between the recommendation and this standard are as follows.

ITU-T X.1141	TTAS.IT-X1141_2	Outline
1. Scope	1.1. Scope	equaled(trans)
2. References	1.2. References	equaled(trans)
3. Definitions	1.3. Definitions	equaled(trans)
4. Abbreviations	1.4. Abbreviations	equaled(trans)
5. Conventions	1.5. Conventions	equaled(trans)
7. Common data types	1.6. Common data types	equaled(trans)
10. Bindings for SAML	2~3. Bindings for SAML	equaled(trans)

6. Statement of Intellectual Property Rights

IPRs related to the present document may have been declared to TTA. The information pertaining to these IPRs, if any, is available on the TTA Website.

No guarantee can be given as to the existence of other IPRs not referenced on the TTA website.

And, please make sure to check before applying the standard.

7. Statement of Testing and Certification

7.1. Object of Testing and Certification

- None

7.2. Standards of Testing and Certification

- None

8. History of Standard

8.1. Change History

Edition	Issued date	Outline
The 1st edition	2005. 12. 21.	Established TTAS.IT-X1141
The 2nd edition	2006. 12. 27.	Revised TTAS.IT-X1141_2

8.2. Revisions

- None

목 차

Contents

SAML 2.0 바인딩
(Bindinds for SAML 2.0)

1. 표준의 구성 및 범위

SAML 2.0 은 시스템 엔티티가 어떤 주체에 대하여 생성한 주장의 문법과 처리 규칙을 정의한다. 이와 같은 주장을 만들거나 또는 의지하기 위해, SAML 시스템 엔티티들은 주장 자체 또는 주장의 주체에 대한 내용을 통신하기 위해 다른 프로토콜을 사용할 수 있다. SAML 2.0 은 SAML 보장의 구조, 관련된 프로토콜 집합, 그리고 SAML 시스템을 관리하는데 관련된 처리 규칙들을 정의한다.

SAML 주장과 프로토콜 메시지들은 XML 로 인코딩되어 있으며, XML 네임스페이스를 사용한다. 이것들은 일반적으로 HTTP POST 또는 XML 로 인코딩된 SOAP 메시지와 같은 전송을 위한 다른 구조에 내장된다. SAML 2.0 은 또한 SAML 프로토콜 메시지들을 내장하고 전송하기 위한 프레임워크를 제공하는 SAML 바인딩을 명기한다. 더욱이, SAML 2.0 은 SAML 특징들을 사용할 때, 특정 사용예(use case)를 달성하고 상호운용성을 달성하기 위해, SAML 주장과 프로토콜을 어떻게 사용해야 하는지에 대한 기본 프로파일 집합을 제공한다.

SAML 2.0 은 다음을 정의한다.

1. SAML 에 대한 적합성 요구사항;
2. SAML 주장과 프로토콜:
- SAML 주장 스키마,
- SAML 프로토콜 스키마;
3. SAML 바인딩;
4. SAML 프로파일:
- SAML ECP 프로파일 스키마,
- SAML X.500/LDAP 속성 프로파일 스키마,
- SAML DCE PAC 속성 프로파일 스키마,

- SAML XACML 속성 프로파일 스키마;

5. SAML 메타데이터;

6. SAML 메타데이터 스키마;

7. SAML 인증 문맥.

2. 참고 표준(권고)

다음 권고안들과 다른 참조들은 SAML 2.0 에서 참조되는 것들이다. SAML 2.0 의 발간시에는 모두 유효한 상태이다. 모든 권고안들과 다른 참조들은 개정될 수 있으며, SAML 2.0 에 기반으로 하는 모든 사용자들은 아래 나열된 권고안들과 다른 참조들에 대하여 가장 최신 판을 적용할 수 있다. ITU 의 전기통신 표준국(Telecommunications Standardization Bureau)에서 현재 유효한 ITU-T 권고안들의 리스트를 유지한다. ITEF 는 최근에 폐지된 것들과 함께 RFC 리스트를 유지한다. W3C, Unicode Consortium 과 Liberty Alliance 도 가장 최신의 권고안들과 다른 문서들에 대한 리스트를 유지한다.

- ITU-T Recommendation X.660, Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: General procedure, 2004.
- ITU-T Recommendation X.667, Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: Generation and Registration of Universally Unique Identifiers (UUIDs) and their Use as ASN.1 Object Identifier Components, 2004.
- ITU-T Recommendation X.680, Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation, 2002.
- ITU-T Recommendation X.800, Information processing systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture, 1991.
- ITU-T Recommendation X.811, Security Frameworks for Open Systems: Authentication Framework, 1995.

- ITU-T Recommendation X.812, Security Frameworks for Open Systems: Access control framework, 1995.
- ITU-T Recommendation X.1142, Extensible Access Control Markup Language (XACML 2.0), 2006.
- IETF RFC 1034:1987, Domain Names - Concepts and Facilities, 1987.
- IETF RFC 1510:1993, The Kerberos Network Authentication Requestor (V5), 1993.
- IETF RFC 1750:1994, Randomness Recommendations for Security, 1994.
- IETF RFC 1951:1996, DEFLATE Compressed Data Format Specification Version 1.3, 1996.
- IETF RFC 1991:1996, PGP Message Exchange Formats, 1996.
- IETF RFC 2045:1996, Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message, 1996.
- IETF RFC 2119:1997, Key words for use in RFCs to Indicate Requirement Levels, 1997.
- IETF RFC 2246:1999, The TLS Protocol Version 1.0, 1999.
- IETF RFC 2253:1997, Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished, 1997.
- IETF RFC 2396:1998, Uniform Resource Identifiers (URI): Generic Syntax, 1998.
- IETF RFC 2535:1999, Domain Name System Security Extensions, 1999.
- IETF RFC 2616 :1999, Hypertext Transfer Protocol - HTTP/1.1, 1999.
- IETF RFC 2617:1999, HTTP Authentication: Basic and Digest Access Authentication, 1999.
- IETF RFC 2798:2000, Definition of the inetOrgPerson LDAP Object Class, 2000.
- IETF RFC 2828:2000, Internet Security Glossary, 2000.
- IETF RFC 2914:2000 , Congestion Control Principles, 2000.
- IETF RFC 2915:2000, The Naming Authority Pointer (NAPTR) DNS Resource Record, 2000.

- IETF RFC 2945:2000, The SRP Authentication and Key Exchange System, 2000.
- IETF RFC 2965:2000, HTTP State Management Mechanism, 2000.
- IETF RFC 3061:2001, A URN Namespace of Object Identifiers, 2001.
- IETF RFC 3075:2001, XML-Signature Syntax and Processing, 2001.
- IETF RFC 3513:2003, Internet Protocol Version 6 (IPv6) Addressing Architecture, 2003.
- IETF RFC 3023:2001, XML Media Types, 2001.
- IETF RFC 3377:2002, Lightweight Directory Access Protocol (v3): Technical Specification, 2002.
- IETF RFC 3403:2002, Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database, 2002.
- IETF RFC 3546:2003, Transport Layer Security (TLS) Extensions, 2003.
- IETF RFC 3923:2004, End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP), 2004.
- IETF RFC 4122:2005, A Universally Unique IDentifier (UUID) URN Namespace, 2005.
- Liberty Alliance POAS:2003, R. Aarts, Reverse HTTP Binding for SOAP Specification Version 1.0, Liberty Alliance Project, 2003.
- OASIS WSS:2006, WS-Security Core Specification 1.1, February, 2006.
- UNICODE-C, M. Davis, M. J. Dürst, Dürst. Unicode Normalization Forms. UNICODE Consortium, March 2001.
- W3C Canonicalization:2002, Exclusive XML Canonicalization Version 1.0, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/xml-exc-c14n/, 2002.07.02.
- W3C Character Model:2005, Character Model for the World Wide Web 1.0: Fundamentals, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en

Informatique et en Automatique, Keio University),
http://www.w3.org/TR/2005/REC-charmod-20050215/. 2005.02.15.

- W3C Datatypes:2001, XML Schema Part 2: Data types, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/. 2001.05.02
- W3C Encryption:2002, XML Encryption Syntax and Processing, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/, 2002.12.10.
- W3C Web Services Glossary:2004, Web Services Glossary, W3C Note, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/ws-gloss/, 2004.02.11.
- W3C HTML:1999, HTML 4.01 Specification, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/REC-html40/, 1999.12.24.
- W3C Namespaces:1999, Namespaces in XML, W3C Recommendation, Copyright © World Wide Web Consortium (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/REC-xml-names/, 1999.1.14.

- W3C Primer:2005, SOAP Version 1.2 Part 0: Primer, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology,

Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2003/REC-soap12-part0-20030624/, 2005.6.24.

- W3C Signature:2002, XML Signature Syntax and Processing, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/xmldsigcore/, 2002.2.12.
- W3C Signature Schema:2001, XML Signature Schema, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd, 2001.3.1
- W3C String:1998, Requirements for String Identity Matching and String Indexing, W3C Note, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/WD-charreq. 1998.7.10.
- W3C SOAP:2000, Simple Object Access Protocol (SOAP) 1.1, W3C Note, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2000/NOTE-SOAP-20000508, 2000.5.8.
- W3C XHTML:2002, The Extensible HyperText Markup Language (Second Edition), W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/xhtml1/, 2002.8.1.
- W3C XML 1.0:2004, Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/REC-xml/, 2004.2.4.
- W3C XML Schema Part 1:2001, XML Schema Part 1: Structures, W3C Recommendation, Copyright © World Wide Web Consortium, (Massachusetts

Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/, 2001.5.2.

주의 – SAML 2.0 문서 내에 있는 문서에 대한 참조는 참조되는 문서의 상태를 제공하지는 않는다.

3. 용어 정의 및 약어

3.1. 용어 정의

SAML 2.0 에 대해, 다음과 같은 용어 정의가 적용된다.

3.1.1. 들여온 정의들(Imported definitions)

- SAML 2.0 은 ITU-T Rec. X.667 에서 정의된 다음 용어들을 사용한다:
  a) UUID
- SAML 2.0 은 ITU-T Rec. X.680 에서 정의된 다음 용어들을 사용한다:
  a) 객체 식별자(Object identifier);
  b) 오픈 타입 표기법(Open type notation).
- SAML 2.0 은 ITU-T Rec. X.811 에서 정의된 다음 용어들을 사용한다:
사용자(Principle).
- SAML 2.0 은 ITU-T Rec. X.812 에서 정의된 다음 용어들을 사용한다:
  a) 접근 제어 정보(Access control information);
  b) 사용자(User).
- SAML 2.0 은 W3C 웹 서비스 어휘에서 정의된 다음 용어들을 사용한다:
  a) 초기 SOAP 송신자(Initial SOAP sender);
  b) 네임스페이스(Namespace);
  c) 최종 SOAP 수신자(Ultimate SOAP receiver);
  d) XML 스키마(XML schema).

- SAML 2.0 은 IETF RFC 2828 에서 정의된 다음 용어들을 사용한다:

a) 접근(Access);

b) 접근 제어(Access control);

c) 프락시(Proxy);

d) 프락시 서버(Proxy server);

f) 풀(Pull);

e) 푸시(Push);

g) 보안 아키텍처(Security architecture);

h) 보안 정책(Security policy);

i) 보안 서비스(Security service).

- SAML 2.0 은 IETF RFC 2396 에서 정의된 다음 용어들을 사용한다:

a) Uniform resource identifier (URI);

b) URI 참조(URI reference).

3.1.2. 추가적인 용어 정의(Additional definitions)

3.1.2.1. 접근 권한(Access rights)

주체가 자원에 대하여 가질수 있는 인가된 상호작용의 타입을 설명. 예로는 읽기, 쓰기, 실행, 추가, 변경 그리고 삭제를 들 수 있다.

3.1.2.2. 계정(Account)

사용자와 비즈니스 서비스 제공자 사이에 정상적인 거래와 서비스를 제공하기 위한 형식적인 비즈니스 협약.

3.1.2.3. 계정 연결(Account linkage)

서로 다른 두 제공자에서 동일한 사용자를 나타내는 계정을 연관시키는 방법. 이를 통해 두 제공자들은 그 사용자에 대한 정보를 통신할 수 있다. 계정 연결은 속성 공유나 또는 Identity 연계(federation)을 통해 설정될 수 있다.

3.1.2.4. 능동적인 역할(Active role)

예를 들어 자원에 접근하는 등, 어떤 연산을 수행할 때, 시스템 엔티티가 가지는 역할.

3.1.2.5. 관리 도메인(Administrative domain)

하나 또는 그 이상의 관리 정책, 인터넷 도메인 이름 등록들, 공공 법률 엔티티들(예를 들어, 개인, 기업 또는 다른 조직), 호스트, 네트워크 디바이스 그리고 상호 연결되는 네트워크의 집합, 그리고 그들 위에서 동작하는 네트워크 서비스와 응용들의 어떠한 조합으로 정의되는 환경 또는 문맥. 관리 도메인은 하나 또는 그 이상의 보안 도메인을 포함하거나 또는 정의할 수 있다. 하나의 관리 도메인은 단일한 사이트 또는 다중 사이트를 포함할 수 있다. 관리 도메인을 정의하는 특징들은 시간이 지남에 따라 진화할 수 있다. 관리 도메인들은 관리 도메인 경계를 넘어 서비스를 제공하거나 또는 소비하는 것에 대하여 협약을 만들 수 있다.

3.1.2.6. 관리자(Administrator)

시스템을 설치하거나 또는 관리하는 사람 또는 시스템을 이용하여 시스템 엔티티, 사용자와/또는 내용을 관리하는 사람. 관리자는 일반적으로 특정 관리 도메인에 가입하게 되고 하나 이상의 관리 도메인에 가입할 수도 있다.

3.1.2.7. 가맹, 가맹 그룹(Affiliation, affiliation Group)

사용자(principal)에 대한 식별자들의 (연계 관점에서) 단일한 네임스페이스를 공유하는 시스템 엔티티 집합.

3.1.2.8. 익명성(Anonymity)

익명 상태. 이것은 이름이나 신원이 알려지거나 노출되지 않도록 하는 조건을 나타냄.

3.1.2.9. 보장하는 기관(Asserting party)

공식적으로, 하나 또는 그 이상의 SAML 기관을 호스트팅하는 관리 도메인. 비공식적으로, SAML 기관의 한 인스턴스.

3.1.2.10. 주장(Assertion)

주체에 대하여 수행되는 인증 행위, 주체에 대한 속성 정보 또는 명기된 자원에 대하여 주체가 행할 수 있는 인가 데이터 등에 대하여 SAML 기관이 생성한 데이터 조각.

3.1.2.11. 속성(Attribute)

객체의 독특한 특성. 실세계 객체에 대하여, 속성들은 종종 크기, 모양, 무게 및 색깔 등과 같은 물리적인 특징들로 명기된다. 사이버스페이스에서 객체는 크기, 인코딩 타입, 네트워크 주소 등등을 설명하는 속성들을 가질 수 있다. 속성들은 종종 “속성 이름”과 “속성 값(들)로 표현된다. 예를 들어, “foo”는 값 ‘bar’를 가지며, “count”는 값 1 을, “gizmo”는 ‘frob”과 ‘2”를 값들로 가진다.

3.1.2.12. 속성 주장(Attribute assertion)

주체의 속성들에 대한 정보를 운반하는 주장.

3.1.2.13. 속성기관(Attribute authority)

속성 주장들을 생성하는 시스템 엔티티.

3.1.2.14. 인증(Authentication)

인증은 어떤 사람 또는 어떤 사물이 어느 정도의 신뢰 내에서 그것이 자신이 그렇다고 선언하는 것이 정말로 맞는지 아닌지를 결정하는 과정이다.

3.1.2.15. 인증 주장(Authentication assertion)

주체에 대하여 발생된 성공적인 인증 행위에 대한 정보를 운반하는 주장.

3.1.2.16. 인증 기관(Authentication authority)

인증 주장들을 생성하는 시스템 엔티티.

3.1.2.17. 인가(Authorization)

어떤 주체가 특정 자원에 대하여 명기된 타입의 접근을 수행하는 것이 허가되었는지를, 적용가능한 접근제어 정보를 평가함으로써, 결정하는 과정. 일반적으로, 인가는 인증 문맥 내에 있다. 일단 주체가 인증이 되면, 그것은 다른 타입들의 접근을 수행하는 것에 대하여 인가될 수 있다.

3.1.2.18. 인가 결정(Authorization decision)

인가 행위의 결과. 그 결과는 부정적인 될 수 있다. 즉, 그것은 주체가 자원에 대한 어떠한 접근 권한도 없음을 가리킨다.

3.1.2.19. 인가 결정 주장(Authorization decision assertion)

인가 결정에 대한 정보를 운반하는 주장.

3.1.2.20. 후 채널(Back channel)

후 채널은, 예를 들어 사용자 에이전트인 HTTP 클라이언트와 같은 또 다른 시스템 엔티티를 통하여 메시지를 리다이렉트(redirect) 하지 않고 두 시스템 엔티티들 사이에 직접적인 통신을 가리킨다.

3.1.2.21. 바인딩, 프로토콜 바인딩(Binding, protocol binding)

일반적으로, 어떤 프로토콜 메시지와 메시지 교환 패턴을 구체적인 방식으로 또 다른 프로토콜로 매핑시키는 것에 대한 명세임. 예를 들어, SAML <AuthnRequest> 메시지를 HTTP 에 매핑하는 것은 바인딩의 한 예가 된다. 동일한 SAML 메시지를 SOAP 으로 매핑하는 것은 또 다른 바인딩이 된다. SAML 문맥에서는, 각각의 바인딩에 “SAML xxx binding”이라는 패턴의 이름이 주어진다.

3.1.2.22. 크리덴셜(Credentials)

주장되는 사용자(principal) 신원을 확인하기 위해 전송되는 데이터.

3.1.2.23. 최종 사용자(End user)

응용 목적으로 자원을 사용하는 자연인(natural person).

3.1.2.24. 엔티티(Entity)

“시스템 엔티티”를 참고한다.

3.1.2.25. 연계하다(Federate)

둘 또는 그 이상의 엔티티들을 함께 연결하거나 또는 바인딩하기.

3.1.2.26. 연계(Federation)

이 용어는 두가지 의미로 사용된다.:

1. 두 엔티티 사이에 관계를 설정하는 행위.
2. 어떠한 개수의 서비스 제공자들과 아이덴터티 제공자들로 구성된 하나의 연합(association).

3.1.2.27. 연계된 아이덴터티(Federated identity)

제공자들 사이에 그 사용자를 참조하기 위해 사용되는 식별자 집합과 속성들에 대하여 협정(agreement)가 있을 때, 사용자(principal)의 아이덴터티는 연계가 되었다고 말해진다.

3.1.2.28. 전 채널(Front channel)

전 채널은 두 개의 HTTP 로 통신하는 서버들이 “HTTP redirect” 메시지를 채용하고 이를 통해, 예를 들어 웹 브라우저 또는 다른 어떠한 HTTP 클라이언트인 사용자 에이전트를 경유하여 상호간에 메시지를 전달하는 경우에 효과가 발생되는 통신 채널을 가리킨다.

3.1.2.29. 식별자(Identifier)

시스템 엔티티들 유일하게 가리키도록 시스템 엔티티에 매핑된 데이터 객체. 예를 들어 문자열이 될 수 있음. 시스템 엔티티는 그것을 가리키는 다중 식별자를 가질 수 있다. 하나의 식별자는 본질적으로 엔티티의 “구별되는 속성”이다.

3.1.2.30. 아이덴터티, 신원(Identity)

엔티티의 본질. 어떤 사물의 아이덴터티는 어떤 사물의 특징들로 종종 설명된다. 이 특성들 중에 식별자들이 포함될 수 있다.

3.1.2.31. 아이덴터티 탈연계(Identity defederation)

제공자들이 일정 집합의 식별자와/또는 속성들을 통해 사용자(principal)을 참조하는 것을 그만두기로 동의할 때, 발생하는 동작.

3.1.2.32. 아이덴터티 연계(Identity federation)

사용자(principal)을 위해 연계된 아이덴터티를 생성하는 동작.

3.1.2.33. 아이덴터티 제공자(Identity provider)

사용자(principal)들을 위해 아이덴터티 정보를 생성하고, 유지하며, 관리하고 그리고 웹 브라우저 프로파일과 같이 하나의 연계 내에서 다른 서비스 제공자에게 사용자(principal) 인증을 제공하는 일종의 서비스 제공자.

3.1.2.34. 아이덴터티 제공자 라이트(Identity provider lite)

단지 SAML 에서 요구되는 부분만을 사용하여, 사용자(principal)들을 위해 아이덴터티 정보를 생성하고, 유지하며, 관리하고 그리고 웹 브라우저 프로파일과 같이 하나의 연계 내에서 다른 서비스 제공자에게 사용자(principal) 인증을 제공하는 일종의 서비스 제공자.

3.1.2.35. 로그인, 로그온, 사인-온(Login, logon, sign-on)

일종의 처리. 이 처리를 통해 사용자가 인증기관에게 크리덴셜을 제출하고 간단한 세션을 설정하고 그리고 선택적으로 리치(rich) 세션을 설정한다.

3.1.2.36. 로그아웃, 로그오프, 사인-오프(Logout, logoff, sign-off)

일종의 처리. 이 처리를 통해 사용자는 단순 세션 또는 리치(rich) 세션을 종료하기를 원한다는 것을 알린다.

3.1.2..37. 마크업 언어(Markup language)

특수한 목적으로 XML 문서의 구조에 적용되는 일단의 XML 요소들과 XML 속성들. 마크업 언어는 일반적으로 일단의 XML 스키마들과 동반되는 문서로 정의된다.

3.1.2.38. 이름 제한자(Name qualifier)

다른 사용자들(principals)을 나타내기 위해, (연계 관점에서) 하나 이상의 네임스페이스에서 사용될 수 있는 하나의 식별자가 모호해지지 않도록 해 주는 문자열.

3.1.2.39. 기관, 당사자(Party)

비공식적으로, 주장을 수신하거나 또는 자원을 접근하는 것과 같은 어떤 처리나 통신에 참여하는 하나 또는 그 이상의 사용자들(principals).

3.1.2.40. 영속적인 의사익명(Persistent pseudonym)

다중 세션에 걸쳐있는 확장된 기간 동안에 주어진 의지하는 기관이 사용자를 식별할 수 있도록, 어떤 아이덴터티 제공자에 의해 할당된 프라이버시-보호형 이름 식별자. 아이덴터티 연계를 나타내는데 사용될 수 있다.

3.1.2.41. 정책 결정점(Policy decision point (PDP))

자신을 위해 인가 결정을 내리거나 또는 이와 같은 결정을 요구하는 다른 시스템 엔티티를 위해 인가 결정을 내리는 시스템 엔티티. 예를 들어, SAML PDP 는 인가 결정 요청들을 받아들여, 응답으로 인가 결정 주장들을 생성한다. PDP 는 인가 결정 기관이다.

3.1.2.42. 정책 집행점(Policy enforcement point (PEP))

인가 결정을 요청하고 뒤이어 집행하는 시스템 엔티티. 예를 들어, SAML PEP 는 일가 결정 요청들을 PDP 에게 전달하고, 응답으로 수신되는 인가 결정 주장들을 처리한다.

3.1.2.43. 사용자 아이덴터티(Principal identity)

일반적으로 식별자인 어떤 사용자 아이덴터티의 표현.

3.1.2.44. 프로파일(Profile)

여러 목적 중에 하나를 위한 일단의 규칙들. 각각의 집합은 “SAML xxx 프로파일” 또는 “xxx SAML 프로파일” 패턴으로 이름이 주어진다.

1. 어떤 프로토콜 또는 다른 사용 문맥에 주장을 내장시키거나 또는 그것들로부터 추출하는 방법에 대한 규칙들.
2. 특수한 사용 문맥에서 SAML 프로토콜 메시지를 사용하는 것에 대한 규칙들.
3. SAML 로 표현된 속성들을 또 다른 속성 표현 시스템으로 매핑시키는 것에 대한 규칙들. 이와 같은 규칙의 집합은 “속성 프로파일”로 알려진다.

3.1.2.45. 프로토콜 바인딩(Protocol binding)

“바인딩”을 참고한다.

3.1.2.46. 제공자(Provider)

아이덴터티 제공자들과 서비스 제공자들 둘 다를 가리키는 포괄적인 표현.

3.1.2.47. 의지하는 기관(측)(Relying party)

다른 시스템 엔티티가 제공한 정보를 기반으로 행동을 취할 것을 결정하는 시스템 엔티티. 예를 들어, SAML 의지하는 기관은 주체에 대하여 보장하는 기관(SAML 기관)이 제공한 주장들을 의지한다.

3.1.2.48. 요청자(Requester)

또 다른 시스템 엔티티(SAML 기관, 응답자)에게 서비스를 요청하기 위해 SAML 프로토콜을 활용하는 시스템 엔티티. 많은 시스템 엔티티들이 클라이언트와 서버 둘 모두로서 동시에 또는 순차적으로 동작하기 때문에, 이 표시법에서 “클라이언트”라는 용어는 사용되지 안 된다. SAML SOAP 바인딩이 사용중인 경우에는, SAML 요청자는 초기 SOAP 송신자와 구조적으로 분리된다.

3.1.2.49. 자원(Resource)

(예를 들어, 파일 형태나 메모리 형태, 등등으로) 하나의 정보 시스템에 포함되는 데이터, 또한:

1. 시스템이 제공하는 서비스.
2. 시스템 장비의 한 항목(다른 말로, 하드웨어, 펌웨어, 소프트웨어 또는 문서등과 같은 시스템 컴포넌트)

3.1.2.50. 응답자(Responder)

또 다른 시스템 엔티티(요청자)로부터 전달받은 서비스 요청에 대하여 응답하기 위해 SAML 프로토콜을 활용하는 시스템 엔티티(SAML 기관). 많은 시스템 엔티티들이 클라이언트와 서버 둘 모두로서 동시에 또는 순차적으로 동작하기 때문에, 이 표시법에서 “서버”라는 용어는 사용되지 안 된다. SAML SOAP 바인딩이 사용중인 경우에는, SAML 응답자는 최종 SOAP 수신자와 구조적으로 분리된다.

3.1.2.51. 역할, 롤(Role)

사전들은 역할을 “수행자에 의해 동작되는 특성” 또는 “함수 또는 위치)로 정의한다. 시스템 엔티티들은 예를 들어 능동적인 역할들과 수동적인 역할들과 같은 다양한 타입들의 역할들을 순차적으로/또는 동시적으로 수행한다. 관리자의 개념은 종종 역할의 한 예이다.

3.1.2.52. SAML 아티팩트(SAML artifact)

일반적으로 더 크고, 가변-크기의 SAML 프로토콜 메시지를 가리키는 작고, 고정-크기를 가지는 구조화된 데이터 객체. SAML 아티팩트들은 “3xx Redirection” 상태 코드들을 가지는 HTTP 응답 메시지들과 뒤따르는 HTTP GET 메시지들과 같이 URL 에 내장되고 HTTP 메시지들을 통해 운반되도록 설계된다. 이런 방식으로, 서비스 제공자는 간접적으로, 사용자 에이전트를 경유하여, 다른 제공자에게 SAML 아티팩트를 전달할 수 있다. 다른 제공자는 artifact 를 제공하는 제공자와의 직접적은 상호작용을 통해 SAML 아티팩트를 디레퍼런스(dereference)하여 SAML 프로토콜 메시지를 얻을 수 있다.

3.1.2.53. SAML 기관(SAML authority)

SAML 도메인 모델에서 주장들을 발급하는 추상적인 시스템 엔티티. 속성 기관, 인증 기관, 정책 결정점(PDP)를 또한 참고한다.

3.1.2.54. 보안(Security)

정보의 기밀성을 보장하고, 그것을 처리하는데 사용되는 시스템과 네트워크를 보호하고, 그들에 대한 접근을 제어하는 일단의 보호방법들. 보안은 일반적으로 비밀(secrecy), 기밀성, 무결성, 이용가능성 등의 개념을 포괄한다. 이것은

어떤 시스템이 잠재적으로 상호연관된 공격들을 방어하는 것을 보장하기 위한 것이다.

3.1.2.55. 보안 주장(Security assertion)

보안 아키텍처의 문맥에서 철저히 검사된 주장.

3.1.2.56. 보안 문맥(Security context)

개별적인 SAML 프로토콜 메시지에 대하여, 메시지의 보안 문맥은 만약 있다면 메시지의 보안 헤더 블록들과 수신자에게 메시지를 배달할 때, 사용될 수 있는 다른 보안 메커니즘들의 의미적인 합(semantic union)이다. HTTP, TLS 와 IPSEC 등과 같은 하부 네트워크 스택 레이어들에서 채택되는 보안 메커니즘들이 후자의 예가 된다.

3.1.2.57. 보안 도메인(Security domain)

일단의 자원들과 그들 자원들을 접근하는 것이 인가된 시스템 엔티티들을 포함하여, 보안 모델과 보안 아키텍처에서 정의된 환경 또는 문맥. 하나 또는 그 이상의 보안 도메인들이 단일 관리 도메인(administrative domain)에 존재할 수 있다. 어떠한 보안 도메인을 정의하는 특징들은 시간이 지남에 따라 일반적으로 진화한다.

3.1.2.58. 보안 정책 표현(Security policy expression)

사용자(principal) 아이덴터티들과 또는 그것의 속성들을 허용가능한 동작들(actions)로 매핑하는 것. 보안 정책 표현은 종종 본질적으로 접근 제어 리스트가 된다.

3.1.2.59. 서비스 제공자(Service provider)

어떤 시스템 엔티티에게 주어진 역할. 이 역할을 통해 그 시스템 엔티티는 사용자들(principals) 또는 다른 시스템 엔티티들에게 서비스들을 제공한다.

3.1.2.60. 서비스 제공자 라이트(Service provider lite)

어떤 시스템 엔티티에게 주어진 역할. 이 역할을 통해 그 시스템 엔티티는 단지 필요한 SAML 프로토콜 부분만을 사용하여, 사용자들(principals) 또는 다른 시스템 엔티티들에게 서비스들을 제공한다.

3.1.2.61. 세션(Session)

상호작용 기간 동안 상호작용에 대한 일부 상태를 유지하는 것을 특징으로 하는, 종종 사용자를 포함하는(Principal), 시스템 엔티티들의 지속적인 상호작용.

3.1.2.62. 세션 기관(Session authority)

세션들과 관련된 상태를 어떤 시스템 엔티티가 유지할 때, 그 기관에게 주어진 역할.

3.1.2.63. 세션 참여자(Session participant)

어떤 기관이 적어도 하나의 세션 기관과 어떤 세션에 참여할 때, 그 기관에게 주어진 역할.

3.1.2.64. 사인-오프(Sign-off)

"로그아웃"을 참고한다.

3.1.2.65. 사인-온(Sign-on)

"로그인"을 참고한다.

3.1.2.66. 사이트(Site)

지리적인 또는 DNS 이름 관점에서 하나의 관리 도메인을 나타내는 비공식적인 용어. 이것은 어떤 관리 도메인의 특정 지리적인 또는 위상적인(topological) 부분을 나타낼 수도 있고, 또는 하나의 ASP 사이트에서 그렇듯이, 다중 관리 도메인들을 포괄할 수도 있다.

	Attribute Authority
ASN.1	Abstract Syntax Notation One
ASP	Application Service Provider
CA	Certification Authority
CMP	Certificate Management Protocol
CRL	Certificate Revocation List
DDDS	Dynamic Delegation Discovery System
DCE	Distributed Computing Environment
DNS	Domain Name System
ECP	Enhanced Client/Proxy
HTTP	HyperText Transfer Protocol
HTTPS	Secure HyperText Transport Protocol
IdP	Identity Provider
IdP Lite	Identity Provider Lite
IP	Internet Protocol
IPSEC	Internet Protocol SECurity
MD5	Message Digest algorithm 5
MIME	Multipurpose Internet Mail Extensions
NAPTR	Naming Authority PoinTeR

OID	Object IDentifier
PAC	Privilege Attribute Certificates
PAOS	Reverse SOAP
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PGP	Pretty Good Privacy
PKI	Public-Key Infrastructure
POP	Proof Of Possession
RA	Registration Authority
RSA	Rivest Shamir Adleman public key algorithm
SHA-1	Secure Hash Algorithm 1
SP	Service Provider
SPKI	Simple Public Key Infrastructure
SP Lite	Service Provider Lite
SSO	Single Sign On
TLS	Transport Layer Security protocol
URI	Uniform Resource Identifier
UTC	Coordinated Universal Time
UUID	Universal Unique IDentifier
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

4. 관례

SAML 2.0 에서 사용되는 키워드인 "해야만 한다(must)", "하지 않아야만 한다(must not)", "요구된다(required)", "일 것이다(shall)", "이지 않을 것이다(shall not)", "해야 한다(should)", "하지 않아야 한다(should not)", "권고된다(recommended)", "일(할) 수 있다(may)", "선택적인(optional)" 은 IETF RFC 2119 에서 설명된 것과 같이 해석되어야 한다.

SAML 2.0 은 W3C XML 스키마 Part 1, W3C 스키마 Part 2 와 그들 표준들의 규범적 텍스트(normative text)를 사용하여 XML 인코딩된 SAML 주장과 프로토콜 메시지들의 문법과 의미를 설명한다. SAML 2.0 의 SAML 스키마 문서들과 스키마 리스트 사이에 불일치가 발생할 경우에는, 스키마 문서가 높은 우선순위를 가진다. 어떤 경우에는,

SAML 2.0 이 스키마 문서에 의해 가리키는 것 이상의 제약을 가하는 경우가 있다는 것에 주의해야 한다.

5. 공통 데이터 타입 Common data types

다음 하부 절들은 SAML 스키마들에서 나타나는 공통된 데이터 타입들을 어떻게 사용하고 해석하는지를 정의한다.

5.1. 문자열 값(String Values)

모든 SAML 문자열 값들은 xs:string 타입을 가지며, 이 타입은 W3C XML 스키마 데이터타입들 표준에 내장(built in) 되어 있다. SAML 2.0 에서 별다른 언급이 없으면, SAML 메시지들에 존재하는 모든 문자열들은 적어도 하나 이상의 공백이 아닌 문자(non-whitespace)로 구성되어야만 한다.

이 SAML 2.0 또는 특정 프로파일들에서 별다른 언급이 없으면, XML 스키마 xs:string 타입을 가지거나 또는 이 문자열 타입으로부터 유도된 타입을 가지는 SAML 문서 내의 모든 요소들은 정확한 이진 비교(exact binary comparison)를 사용하여 비교되어야만 한다. 특히, SAML 구현과 배치(deployment)들은 대소문자를 구분하지 않는 문자열 비교, 공백의 정규화 또는 절단(trimming) 또는 숫자나 화폐와 같이 로케일에 따라 고유한(locale-specific) 변환 등에 의존하지 않아야만 한다. 이 요구는 W3C 문자열의 요구사항을 따르게 하기 위해 의도된 것이다.

만약 어떤 구현이 다른 문자 인코딩(encodings) 방식들을 사용하여 표현된 값들을 비교한다면, 그 구현은 두 값을 유니코드 문자 인코딩인 정규화 폼 C(Normalization Form C)로 변환하고 그것들에 대하여 정확한 이진 비교를 수행한 것과 같은 결과를 반환하는 비교 방법을 사용해야만 한다. 이 요구는 W3C 문자 모델과 특히, 유니코드-정규화 텍스트(Unicode-normalized Text)들에 대한 규칙을 따르게 하기 위해 의도된 것이다.

SAML 문서 형태로 받은 데이터와 외부 소스로부터 받은 데이터를 비교하는 응용(application)은 XML 에 대해 규정된 정규화 규칙을 고려해야만 한다. 요소들 내에

포함된 텍스트(text)는 라인의 끝이 라인피드 문자들(ASCII code 10Decimal)을 사용하여 나타내도록 정규화된다. 문자열들 (또는 문자열로부터 유도된 타입들)로 정의된 XML 속성 값들은 W3C XML 1.0, 3.3.3 절에서 설명된 것처럼 정규화된다. 모든 공백 문자들은 스페이스(blanks) (ASCII code 32Decimal)로 대체된다.

SAML 2.0 은 XML 속성 값들 또는 요소 내용에 대하여 대조(collation) 또는 정렬 순서를 정의하지 않는다. SAML 구현들은 값들에 대하여 특정한 정렬 순서들에 의존하지 않아야만 한다. 왜냐하면 처리에 참여한 호스트(host)들에서 설정된 로케일(locale)에 따라, 그 정렬 순서들이 달라지기 때문이다.

5.2. URI 값(URI Values)

모든 SAML URI 참조 값들은 **xs:anyURI** 타입을 가지며, 이 타입은 W3C XML 스키마 데이터타입들에 내장(built in) 되어 있다.

SAML 2.0 에서 다르게 지시되지 않는다면, SAML 에서 정의된 속성들 또는 요소들 내에서 사용되는 모든 URI 참조 값들은 적어도 하나 이상의 공백이 아닌 문자로 구성되어야만 하며, 절대경로를 표현하도록 요구된다.

SAML 2.0 은 상태코드, 포맷 타입, 속성과 시스템 엔티티 이름들 등과 같은 식별자들로써 URI 참조를 광범위하게 사용한다. 따라서, 똑 같은 URI 가 다른 시각에 다른 정보를 나타내는데 절대로 사용되지 않도록, URI 값들이 유일하고 동시에 일관되도록(consistent) 하는 것이 필수적이다.

5.3. 시간 값(Time Value)

모든 SAML 의 시각 값들은 **xs:dateTime** 타입을 가지며, 이 타입은 W3C XML 스키마 데이터타입들에 내장(built in) 되어 있다. 모든 SAML 시각 값들은 시간대(time zone) 컴포넌트가 없는 UTC 형식(form)으로 표현되어야만 한다.

SAML 시스템 엔티티들은 1000 분의 1 초보다 더 정교한 시각에 의존하지 않아야 한다. 구현들은 윤초(leap seconds)를 명기하는 시각 값들을 생성하지 않아야만 한다.

5.4. ID 와 ID 참조 값(ID and ID Reference Values)

xs:ID 단순 타입은 주장들, 요청 및 응답에 대한 SAML 식별자들(identifiers)을 선언하는데 사용된다. SAML 2.0 에서 xs:ID 타입으로 선언된 값들은 xs:ID 타입 자체의 정의에 의해 주어진 특성뿐만 아니라 다음과 같은 특성들을 만족시켜야만 한다:

- 식별자들은 할당하는 어떠한 기관(party)도 자신 또는 다른 기관(party)이 다른 데이터 객체에게 우연히 동일한 식별자를 할당할 수 있는 가능성이 거의 무시할 수 있을 정도라는 것을 보장해야만 한다.
- 어떤 데이터 객체가 자신이 특정한 식별자를 가지고 있다고 선언한 곳에, 그와 같은 선언은 정확히 하나만 있어야만 한다.

SAML 시스템 엔티티가 그것이 생성하는 식별자가 유일하다는 것을 보장하는 메커니즘은 시스템 구현에 의해 결정된다. 랜덤(random) 또는 의사랜덤(pseudorandom) 기술이 채택된 경우에, 임의적으로 선택된 두 개의 식별자가 서로 동일할 확률은 2^{-128} 보다 작거나 같아야만 하고, 2^{-160} 보다 작거나 같아야 한다. 이 요구는 128 비트와 160 비트 사이의 길이를 갖는 임의적으로 선택된 값을 인코딩함으로써 충족될 수 있다. 인코딩은 xs:ID 데이터타입을 정의하는 규칙을 준용해야만 한다. 의사랜덤 발생기는 서로 다른 시스템들 사이에 바람직한 유일성 특성을 보장하기 위해 유일한 값(material)으로 시드(seed)를 설정하여야만 한다.

xs:NCName 단순 타입은 SAML 에서 xs:ID 타입의 식별자들을 참조하는데 사용된다. 이렇게 하는 이유는 xs:IDREF 가 이런 목적으로 사용될 수 없기 때문이다. SAML 에서, SAML 식별자 참조에 의해 참조되는 요소는 식별자 참조가 사용되는 문서와 다른 문서에서 실질적으로 정의될 수 있다. xs:IDREF 를 사용하게 되면, 그것의 값이 동일한 XML 문서에 있는 어떤 요소의 ID 속성 값과 매치(match) 되어야 한다는 요구를 위반하게 될 것이다.

6. 추가적인 프로토콜 바인딩을 명기하기 위한 가이드라인

이 표준에서는 일단의 선택된 프로토콜 바인딩들을 정의한다. 그러나, 향후 다른 프로토콜 바인딩 집합들이 개발되는 것 역시 가능하다. 이번 절은 추가적인 바인딩을 명기하기를 원하는 제 3 자를 위한 가이드라인을 제공한다. 다음은 각각의 프로토콜 바인딩에서 반드시 기술되어야만 하는 이슈들의 체크 리스트이다.

- 세 가지 조각의 식별하는 정보를 명기한다: 프로토콜 바인딩을 유일하게 식별하는 URI, 저자에 대한 우편 또는 전자적인 연락 정보, 그리고 새로운 바인딩이 갱신하거나 또는 폐기시키는 이전에 정의된 바인딩 또는 프로파일에 대한 참조.
- 바인딩에 연관된 당사자들(parties) 사이의 상호작용들을 설명한다. 각각의 당사자에 의해 사용되는 어플리케이션과 각각의 상호작용에 관련된 프로토콜들에 대한 어떠한 제약들도 명확히 선언되어야 한다.
- 얼마나 많은 당사자들이 관련되어 있고 중개자들(intermediaries)이 관련되어 있는지를 포함하여, 각각의 상호작용에 관련된 당사자들을 식별한다(identify).
- 인증이 필요한지 여부와 수용할 수 있는 인증 타입들을 포함하여, 각각의 상호작용에 연관된 당사자들의 인증 방법을 명기한다.
- 메시지 무결성을 보장하는데 사용된 메커니즘들을 포함하여, 메시지 무결성에 대한 지원 수준(level)을 식별한다.
- 제 3 자가 SAML 메시지와 주장의 내용을 볼 수 있는지 여부와, 바인딩이 기밀성(confidentiality)를 요구하는지 여부, 그리고 기밀성을 달성하기 위해 권고되는 메커니즘들을 포함하여, 기밀성에 대한 지원 수준을 식별한다.
- 특히 SAML 주장 또는 메시지를 받고 처리하는 과정에서 발생하는 에러 상태들을 포함하여, 각각의 참여자에서 발생하는 에러 상태들 포함하는 에러 상태를 식별한다.
- 위협들에 대한 분석과 대처방안(countermeasures)를 포함하여, 보안 고려사항들을 식별한다.

- 특정한 통신 프로토콜을 수반하거나 또는 특정한 프로파일에서 사용된 바인딩에 대한 지원이 효과적이며 상호운용될 수 있는 방식으로 공표될(advertised) 수 있도록, 메타데이터 고려사항을 식별한다.

7. 프로토콜 바인딩

이 표준은 통신 프로토콜과 프레임워크에서 SAML 주장과 요청-응답 메시지를 사용하는 것에 대한 SAML 프로토콜 바인딩을 명기한다.

SAML 요청-응답 메시지 교환을 표준 메시징 또는 통신 프로토콜들로 매핑하는 것을 SAML 프로토콜 바인딩(또는 바인딩)이라고 부른다. SAML 요청-응답 메시지 교환들을 특정 통신 프로토콜인 <FOO>로 매핑하는 인스턴스(instance)는 *SAML 을 위한 <FOO> 바인딩* 또는 *SAML <FOO> 바인딩*이라고 명명한다.

예를 들어, SAML SOAP 바인딩은 SAML 요청과 응답 메시지 교환이, 어떻게 SOAP 메시지 교환으로 매핑되는지를 설명한다.

이 표준의 목적은, 서로 독립적으로 구현된 SAML 기준을 준용하는(SAML conforming) 소프트웨어가 표준 메세징 또는 통신 프로토콜을 이용할 때, 상호운용되는 것을 보장하기 위하여 일단의 선택된 집합의 바인딩들을 충분히 상세하게 명기하는 것이다.

별다르게 명기하지 않으면, 바인딩은 samlp:RequestAbstractType 과 samlp:StatusResponseType 타입들로부터 유도된 어떠한 SAML 프로토콜 메시지를 전송하는 것을 지원해야 한다. 더욱이, 바인딩이 “SAML 요청과 응답들”을 참조할 때, 그것은 이들 타입들로부터 유도된 임의의 프로토콜 메시지를 의미하는 것으로 이해되어야 한다.

이 표준에서는 다음과 같은 인쇄 관례가 사용된다: <ns:Element>, XMLAttribute, Datatype, OtherKeyword. 일부 경우에서, 꺽은 괄호(angle bracket)는 XML 요소가 아닌 비-터미널(non-terminal)을 가리키는데 사용된다. 이 때는 사용의 의도가 문맥에서 명확히 드러난다.

이 절은 SAML 표준의 일부를 구성하는 프로토콜 바인딩을 정의한다.

7.1. 일반적인 고려사항

이 절은 SAML 에 대하여 정의된 모든 프로토콜 바인딩들의 규범적인 특징들을 설명한다.

7.1.1. RelayState 의 사용

일부 바인딩에서는 상태 정보를 보존하고 운반하기 위해 “RelayState” 메커니즘을 정의한다. 이러한 메커니즘이 SAML 프로토콜의 초기 단계에서 요청 메시지를 운반하는데 사용되면, 이것은 이어서(subsequently) 응답을 운반하는데 사용되는 바인딩의 선택과 사용에 영향을 미치게 된다. 즉, 만약 SAML 요청 메시지가 RelayState 데이터를 수반하면, SAML 응답자는 반드시 RelayState 메커니즘을 지원하는 바인딩을 사용하여 그것의 SAML 프로토콜 응답을 반환해야만 하고, 응답자는 그것이 요청으로 수신했던 RelayState 데이터를 정확히, 대응되는 응답의 RelayState 파라미터에 위치시켜야만 한다.

7.1.2. 보안(Security)

만약 별다르게 언급되지 않는다면, 다음 보안 문장들은 모든 바인딩에 적용된다. 바인딩들은 또한 다음 보안 특징들에 대하여 추가적인 문장들을 만들 수 있다.

7.1.2.1. TLS 1.0 의 사용

만약 별다르게 언급되지 않는다면, TLS 1.0 (IETF RFC 2246)을 사용하는 어떠한 SAML 바인딩에서도, 서버들은 X.509 v3 인증서를 이용하여 클라이언트에게 인증되어야만 한다. 클라이언트는 일반적으로 인증서의 Subject DN 필드, SubjectAltName 속성 등과 같은 인증서의 내용을 검사함으로써, 인증서의 내용을 기반으로 서버의 아이덴터티를 확인해야만 한다.

7.1.2.2. 데이터 기원 인증(Data Origin Authentication)

메시지와 관련된 SAML 요청자와 SAML 응답자 둘 모두에 대한 인증은 선택적이며, 사용 환경에 따라 결정된다. SOAP 메시지 교환 레이어에서 또는 (예를 들어, 많은 바인딩들에서 사용되는 TLS 또는 HTTP 프로토콜과 같은) 기반이 되는 하부 프로토콜(underlying substrate protocol)로부터 가능한 인증 메커니즘이 데이터 기원(origin) 인증을 제공하는데 활용될 수 있다.

트랜스포트(transport) 인증은 SAML 프로토콜 메시지가 중개자를 통해 전달되는 바인딩에서는 단대단(end-end) 기원-인증 요구사항을 충족시키지 못 할 것이다. 이 경우, 메시지 인증이 권고된다.

SAML 은 그 자체로 당사자들이 서로에게 인증되는 메커니즘을 제공하지만, 또한, SAML 은 그 자체에 대한 보안을 제공하기 위해 다른 인증 메커니즘들을 사용할 수 있다.

7.1.2.3. 메시지 무결성

SAML 요청과 응답 둘 모두에 대한 메시지 무결성은 선택적이며, 사용 환경에 따라 결정된다. 기반이 되는 하부 프로토콜의 보안 레이어 또는 SOAP 메시지 교환 레이어에서의 메커니즘이 메시지 무결성을 보장하기 위해 사용될 수 있다.

트랜스포트 무결성은 SAML 프로토콜 메시지가 중개자를 통해 전달되는 바인딩에서는 단대단(end-end) 무결성 요구사항을 충족시키지 못 할 것이다. 이 경우, 메시지 무결성이 권고된다.

7.1.2.4. 메시지 기밀성

SAML 요청과 응답 둘 모두에 대한 메시지 기밀성은 선택적이며, 사용 환경에 따라 결정된다. 기반이 되는 하부 프로토콜의 보안 레이어 또는 SOAP 메시지 교환 레이어에서의 메커니즘이 메시지 기밀성을 보장하기 위해 사용될 수 있다.

트랜스포트 기밀성은 SAML 프로토콜 메시지가 중개자를 통해 전달되는 바인딩에서는 단대단(end-end) 기밀성 요구사항을 충족시키지 못 할 것이다. 이 경우, 메시지 기밀성이 권고된다.

7.1.2.5. 보안 고려사항

배치(deployment)하기 전에, 인증, 메시지 무결성과 기밀성 메커니즘에 대한 각각의 조합이 특정 프로토콜 교환과 배치 환경에서는 어떠한 취약점들이 발생하는지에 대하여 분석되어야 한다. IETF RFC 2617 은 기본 또는 메시지 다이제스트 인증 스킴(scheme)이 사용될 때 HTTP 환경에서 가능한 공격들을 설명한다. 캐싱이 보안에 미치는 영향에 대해서는 특별한 주의가 주어져야 한다.

7.2. SAML SOAP 바인딩

SOAP 은 중앙 집중적이지 않으며, 분산된(decentralized, distributed) 환경에서 구조화된 정보를 교환하려는 의도로 개발된 경량화된 프로토콜이다. SOAP 은 XML 기술들을 사용하여 다양한 기반 프로토콜들 위에서 교환될 수 있는 메시지 구조를 제공하는 확장 가능한 메시징 프레임워크를 정의한다. 이 프레임워크는 특정 프로그래밍 모델과 구현에 특정한 의미들에 의존적인지 않도록 설계되어왔다. SOAP 에 대한 두 가지 주요한 설계 목적은 단순성(simplicity)과 확장성(extensibility)이다. SOAP 은 분산 시스템에서 종종 발견되는 특징들을 메시징 프레임워크에서 생략함으로써 이들 목적을 충족하려 시도한다. 이러한 특징들은 “신뢰성(reliability)”, “보안성(security)”, “상호연관(correlation)”, “라우팅(routing)”, 그리고 “메시지 교환 패턴(Message Exchange Patterns, MEPS)”을 포함한다. 그러나 이러한 특징들이 위에서 언급할 것들로 한정되지는 않는다.

SOAP 메시지는 하나 또는 그 이상의 SOAP 중개자들을 통해 라우트되는 것이 가능하며, 하나의 SOAP 송신자로부터 하나의 SOAP 수신자까지 SOAP 노드들(nodes) 사이에서는 기본적으로 일방향 전송(one-way transmission)이다. 요청/응답에서부터, 다중, back-and-forth “conversational” 교환까지 더 복잡한 상호작용 패턴들을 구현한 어플리케이션에 의해 SOAP 메시지들이 결합될 것으로 예상된다.

SOAP 은 헤더(header)와 몸체(body) 부분을 포함하는 XML 메시지 봉투(envelope)을 정의하여, 데이터와 제어 정보가 전송되도록 한다. SOAP 은 또한 이 봉투와 관련된 처리 규칙과 SOAP 메시지 전송을 위한 HTTP 바인딩을 정의한다.

SAML SOAP 바인딩은 SAML 요청과 응답을 송신하고 수신하는데 어떻게 SOAP 을 사용하는지 그 방법을 정의한다.

SAML 에서와 같이, SOAP 은 다중 기반 트랜스포트(multiple underlying transports) 상에서 사용될 수 있다. 이 바인딩은 프로토콜에 독립적인 측면을 가지고 있지만, 또한 SOAP over HTTP 의 사용이 요구된다. 즉, SOAP over HTTP 는 반드시 구현되어야 하는 것이다.

7.2.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:SOAP

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding

7.2.2. SAML SOAP 바인딩의 프로토콜-독립적인 측면들

이번 절은 HTTP 와 같이, SOAP 메시지들이 그 위에서 전송되는 기반 프로토콜(underlying protocol)에 독립적인 SAML SOAP 바인딩 측면들(aspects)을 정의한다. 이 바인딩은 단지 SOAP 1.1 만을 사용하는 것을 지원한다는 것에 주의한다.

7.2.2.1. 기본 연산

SOAP 1.1 메시지들은 세 가지 요소들로 구성된다: 봉투, 헤더 데이터, 메시지 몸체. SAML 요청-응답 프로토콜 요소들은 SOAP 메시지 몸체 내에 포함되어야만 한다.

SOAP 1.1 은 또한 선택적인 데이터 인코딩 시스템을 정의한다. 이 시스템은 SAML SOAP 바인딩 내에서 사용되지 않는다. 이것은 SAML 메시지들이 표준 SAML

스키마로부터 SOAP 인코딩에 근거한 스키마로 다시 인코딩하지 않고, SOAP 을 사용하여 전송될 수 있다는 것을 의미한다.

SOAP 을 통해 SAML 대화(conversations)에 사용된 시스템 모델은 단순한 요청-응답 모델이다.

- SAML 요청자로 동작하는 시스템 엔티티는 SOAP 메시지의 몸체 내에 SAML 요청 요소를 포함하고, 그것을 SAML 응답자로 동작하는 시스템 엔티티에게 전송한다. SAML 요청자는 SOAP 메시지에 하나 이상의 SAML 요청을 포함하거나 SOAP 몸체에 어떠한 추가적인 XML 요소들을 포함하지 않아야만 한다.
- SAML 응답자는 다른 SOAP 메시지의 몸체 내에 SAML 응답 요소를 포함하여 반환하거나 또는 SOAP 오류(fault)를 생성해야만 한다. SAML 응답자는 SOAP 메시지에 하나 이상의 SAML 응답을 포함하거나 SOAP 몸체에 어떠한 추가적인 XML 요소들을 포함하지 않아야만 한다. 만약 어떤 이유로 SAML 응답자가 SAML 요청을 처리할 수 없다면, 응답자는 SOAP 오류를 생성해야만 한다. 확장 스키마를 발견할 수 없거나 또는 주체가 인가 질의에서 자원에 접근하도록 인가되지 않았다는 것과 같은 SAML 문제 도메인(problem domain)에 속한 에러들 때문에, SOAP 오류 코드들이 송신되지 않아야만 한다.

SOAP 메시지에 포함된 SAML 응답을 받았을 때, SAML 요청자는 SAML 응답자에게 오류 코드나 다른 에러 메시지들을 송신하지 않아야만 한다. 메시지 교환 형식이 단순한 요청-응답 패턴이기 때문에, 에러 조건과 같은 부가적인 항목들을 추가하는 것은 불필요하게 프로토콜을 복잡하게 할 수 있기 때문이다.

W3C SOAP 은 폐기된 네임스페이스를 포함하는 XML 스키마 표준의 초기 드래프트(draft)를 참조한다. SAML 요청자는 단지 최종(final) XML 스키마 네임스페이스만을 참조하는 SOAP 문서들을 생성해야만 한다. SAML 응답자들은 최종 XML 스키마 네임스페이스뿐만 아니라 SOAP 1.1(W3C SOAP 참조)내에서 사용되는 XML 스키마 네임스페이스, 둘 다를 처리할 수 있어야만 한다.

7.2.2.2. SOAP 헤더

SOAP 을 통해 이루어지는 SAML 대화에서 SAML 요청자는 SOAP 메시지에 임의의(arbitrary) 헤더들을 추가할 수 있다. 이 표준은 어떠한 추가적인 SOAP 헤더들로 정의하지 않는다.

주의: 다른 헤더들이 허용될 필요가 있는 것은 SAML 을 인지하고 있는 프로세스(SAML-aware process)의 제어 밖에 있는 일부 SOAP 소프트웨어와 라이브러리들이 SOAP 메시지에 헤더들을 추가할 수도 있기 때문이다. 또한, 일부 헤더들은 메시지의 라우팅을 요구하는 기반 프로토콜을 위해서 필요하거나 또는 메시지 보안 메커니즘에 의해 필요할 수도 있다.

SAML 응답자는 SAML 메시지를 스스로 올바르게 처리하기 위해, SOAP 메시지에 어떠한 헤더가 포함되어야 한다고 요구하지 않아야만 한다. 그러나 SAML 응답자는 기반 라우딩 또는 메시지 보안 요구사항을 가리키는 추가적인 헤더를 요구할 수 있다.

주의: 이와 같이 하는 이유는 별도의 헤더들을 요구하는 것이 SAML 표준을 단편화시키고 이에 따라 상호운용성을 해칠 수 있기 때문이다.

7.2.3. SOAP over HTTP 사용

SAML SOAP 바인딩을 준용한다고 주장하는 SAML 프로세서는 반드시 HTTP 를 이용하여 SOAP 으로 전송되는 SAML(SAML over SOAP over HTTP)을 구현해야만 한다. 이 절은 HTTP 헤더들, 캐싱 그리고 에러 보고(error reporting)을 포함하여, SOAP over HTTP 를 사용하는 것을 자세히 설명한다.

SOAP 을 위한 HTTP 바인딩은 W3C SOAP 6.0 절에서 설명된다. 그것은 SOAPAction 헤더가 SOAP HTTP 요청의 일부로써 사용되는 것을 요구한다. SAML 응답자는 이 헤더의 값을 의지하지 않아야만 한다. SAML 요청자는 SOAPAction 헤더의 값을 다음과 같이 설정할 수 있다.

```
http://www.oasis-open.org/committees/security
```

7.2.3.1. HTTP 헤더

SOAP over HTTP 상에서 이루어지는 SAML 대화에서 SAML 요청자는 HTTP 요청에 임의의(arbitrary) 헤더들을 추가할 수 있다. 이 표준은 어떠한 추가적인 HTTP 헤더들로 정의하지 않는다.

주의: 다른 헤더들이 허용될 필요가 있는 것은 SAML을 인지하고 있는 프로세스(SAML-aware process)의 제어 밖에 있는 일부 HTTP 소프트웨어와 라이브러리들이 HTTP 메시지에 헤더들을 추가할 수도 있기 때문이다. 또한, 일부 헤더들은 메시지의 라우팅을 요구하는 기반 프로토콜을 위해서 필요하거나 또는 메시지 보안 메커니즘에 의해 필요할 수도 있다.

SAML 응답자는 SAML 메시지를 스스로 올바르게 처리하기 위해 HTTP 요청에 어떠한 헤더가 포함되어야 한다고 요구하지 않아야만 한다. 그러나 SAML 응답자는 기반 라우딩 또는 메시지 보안 요구사항을 가리키는 추가적인 헤더를 요구할 수 있다.

주의: 이와 같이 하는 이유는 별도의 헤더들을 요구하는 것이 SAML 표준을 단편화시키고 이에 따라 상호운용성을 해칠 수 있기 때문이다.

7.2.3.2. 캐싱

HTTP 프락시들은 SAML 프로토콜 메시지들을 캐시하지 않아야 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1을 사용할 때, 요청자는 다음과 같이 해야 한다:

- "no-cache, no-store"로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- "no-cache"로 설정된 Pragma 헤더 필드를 포함해야만 한다.

HTTP 1.1을 사용할 때, 응답자는 다음과 같이 해야 한다:

- “no-cache, no-store, must-revalidate, private”으로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- “no-cache”로 설정된 Pragma 헤더 필드를 설정해야만 한다.
- Last_Modified 또는 ETag 헤더와 같이 검증자(Validator)를 포함하지 않아야 한다.

7.2.3.3. 에러 보고

SAML 요청자와 메시지를 교환하는 것을 거부하는 SAML 응답자는 “403 Forbidden” 응답을 반환해야 한다. 이 경우, HTTP 몸체의 내용은 중요하지 않다.

W3C SOAP 6.2 절에서 설명돼 듯이, SOAP 요청을 처리하는 중에 SOAP 에러가 발생하는 경우에는, SOAP HTTP 서버는 “500 Internal Server Error”를 반환하고, 응답으로 SOAP <SOAP-ENV:fault> 요소를 가지는 SOAP 메시지를 포함해야만 한다. 제어권(control)이 SAML 프로세서로 넘어오기 전에 발견된 SOAP 관련 에러에 대하여, 또는 (예를 들어, SOAP XML 네임스페이스가 틀리거나, SAML 스키마를 위치시킬 수 없는 경우, SAML 프로세서가 예외를 던지거나 기타 등등과 같이) SOAP 프로세서가 내부 에러를 보고할 때, 이 타입의 에러가 반환되어야 한다.

SAML 처리시 에러가 발생하는 경우, SOAP HTTP 서버는 “200 OK”을 가지고 응답해야만 하고 SOAP 몸체 내에 SAML 응답으로 SAML 에 명기한 <samlp:Status> 요소를 포함해야만 한다. <samlp:Status> 요소는 스스로 SOAP 몸체에 나타나지 않고, 단지 일부 종류의 SAML 응답 내에서만 나타난다는 사실에 주의한다.

SAML 상태 코드들에 대한 더 많은 정보를 위해서는 SAML 2.0 주장과 프로토콜 표준을 참조한다.

7.2.3.4. 메타데이터 고려사항

특정 프로토콜이나 또는 프로파일에 대하여 SOAP 메시지에 포함된 요청들이 보내지는 URL 엔드포인트(endpoint)들을 가리킴으로써 또는 WSDL 포트/엔드포인트 정의를 가지고, SOAP 바인딩을 지원한다는 사실을 반영해야 한다.

7.2.3.5. SOAP over HTTP 를 이용하는 SAML 메시지 교환 예

다음은 SAML 속성 기관으로부터 속성 문장을 포함하는 주장을 요구하는 질의의 예이다.

```
POST /SamlService HTTP/1.1
Host: www.example.com
Content-Type: text/xml
Content-Length: nnn
SOAPAction: http://www.oasis-open.org/committees/security
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:AttributeQuery xmlns:samlp:="…"
xmlns:saml="…" xmlns:ds="…" ID="_6c3a4f8b9c2d"
Version="2.0"
IssueInstant="2004-03-27T08:41:00Z"
<ds:Signature> … </ds:Signature>
<saml:Subject>
…
</saml:Subject>
</samlp:AttributeQuery>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

다음은 대응되는 응답의 예이다. 이 응답은 요청한 대로, 속성 문장을 포함하는 주장을 제공한다.

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: nnnn
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:Response xmlns:samlp="…" xmlns:saml="…" xmlns:ds="…"
ID="_6c3a4f8b9c2d" Version="2.0"
IssueInstant="2004-03-27T08:42:00Z">
<saml:Issuer>https://www.example.com/SAML</saml:Issuer>
<ds:Signature> … </ds:Signature>
<Status>
<StatusCode Value="…"/>
</Status>
<saml:Assertion>
<saml:Subject>
…
</saml:Subject>
<saml:AttributeStatement>
```

```
                                    ...
                                </saml:AttributeStatement>
                            </saml:Assertion>
                        </samlp:Response>
                    </SOAP-Env:Body>
                </SOAP-ENV:Envelope>
```

7.3. Reverse SOAP(PAOS) 바인딩

이 바인딩은 SOAP 을 위한 Reverse HTTP 바인딩 표준(PAOS:2003 참조)을 활용한다. 구현자들은 이 문서에서 명기된 규칙들뿐만 아니라, PAOS 에서 명기된 일반적인 처리 규칙들에 부합하도록 구현하여야만 한다. 두 가지 규칙에서 충돌이 발생할 경우, Liberty Alliance POAS:2003 이 기준이 된다.

7.3.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:PAOS

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.3.2. 개요

Reverse SOAP 바인딩은 HTTP 요청자가 SAML 요청자에게 SOAP 응답자 또는 SOAP 중개자로써 동작할 수 있는 능력이 있다는 것을 알릴(advertise) 수 있도록 해 주는 메커니즘이다. HTTP 요청자는 SAML 요청자는 HTTP 응답 안에 SAML 요청을 담은 SOAP 봉투를 포함하여 HTTP 요청자에게 전달하는 패턴을 지원할 수 있으며, HTTP 요청자가 뒤따르는(subsequent) HTTP 요청 안에 SAML 응답을 담은 SOAP 봉투를 포함하여 응답한다. 이 메시지 교환 패턴은 ECP SSO 프로파일에서 정의된 사용 케이스를 지원한다. ECP SSO 프로파일에서는 HTTP 요청자가 인증 교환에서 중개자가 된다.

7.3.3. 메시지 교환

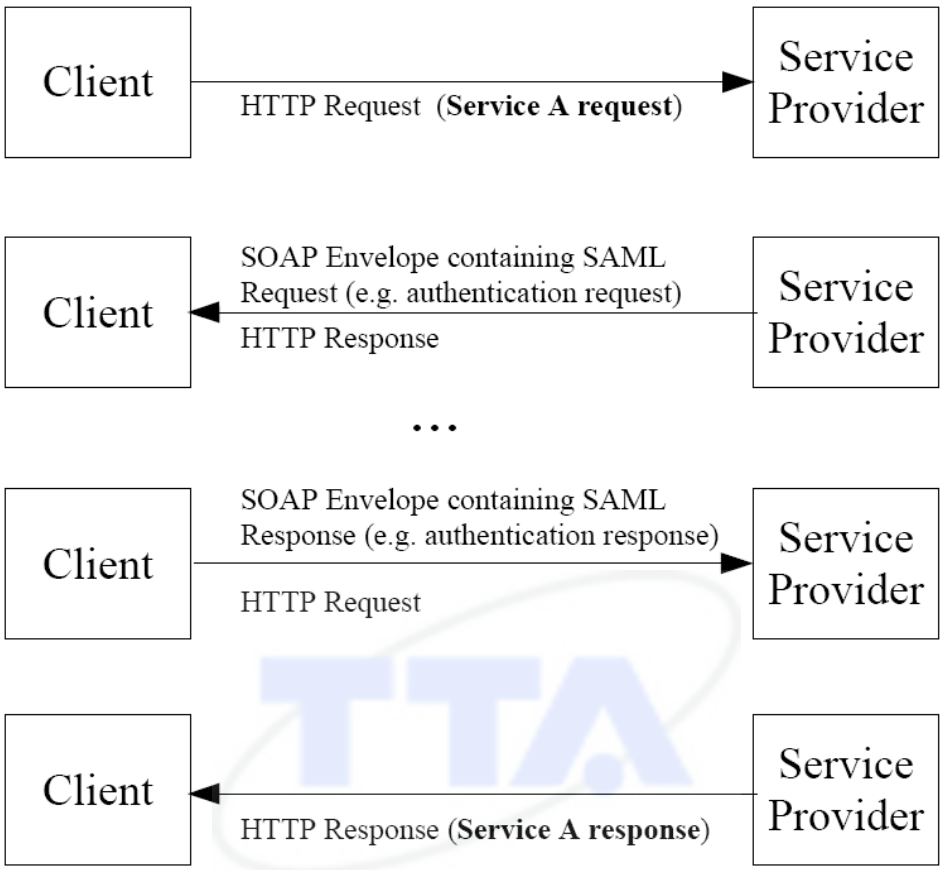
PAOS 바인딩은 두 개의 컴포넌트 메시지 교환 패턴을 포함한다:

1. HTTP 요청자는 HTTP 요청을 SAML 요청자에게 전달한다. SAML 요청자는 SAML 요청 메시지를 포함하는 SOAP 봉투를 포함하는 HTTP 응답을 가지고 응답한다.
2. 뒤이어, HTTP 요청자는 원래의 SAML 요청자에게 SAML 응답 메시지를 포함하는 SOAP 봉투를 지닌 HTTP 요청을 전달한다. SAML 요청자는 단계 1 의 원래의 서비스 요청에 대한 응답으로 HTTP 응답을 가지고 응답한다.

ECP 프로파일은 서비스가 제공되기 전에, 클라이언트를 서비스 제공자에게 인증시키기 위해 PAOS 바인딩을 사용한다. 이것은 (그림 3-1 에서 설명되는 것처럼 다음 단계로 수행된다.

1. 클라이언트가 HTTP 요청을 사용하여 서비스를 요청한다.
2. 서비스 제공자가 SAML 인증 요청을 가지고 응답한다. 이것은 HTTP 응답으로 운반되는 SOAP 요청을 사용하여 전달된다.
3. 클라이언트는 SAML 인증 응답을 운반하는 SOAP 응답을 반환한다. 이것은 새로운 HTTP 요청을 사용하여 전달된다.

4. 서비스 제공자 인증과 인가가 성공이라고 가정하면, 서비스 제공자는 HTTP 응답으로 원래의 서비스 요청에 대하여 응답할 수 있다.



(그림 3-1) PAOS 바인딩 메시지 교환

HTTP 요청자는 PAOS:2003 표준에서 정의된 HTTP 헤더들을 사용하여, 그것의 HTTP 요청에 이 reverse SOAP 바인딩을 처리할 수 있는 능력이 있다는 것을 명확하게 알린다:

- HTTP Accept 헤더 필드는 “application/vnd.paos+xml” 내용 타입을 수용할 수 있다는 것을 가리켜야만 한다.
- HTTP PAOS 헤더 필드는 존재해야 하며, 지원하는 PAOS 버전이 최소한 “urn:liberty:paos:2003-08”이 되거나 또는 그 이상이 되도록 PAOS 버전을 명기해야만 한다.

서비스 값과 같은 추가적인 PAOS 헤더들은 PAOS 바인딩을 사용하는 프로파일들에 의해 명기될 수 있다. HTTP 요청자는 임의의 헤더들을 HTTP 요청에 추가할 수 있다.

이 바인딩은 RelatState 메커니즘을 정의하지 않는다는 것에 주의한다. 따라서, 만약 필요하다면, 이 바인딩을 이용하는 특정 프로파일들은 RelayState 와 같은 메커니즘을 정의해야만 한다. SOAP 헤더가 이 목적으로 사용될 수 있다.

이번 절은 메시지 교환의 두 단계에 대하여 좀 더 상세히 설명한다.

7.3.3.1. HTTP 요청, SOAP 응답으로 SAML 요청

임의의 HTTP 요청에 대한 응답으로, HTTP 응답자는 이 바인딩을 사용하여 SAML 요청 메시지를 반환할 수 있다. 이 때, HTTP 응답자는 어떠한 추가적인 몸체 내용을 가지고 있으며, SOAP 몸체에 단일한 SAML 요청 메시지를 포함하는 SOAP 1.1 봉투를 반환한다. SOAP 봉투는 PAOS, SAML 프로파일 또는 추가적인 표준들에 의해 정의된 임의의 SOAP 헤더들을 포함할 수 있다.

SAML 요청 메시지가 HTTP 요청자에게 배달되는 반면, 특정 프로파일에서 정의된 것처럼 HTTP 요청자가 중개자로 동작하는 경우와 같이, 실질적으로 의도된 수신자가 또 다른 시스템 엔티티일 수 있다.

7.3.3.2. SOAP 요청으로 SAML 응답, HTTP 응답

HTTP 요청자가 PAOS 바인딩을 사용하여 의도된 수신자에게 SAML 응답 메시지를 배달할 때, HTTP 요청자는 HTTP 요청에 있는 SOAP 봉투의 SOAP 몸체에 단 하나의 요소로 SAML 응답 메시지를 위치시킨다. HTTP 요청자는 SAML 응답의 개시자(originator)일 수도 있고 그렇지 않을 수도 있다. SOAP 봉투는 PAOS, SAML 프로파일 또는 추가적인 표준들에 의해 정의된 임의의 SOAP 헤더들을 포함할 수 있다. SAML 교환은 완전한 것으로 간주되며, HTTP 응답은 이 바인딩에서는 명기되지 않는다.

프로파일들은 이 바인딩에서 커버하는(cover) 교환들 중에, SOAP 이 아닌 응답의 HTTP 내용에 대하여 추가적인 제약을 정의할 수 있다.

7.3.4. 캐싱

HTTP 프락시들은 SAML 프로토콜 메시지들을 캐시하지 않아야 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1 을 사용할 때, SAML 프로토콜 메시지를 송신하는 요청자는 다음과 같이 해야 한다:

- "no-cache, no-store"로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- "no-cache"로 설정된 Pragma 헤더 필드를 포함해야만 한다.

HTTP 1.1 을 사용할 때, SAML 프로토콜 메시지를 반환하는 응답자는 다음과 같이 해야 한다:

- "no-cache, no-store, must-revalidate, private"으로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- "no-cache"로 설정된 Pragma 헤더 필드를 설정해야만 한다.
- Last_Modified 또는 ETag 헤더와 같이 검증자(Validator)를 포함하지 않아야 한다.

7.3.5. 보안 고려사항

PAOS 바인딩에서 HTTP 요청자는 SOAP 중개자로 동작할 수 있으며, 요청자가 그렇게 동작할 때, 기원 인증(origin authentication), 무결성 및 기밀성에 대한 트랜스포트 레이어(transport layer) 보안이 단대단 보안 요구사항을 만족시키지 못할 지도 모른다. 이 경우, SOAP 메시지 레이어에서의 보안이 권고된다.

7.3.5.1. 에러 보고

표준 HTTP 와 SOAP 에러 관례(conventions)가 준수되어야만 한다. SAML 처리 중에 발생한 에러들은 HTTP 또는 SOAP 레이어에 보고되지 않아야만 하고, 에러 <samlp:Status> 요소를 가지는 SAML 응답 메시지를 사용하여 처리되어야만 한다.

7.3.5.2. 메타데이터 고려사항

특정 프로토콜이나 또는 프로파일에 대하여 SOAP 봉투에 포함된 HTTP 요청들과/또는 SOAP 봉투에 포함된 SAML 프로토콜 메시지들이 보내지는 URL 엔드포인트들을 가리킴으로써, PAOS 바인딩을 지원한다는 사실을 반영해야 한다. 단일 엔드포인트 또는 요청과 응답이 분리된 엔드포인트들이 제공될 수 있다.

7.4. HTTP Redirect 바인딩

HTTP Redirect 바인딩은 SAML 프로토콜 메시지들이 URL 파라미터들 내에서 전송될 수 있도록 해 주는 메커니즘을 정의한다. 허용가능한 URL 길이는 이론적으로는 무한하지만, 실질적으로는 예측할 수 없을 정도로 제약되어 있다. 따라서, XML 메시지를 URL 로 운반하기 위해서는 특수화된 인코딩 방식이 필요하다. 더 크거나 또는 더 복잡한 메시지 내용은 HTTP POST 또는 Artifact 바인딩을 사용하여 송신될 수 있다.

두 가지 다른 바인딩을 사용하는 단일한 프로토콜 교환에서 요청과 응답 메시지를 전송하기 위해, 이 바인딩은 HTTP POST 바인딩(3.5 절 참조)와 HTTP Artifact 바인딩(3.6 절 참조)과 결합될 수 있다.

이 바인딩은 메시지 인코딩의 사용을 수반한다. 이 바인딩의 정의는 하나의 특정 메시지 인코딩 정의를 포함하는 반면, 다른 메시지 인코딩들이 정의되어 사용될 수도 있다.

7.4.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.4.2. 개요

HTTP Redirect 바인딩은 SAML 요청자와 응답자가 중개자로써 (IETF RFC 2616 에 정의된 것처럼) HTTP 사용자 에이전트를 사용하여 통신하는 경우에 사용될 목적을 가진다. 예를 들어, 만약 통신 당사자들이 직접적인 통신 경로를 공유하지 못하는 경우, 이 바인딩이 필요할 수 있다. 만약 사용자 에이전트가 응답자에게 인증 받아야 할 때처럼, 요청을 이행하기 위해 응답자가 사용자 에이전트와 상호작용을 해야 한다면, 이 바인딩이 또한 필요해 질 수 있다.

일부 HTTP 사용자 에이전트는 프로토콜 교환에 있어 좀 더 능동적인 역할을 수행할 능력을 가질 수 있으며, SOAP 과 reverse SOAP 바인딩과 같이 HTTP 를 사용하는 다른 바인딩을 지원할 수 있다. 이 바인딩은 일반적인 웹 브라우저의 능력을 벗어나는 어떤 것도 가정하지 않는다.

7.4.3. RelayState

RelayState 는 이 바인딩으로 전송되는 SAML 프로토콜 메시지와 함께 포함될 수 있다. 이 값은 길이로는 80 바이트를 넘지 않아야만 하고, 메시지 전송 중에 존재할지도 모르는 다른 어떠한 보호 방법과도 독립적으로 메시지를 생성하는 엔티티에 의해 무결성이 보호되어야 한다. 서명은 공간 제약을 고려하면 실질적인 방안이 되지 못한다. 그러나, 그 값이 제 3 자 변조에 노출되기 때문에, 엔티티는 체크섬(checksum), 의사랜덤(pseudo-random) 값 또는 유사한 방식을 이용하여 그 값이 변조되지 않았음을 보장해야 한다.

만약, SAML 요청 메시지가 RelayState 데이터를 수반한다면, SAML 응답자 또한 RelayState 메커니즘을 지원하는 바인딩을 사용하여 그것의 SAML 프로토콜 응답을 반환해야만 하고, SAML 응답자는 응답 안에 대응되는 RelayState 파라미터를 위치시키고, 그것이 요청에서 수신한 것과 정확히 같은 데이터를 파라미터 값으로 설정해야만 한다.

만약 그러한 값이 SAML 요청 메시지와 함께 포함되지 않는다면, 또는 만약 SAML 응답 메시지가 대응되는 요청 없이 생성되었다면, SAML 응답자는 프로파일의 사용 또는 당사자들 사이의 이전 협정에 따라 수신자에 의해 해석되는 RelayState 데이터를 포함시킬 수 있다.

7.4.4. 메시지 인코딩

메시지들은 URL 인코딩 기술을 이용하여 이 바인딩에서 사용되도록 인코딩되고 HTTP GET 방식을 사용하여 전송된다. 효과가 있는 제약사항들에 따라, 여러 가지 방식들이 XML 을 URL 로 인코드하는데 사용될 수 있다. 이 표준은 다른 방식들을 배제하지 않으면서 하나의 방식을 정의한다. 바인딩 엔드포인트들은 적절할 때, 메타데이터를 사용하여 그들이 어떠한 인코딩들을 지원하는 지를 가리켜야 한다. 특정 인코딩들은 정의될 때, 유일하게 그것을 식별할 수 있는 URI 를 가지도록 해야만 한다. 모든 가능한 SAML 메시지들이 특정 규칙 집합으로 인코딩될 수 있도록 하는 것이 요구되지는 않지만, 그러나 규칙들이 어떠한 메시지 또는 내용이 그렇게 인코딩될 수 있는지 또는 될 수 없는지를 명확히 나타나야만 한다.

URL 인코딩은 URL 질의 문자열(URL query string) 내에 메시지 전체를 위치시켜야만 한다. 그리고 메시지 수신자의 엔드포인트를 위하여 URL 의 나머지는 보존(reserve)해야만 한다.

SAMLEncoding 이라고 불리는 질의 문자열 파라미터는 사용되는 인코딩 메커니즘을 식별하기 위해 예약되어 있다. 만약, 이 파라미터가 생략되면, 그 값은 urn:oasis:names:tc:SAML:2.0:bindings:URL-Encoding:DEFLATE 이라고 가정된다.

이 바인딩을 지원하는 모든 엔드포인트들은 다음 하부 절에서 설명되는 DEFLATE 인코딩을 지원해야만 한다.

7.4.4.1. DEFLATE 인코딩

Identification: urn:oasis:names:tc:SAML:2.0:bindings:URL-Encoding:DEFLATE

SAML 프로토콜 메시지는 DEFLATE 압축 방식(IETF RFC 1951 참조)을 통해 URL 로 인코드될 수 있다. 이러한 인코딩에서는, 원래의 SAML 프로토콜 메시지를 XML 직렬화하는데 다음 절차가 적용되어야 한다.

1. <ds:Signature> XML 요소 자체를 포함한, SAML 프로토콜 메시지 상의 어떠한 서명도 제거되어야만 한다. 만약 메시지의 내용이 서명된 SAML 주장과 같이 또 다른 서명을 포함한다면, 이 내장된 서명(embedded signature)는 제거되지 않는다는 것에 주의한다. 따라서 서명된 내용을 포함하는 SAML 프로토콜 메시지들은 이 메커니즘을 통해 인코드되지 않아야 한다.
2. 그 다음, DEFLATE 압축 메커니즘이 IETF RFC 1951 에서 명기된 것처럼 원래의 SAML 프로토콜 메시지의 나머지 전체 XML 내용에 적용된다.
3. 그 다음, 압축된 데이터는 IETF RFC 2045 에서 명기된 규칙에 따라 base64-인코딩된다. 라인피드(Linefeed)나 다른 공백(whitespace)들은 결과로부터 반드시 제거되어야만 한다.
4. 그 다음, base64-인코딩된 데이터는 URL-인코드되고 질의 문자열 파라미터로써 URL 에 추가된다. 만약 메시지가 SAML 요청 메시지인 경우에는 질의 문자열 파라미터는 SAMLRequest 가 되고, 만약 메시지가 SAML 응답 메시지인 경우에는, 질의 문자열은 SAMLResponse 가 된다.
5. 만약 RelayState 데이터가 SAML 프로토콜 메시지에 수반되면, RelaySate 는 URL-인코드되고 RelayState 라고 명명된 추가적인 질의 문자열 파라미터에 놓여야만 한다.
6. 만약 원래의 SAML 프로토콜 메시지가 XML 디지털 서명을 사용하여 서명되었다면, 위에서 명기된 것처럼 인코드된 데이터를 커버하는 새로운 서명이 아래 나열된 규칙을 사용하여 추가되어야만 한다.

XML 디지털 서명들은 공간 문제 때문에, 위의 규칙들에 따라 직접적으로 URL-인코드되지는 않는다. 만약 기반 SAML 프로토콜 메시지가 XML 서명으로 서명된다면, URL-인코드된 형식의 메시지는 다음과 같이 서명되어야만 한다.

1. 서명 알고리즘 식별자는 SigAlg 로 명명된 추가적인 질의 문자열 파라미터로써 포함되어야만 한다. 이 파라미터의 값은 XML Signature 또는 알고리즘을 지배하는 어떠한 표준에 의해 명기된 URI 이어야만 한다. 이 URI 는 URL-인코드된 SAML 프로토콜 메시지를 서명하는 데 사용된 알고리즘을 식별한다.

a.

```
SAMLRequest=value&RelayState=value&SigAlg=value
SAMLResponse=value&RelayState=value&SigAlg=value
```

b. 그 결과 생성되는 문자열의 바이트들이 서명 알고리즘의 입력이 되는 octet 문자열이 된다. 원래의 질의 문자열에 있는 다른 어떠한 내용도 포함되지 않으며 서명되지 않는다.

c. 서명 값은 공백이 제거되며 base64 인코딩(IETF RFC 참조)을 사용하여 인코드되어야만 하고 그리고 Signature 로 명명된 질의 문자열 파라미터로 포함되어야만 한다. base64-인코드된 서명 값에서 일부 문자들은 추가되기 전에 그 자체로 URL-인코딩될 수 있다는 사실에 주의한다.

d. 다음 서명 알고리즘들(W3C Signature 참조)과 그들의 URI 대표는 이 인코딩 메커니즘에서 지원되어야만 한다:

- DSAwithSHA1 - http://www.w3.org/2000/09/xmldsig#dsa-sha1
- RSAwithSHA1 - http://www.w3.org/2000/09/xmldsig#rsa-sha1

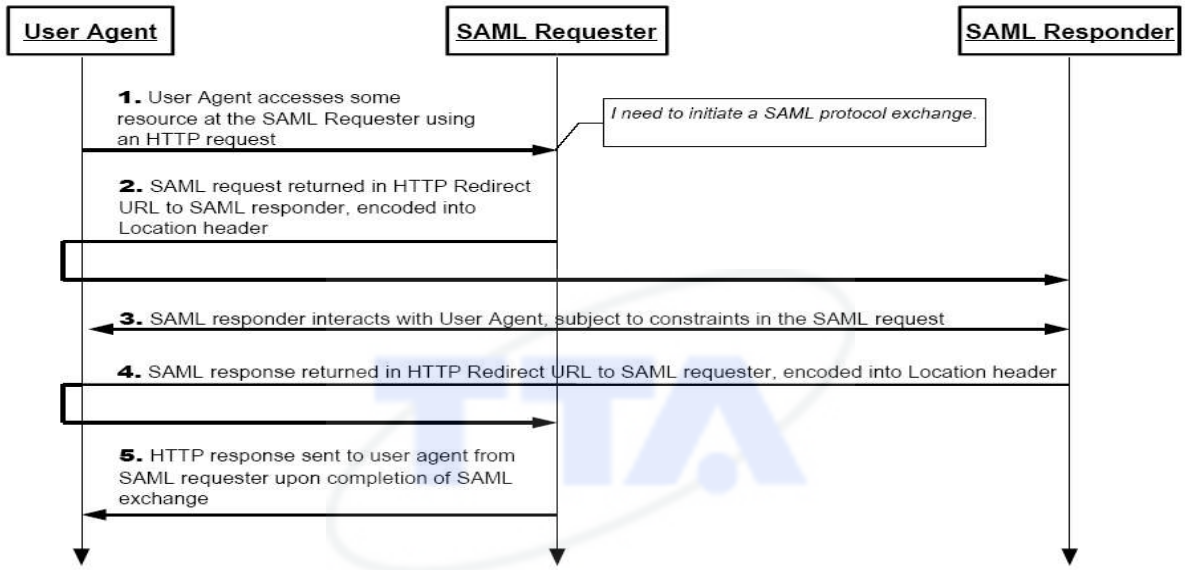
서명을 검증할 때, 결과로 생성되는 URL 의 질의 문자열 파라미터들이 검증되는 순서는 이 바인딩에서 규정하지 않는다. 파라미터들은 어떠한 순서로도 나타날 수 있다. 만약 있다면, 서명을 검증하기 전에, 의지하는 측은 검증하려는 파라미터 값들이 위의 서명 규칙들에서 요구되는 것과 같은 순서인 것을 보장해야만 한다.

URL-인코딩은 정규형(canonical)이 아니다: 즉, 주어진 값에 대하여 여러 가지 합법적인 인코딩들이 있을 수 있다. 따라서, 의지하는 측은 질의 문자열을 통해 그것이 수신했던 원래의 URL-인코드된 값들을 사용하여 검증 단계를 수행해야만 한다. 결과로 나타나는 인코딩이 서명자의 인코딩과 일치하지 않을 수 있기 때문에, 파라미터들이 소프트웨어에 의해 처리된 후에 다시 인코드(re-encode) 하는 것이 충분하지는 않다.

만약 RelayState 값이 없다면, 공백 파라미터 이름으로써 포함되는 것이 아니고 전체 파라미터는 서명 계산에서 생략되어야 한다.

7.4.5. 메시지 교환

이 바인딩을 통해 SAML 대화에 사용된 시스템 모델은 요청-응답 모델이다. 그러나 이들 메시지들은 HTTP 응답으로 사용자 에이전트에게 송신되고 HTTP 요청으로 메시지 수신자에게 배달된다. 이들 교환이 발생하기 전, 사이 그리고 이후의 HTTP 상호작용은 명기되지 않는다. SAML 요청자와 SAML 응답자는 둘 다 HTTP 응답자라고 가정한다. 다음 순서도는 교환되는 메시지를 설명한다.



1. 초기에, 사용자 에이전트는 시스템 엔티티에게 임의의 HTTP 서비스를 요청한다. 요청을 처리하는 과정에, 시스템 엔티티는 SAML 프로토콜 교환을 시작할 것을 결정한다.
2. SAML 요청자로 동작하는 시스템 엔티티는 SAML 요청을 반환함으로써 단계 1 에서 사용자 에이전트로부터 받은 HTTP 요청에 응답한다. SAML 요청은 HTTP 응답의 Location 헤더로 인코드되어 반환되며, HTTP 상태는 303 또는 302 둘 중에 하나이어야만 한다. SAML 요청자는 IETF RFC 2616 에서 정의된 것처럼, 사용자 에이전트의 메시지 송신을 용이하게 하기 위해 HTTP 응답에 추가적인 표현(presentation)과 내용을 포함할 수 있다. 사용자 에이전트는 SAML 응답자에게 HTTP GET 요청을 넘겨줌으로써 SAML 요청을 배달한다.

3. 일반적으로, SAML 응답자는 SAML 응답을 즉시 반환함으로써 SAML 요청에 응답할 수 있거나 또는 요청을 이행하기 위해 필요한 사용자 에이전트와 일련의 상호작용을 용이하게 하기 위한 임의의 내용을 반환할 수 있다. 특정 프로토콜과 프로파일은 (예를 들어 <samlp:AuthRequest>에서 IsPassive 속성과 같이) 이러한 상호작용을 허용하려는 요청자의 의향 정도(level of willingness)를 가리키는 메커니즘을 포함할 수 있다.
4. 결국, 응답자는 SAML 요청자에게 반환되도록, SAML 응답을 사용자 에이전트에게 반환해야 한다. SAML 응답은 단계 2 에서의 SAML 요청에 대하여 설명된 것과 동일한 방식으로 반환된다.
5. SAML 응답을 받자마자, SAML 요청자는 사용자 에이전트에게 임의의 HTTP 응답을 반환한다.

7.4.5.1. HTTP 와 캐싱 고려사항

HTTP 프락시들과 사용자 에이전트 중개자는 SAML 프로토콜 메시지들을 캐시하지 않아야 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1 을 사용하여 SAML 프로토콜 메시지를 반환할 때, HTTP 응답자는 다음과 같이 해야 한다:

- “no-cache, no-store”로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- “no-cache”로 설정된 Pragma 헤더 필드를 포함해야만 한다.

HTTP 헤더 사용에 대한 다른 제약사항은 없다.

7.4.5.2. 보안 고려사항

사용자 에이전트 중개자가 존재한다는 것은 요청자와 응답자가 단대단 인증, 무결성과 기밀성을 위해 트랜스포트 레이어를 의지할 수 없다는 것을 의미한다. 만약 인코딩 방법이 서명을 위한 수단을 명기한다면, URL-인코드된 메시지들은 기원 인증과 무결성을 제공하기 위해 서명될 수 있다.

만약 메시지가 서명되면, 프로토콜 메시지의 루트 SAML 요소에 있는 Destination XML 속성은 송신자가 사용자 에이전트에게 메시지를 그 곳으로 배달하라고 명령하는 URL 을 포함해야만 한다. 그 다음, 수신자는 그 값이 메시지가 수신된 위치와 일치하는지 검증해야만 한다.

만약 요청 또는 응답의 내용이 사용자 에이전트 중개자에게 노출되지 않아야 한다면, 이 바인딩은 사용되지 않아야 한다. 만약 그렇지 않다면, SAML 요청과 SAML 응답 둘 모두의 기밀성은 선택적이며 사용 환경에 따라 결정된다. 만약 기밀성이 필요하다면, TLS 1.0 은 사용자 에이전트와, SAML 요청자와 응답자 사이에서 전송되는 메시지를 보호하기 위해 사용되어야 한다.

URL-인코드되는 메시지들은 HTTP "Referer" 헤더뿐만 아니라 다양한 HTTP 로그들에 노출될 수 있다.

배치하기 전에, 인증, 메시지 무결성과 기밀성 메커니즘에 대한 각각의 조합이 특정 프로토콜 교환과 배치 환경에서 어떠한 취약점들이 발생하는지에 대하여 분석되어야 한다.

일반적으로, 이 바인딩은 서명을 통해 얻어지는 메시지 수준의 인증과 무결성 보호를 의지하지만, 사용자 에이전트 중개자에게 메시지의 기밀성을 유지하는 기능을 지원하지는 않는다.

7.4.6. 에러 보고

SAML 요청자와 메시지를 교환하는 것을 거부하는 SAML 응답자는 urn:oasis:names:tc:SAML:2.0:status:RequestDenied 를 차상위 수준(second-level) <samlp:StatusCode> 값으로 가지는 SAML 응답 메시지를 반환해야 한다.

메시지 교환 중에 발생하는 HTTP 상호작용들은 SAML 처리시 발생하는 실패를 가리키기 위해 HTTP 에러 상태 코드들을 사용하지 않아야만 한다. 왜냐하면 사용자 에이전트가 SAML 프로토콜 교환에 참여하는 단 하나의 당사자는 아니기 때문이다. 또한 SAML 메타데이타 표준을 참조한다. SAML 상태 코드에 대한 더 많은 정보는 SAML 2.0 주장과 프로토콜 표준을 참조한다.

7.4.7. 메타데이터 고려사항

특정 프로토콜이나 또는 프로파일에 대한 요청과 응답이 보내져야 하는 URL 엔드포인트들을 가리킴으로써, HTTP Redirect 바인딩을 지원한다는 사실을 반영해야 한다. 단일 엔드포인트 또는 분리된 요청과 응답 엔드포인트들 둘 중 어느 것도 제공될 수 있다.

7.4.8. HTTP Redirect 를 이용하는 SAML 메시지 교환 예

이 예에서 <LogoutRequest>와 <LogoutResponse> 메시지 쌍이 HTTP Redirect 바인딩을 사용하여 교환된다.

우선, 교환되는 실제 SAML 프로토콜 메시지들은 다음과 같다.

```
<samlp:LogoutRequest
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z"
Version="2.0">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
<NameID Format="urn:oasis:names:tc:SAML:2.0:nameidformat:persistent">
005a06e0-ad82-110d-a556-004005b13a2b
</NameID>
<samlp:SessionIndex>1</samlp:SessionIndex>
</samlp:LogoutRequest>
```

```
<samlp:LogoutResponse
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="b0730d21b628110d8b7e004005b13a2b"
InResponseTo="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z" Version="2.0">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
</samlp:LogoutResponse>
```

단계 1 에서 사용자 에이전트로부터 발생하는 초기 HTTP 요청은 이 바인딩에서 정의되지 않는다. 로그아웃 프로토콜 교환을 시작하기 위해, SAML 요청자는 서명된 SAML 요청 메시지를 포함하는 다음과 같은 HTTP 응답을 반환한다. SAMLRequest 파라미터 값은 실질적으로 위의 요청 메시지로부터 도출된다. 서명 부분은 단지 예일뿐이며 실제 계산의 결과는 아니다. 아래 HTTP Location 헤더에 있는 라인피드는 문서의 정렬을 위해 존재하는 것이며 실제 헤더 값에는 라인피드가 존재하지 않는다.

```
HTTP/1.1 302 Object Moved
Date: 21 Jan 2004 07:00:49 GMT
Location:
https://ServiceProvider.com/SAML/SLO/Browser?SAMLRequest=fVFdS8MwFH0f7D%
2BUvGdNsq62oSsIQyhMESc%2B%2BJYlmRbWpObeyvz3puv2IMjyFM7HPedyK1DdsZd
b%2F%
2BEHfLFfgwVMTt3RgTwzazIEJ72CFqRTnQWJWu7uH7dSLJjsg0ev%2FZFMIttiBWADtt6R
%
2BSyJr9msiRH7O70sCm31Mj%2Bo%2BC%
2B1KA5GlEWeZaogSQMw2MYBKodrIhjLKONU8FdeSsZkVr6T5M0GiHMjvWCknqZXZ2Oo
PxF7kG
naGOuwxZ%2Fn4L9bY8NC%
2By4du1XpRXnxPcXizSZ58KFTeHujEWkNPZyIsh9bAMYYUjO2Uiy3jCpTCMo5M1StVjmN
9SO1
50sI9IU6RV2Dp0vsLly7NM7YU82r9B90PrvCf85W%2FwL8zSVQzAEAAA%3D%
3D&RelayState=0043bfc1bc45110dae17004005b13a2b&SigAlg=http%3A%2F%
2Fwww.w3.org%2F200%2F09%2Fxmldsig%23rsasha1&
Signature=NOTAREALSIGNATUREBUTTHEREALONEWOULDGOHERE
Content-Type: text/html; charset=iso-8859-1
```

특별히 명기되지 않는 상호작용들이 일어날 수 있다. 이와 같은 상호작용 후에, SAML 응답자는 서명된 SAML 응답 메시지를 포함하는 아래 HTTP 응답을 반환한다. 다시, SAMLResponse 파라미터 값은 실질적으로 위의 응답 메시지로부터 도출된다. 서명 부분은 단지 예일뿐이며 실제 계산의 결과는 아니다.

```
HTTP/1.1 302 Object Moved
Date: 21 Jan 2004 07:00:49 GMT
Location:
https://IdentityProvider.com/SAML/SLO/Response?SAMLResponse=fVFNa4QwEL0X%
2Bh8k912TaDUGFUp7EbZQ6rKH3mKcbQVNJBOX%2FvxaXQ9tYec0vHlv3nzkqIZ%2BlAf
7YSf%
2FBjhagxB8Db1BuZQKMjkjrcIOpVEDoPRa1o8vB8n3VI7OeqttT1bJbbJCBOc7a8j9XTBH
9Vy
QhqYRbTlrEi4Yo61oUqA0pvShYZHiDQkqs411tAVpeZPqSAgNOkrOas4zzcW55ZII4liJrTX
i
BJVBr4wvCJ877ijbcXZkmaRUxtk7CU7gcB5mLu8pKVddvghd%
2Ben9iDIMa3CXTsOrs5euBbfXdgh%2F9snDK%2FEqW69Ye%2BUnvGL%2F8CfbQnBS
%
2FQS3z4QLW9aT1oBIws0j%2FGOyAb9%2FV34Dw5k779IBAAA%
3D&RelayState=0043bfc1bc45110dae17004005b13a2b&SigAlg=http%3A%2F%
2Fwww.w3.org%2F200%2F09%2Fxmldsig%23rsasha1&
Signature=NOTAREALSIGNATUREBUTTHEREALONEWOULDGOHERE
Content-Type: text/html; charset=iso-8859-1
```

7.5. HTTP POST 바인딩

HTTP POST 바인딩은 SAML 프로토콜 메시지가 HTML 폼 컨트롤(form control)의 base64-인코드된 내용으로 전송될 수 있도록 해 주는 메커니즘을 정의한다.

두 가지 다른 바인딩을 사용하는 단일한 프로토콜 교환에서 요청과 응답 메시지를 전송하기 위해, 이 바인딩은 HTTP Redirect 바인딩(3.4 절 참조)와 HTTP Artifact 바인딩(3.6 절 참조)과 결합될 수 있다 MAY).

7.5.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.5.2. 개요

HTTP POST 바인딩은 SAML 요청자와 응답자가 중개자로 (IETF RFC 2616 에 정의된 것처럼) HTTP 사용자 에이전트를 사용하여 통신하는 경우에 사용될 목적을 가진다. 예를 들어, 만약 통신 당사자들이 직접적인 통신 경로를 공유하지 못하는 경우, 이 바인딩이 필요할 수 있다. 만약 사용자 에이전트가 응답자에게 인증 받아야 할 때처럼, 요청을 이행하기 위해 응답자가 사용자 에이전트와 상호작용을 해야 한다면, 이 바인딩이 또한 필요해 질 수 있다.

일부 HTTP 사용자 에이전트는 프로토콜 교환에 있어 좀 더 능동적인 역할을 수행할 능력을 가질 수 있으며, SOAP 과 reverse SOAP 바인딩과 같이 HTTP 를 사용하는 다른 바인딩을 지원할 수 있다. 이 바인딩은 일반적인 웹 브라우저의 능력을 벗어나는 어떤 것도 가정하지 않는다.

7.5.3. RelayState

RelayState 는 이 바인딩으로 전송되는 SAML 프로토콜 메시지와 함께 포함될 수 있다. 이 값은 길이로는 80 바이트를 넘지 않아야만 하며, 메시지 전송 중에 존재할지도 모르는 다른 어떠한 보호 방법과도 독립적으로 메시지를 생성하는 엔티티에 의해 무결성이 보호되어야 한다. 서명은 공간 제약을 고려하면 실질적인 방안이 되지 못한다. 그러나, 그 값이 제 3 자 변조에 노출되기 때문에, 엔티티는 체크섬(checksum),

의사랜덤(pseudo-random) 값 또는 유사한 방식을 이용하여 그 값이 변조되지 않았음을 보장해야 한다.

만약, SAML 요청 메시지가 RelayState 데이터를 수반한다면, SAML 응답자 또한 RelayState 메커니즘을 지원하는 바인딩을 사용하여 그것의 SAML 프로토콜 응답을 반환해야만 하고, SAML 응답자는 응답 안에 대응되는 RelayState 파라미터를 위치시키고, 그것이 요청에서 수신한 것과 정확히 같은 데이터를 파라미터 값으로 설정해야만 한다.

만약 그러한 값이 SAML 요청 메시지와 함께 포함되지 않는다면, 또는 만약 SAML 응답 메시지가 대응되는 요청 없이 생성되었다면, SAML 응답자는 프로파일의 사용 또는 당사자들 사이의 이전 협정에 따라 수신자에 의해 해석되는 RelayState 데이터를 포함시킬 수 있다.

7.5.4. 메시지 인코딩

메시지들은 XML 을 HTML 폼 컨트롤 안으로 인코딩함으로써 이 바인딩에서 사용되도록 인코딩되고, HTTP POST 방식을 사용하여 전송된다. SAML 프로토콜 메시지는 메시지의 XML 표현(representation)에 base64 인코딩 규칙을 적용하고 그 결과를 W3C HTML 17 장에 정의된 것처럼 폼(form) 내에 은닉(hidden) 폼 컨트롤에 위치시킴으로써 폼-인코드된다. HTML 문서는 W3C XHTML 을 따라야만 한다.

만약 메시지가 SAML 요청이면, 그러면 폼 컨트롤은 SAMLRequest 로 명명되어야만 한다. 만약 메시지가 SAML 응답이면, 그러면 폼 컨트롤은 SAMLResponse 로 명명되어야만 한다. 어떠한 추가적인 폼 컨트롤이나 표현(presentation)도 포함될 수 있으나 수신자가 메시지를 처리하기 위해 요구되지 않아야만 한다.

만약 “RelayState” 값이 SAML 프로토콜 메시지에 수반되면, 그것은 SAML 메시지와 함께 동일한 폼 내의 RelayState 로 명명된 추가적인 은닉 폼 컨트롤에 놓여져야만 한다.

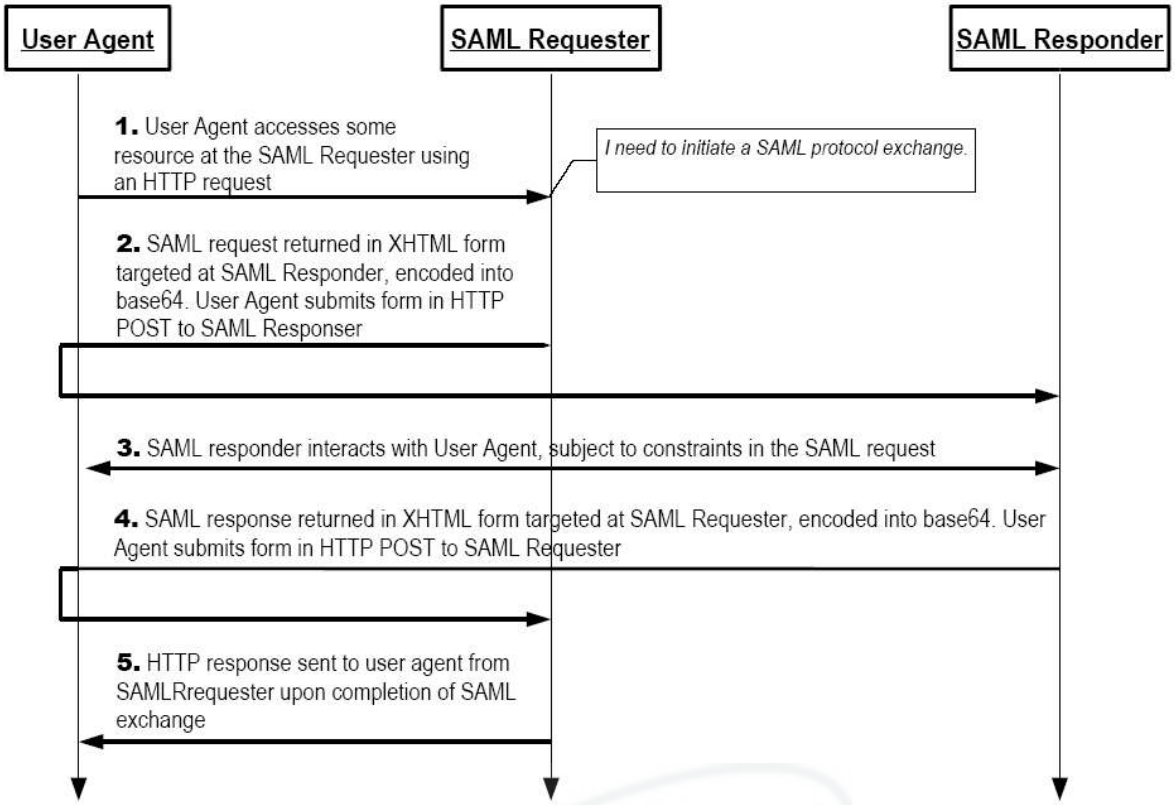
폼의 action 속성은 SAML 메시지가 배달되는, 이 바인딩을 사용하는 프로토콜 또는 프로파일에 대한 수신자의 HTTP 엔드포인트이어야만 한다. method 속성은 “POST”로 설정되어야만 한다.

사용자 에이전트에 의해 지원되는 어떠한 기술도 폼의 전송(submission)을 일으키는데 사용될 수 있고, 전송 컨트롤(submit control)과 클라이언트-측 스크립팅 명령어와 같이, 이것을 지원하는데 필요한 어떠한 폼 내용도 포함될 수 있다. 그렇지만, 수신자는 폼 제출이 어떠한 메커니즘에 의해 이루어졌는지에 관계 없이 메시지를 처리할 수 있어야만 한다.

포함된 어떠한 폼 컨트롤 값들이 안전하게 XHTML 문서에 포함되도록 하기 위해서는 변형되어야만 한다. 이것은 인용부호(quote)와 같은 문자들을 HTML 엔티티등과 같은 것으로 변형하는 것을 포함한다.

7.5.5. 메시지 교환

이 바인딩을 통해 SAML 대화를 위해 사용된 시스템 모델은 요청-응답 모델이다. 그러나 이들 메시지들은 HTTP 응답으로 사용자 에이전트에게 송신되고 HTTP 요청으로 메시지 수신자에게 배달된다. 이들 교환이 발생하기 전, 사이 그리고 이후의 HTTP 상호작용은 명기되지 않는다. SAML 요청자와 SAML 응답자는 둘 다 HTTP 응답자라고 가정한다. 다음 순서도는 교환되는 메시지를 설명한다.



1. 초기에, 사용자 에이전트는 시스템 엔티티에게 임의의 HTTP 서비스를 요청한다. 요청을 처리하는 과정에, 시스템 엔티티는 SAML 프로토콜 교환을 시작할 것을 결정한다.
2. SAML 요청자로 동작하는 시스템 엔티티가 SAML 요청을 반환함으로써 사용자 에이전트로부터 받은 HTTP 요청에 응답한다. 이 요청은 3.5.4 절에서 정의된 폼과 내용을 포함하는 XHTML 문서로 반환된다. 사용자 에이전트는 HTTP POST 요청을 SAML 응답자에게 넘겨줌으로써 SAML 요청을 배달한다.
3. 일반적으로, SAML 응답자는 SAML 응답을 즉시 반환함으로써 SAML 요청에 응답할 수 있거나 또는 요청을 이행하는데 필요한 사용자 에이전트와 일련의 상호작용을 용이하게 하는 임의의 내용을 반환할 수 있다. 특정 프로토콜과 프로파일은 (예를 들어 <samlp:AuthRequest>에서 IsPassive 속성과 같이) 이러한 상호작용을 허용하려는 요청자의 의향 정도(level of willingness)를 가리키는 메커니즘을 포함할 수 있다.

4. 결국, 응답자는 SAML 요청자에게 반환되도록, SAML 응답을 사용자 에이전트에게 반환해야 한다. SAML 응답자는 단계 2 에서의 SAML 요청에 대하여 설명된 것과 동일한 방식으로 반환된다.
5. SAML 응답을 받자마자, SAML 요청자는 사용자 에이전트에게 임의의 HTTP 응답을 반환한다.

7.5.5.1. HTTP 와 캐싱 고려사항

HTTP 프락시들과 사용자 에이전트 중개자는 SAML 프로토콜 메시지들을 캐시하지 않아야 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1 을 사용하여 SAML 프로토콜 메시지를 반환할 때, HTTP 응답자는 다음과 같이 해야 한다:

- "no-cache, no-store"로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- "no-cache"로 설정된 Pragma 헤더 필드를 포함해야만 한다.

HTTP 헤더 사용에 대한 다른 제약사항은 없다.

7.5.5.2. 보안 고려사항

사용자 에이전트 중개자가 존재한다는 것은 요청자와 응답자가 단대단 인증, 무결성과 기밀성을 위해 트랜스포트 레이어를 의지할 수 없고 대신 그들이 수신 받은 메시지들을 인증해야만 한다는 것을 의미한다. SAML 은 이러한 경우에 인증과 무결성을 위해 프로토콜 메시지에 대한 서명을 제공한다. Form-인코드된 메시지는 base64 인코딩이 적용되기 전에 서명될 수 있다.

만약 메시지가 서명되면, 프로토콜 메시지의 루트 SAML 요소에 있는 Destination XML 속성은 송신자가 사용자 에이전트에게 메시지를 그 곳으로 배달하라고 명령하는 URL 을 포함해야만 한다. 그 다음, 수신자는 그 값이 메시지가 수신된 위치와 일치하는지 검증해야만 한다.

만약 요청 또는 응답의 내용이 사용자 에이전트 중개자에게 노출되지 않아야 한다면, 이 바인딩은 사용되지 않아야 한다. 만약 그렇지 않다면, SAML 요청과 SAML 응답 둘 모두의 기밀성은 선택적이며 사용 환경에 따라 결정된다. 만약 기밀성이 필요하다면,

TLS 1.0 이 사용자 에이전트와, SAML 요청자와 응답자 사이에서 전송되는 메시지를 보호하기 위해 사용되어야 한다.

일반적으로, 이 바인딩은 서명을 통해 얻어지는 메시지 수준의 인증과 무결성 보호를 의지하지만, 사용자 에이전트 중개자에게 메시지의 기밀성을 유지하는 기능을 지원하지는 않는다.

또한 SAML 프로토콜 메시지와, 만약 있다면, “RelayState” 값 사이의 관계에 대한 무결성을 보호하기 위한 어떠한 메커니즘도 정의되어 있지 않다. 즉, 공격자는 각각의 SAML 프로토콜 메시지와 관련된 “RelayState” 값들을 교환함으로써 잠재적으로 한 쌍의 유효한 HTTP 응답을 재 조립할 수 있다. 개별적인 “RelayState”와 SAML 메시지 값들은 무결성이 보호될 수 있으나 그들의 조합은 그렇지 않다. 따라서, “RelayState” 정보의 생산자와 소비자는 (SAML 메시지에 있는 정보를 기반으로 하는 것과 같은) 추가적인 조치를 취하지 않고는 민감한 상태 정보를 “RelayState” 값과 연관시키지 않도록 주의를 기울여야만 한다.

7.5.6. 에러 보고

SAML 요청자와 메시지를 교환하는 것을 거부하는 SAML 응답자는 urn:oasis:names:tc:SAML:2.0:status:RequestDenied 를 차상위 수준(second-level) <samlp:StatusCode> 값으로 가지는 SAML 응답 메시지를 반환해야 한다.

메시지 교환 중에 발생하는 HTTP 상호작용들은 SAML 처리시 발생하는 실패를 가리키기 위해 HTTP 에러 상태 코드들을 사용하지 않아야만 한다. 왜냐하면 사용자 에이전트가 SAML 프로토콜 교환에 참여하는 단 하나의 당사자는 아니기 때문이다.

SAML 상태 코드에 대한 더 많은 정보는 SAML 2.0 주장과 프로토콜 표준을 참조한다.

7.5.7. 메타데이터 고려사항

특정 프로토콜이나 또는 프로파일에 대한 요청과 응답이 보내져야 하는 URL 엔드포인트들을 가리킴으로써, HTTP POST 바인딩을 지원한다는 사실을 반영해야 한다.

단일 엔드포인트 또는 분리된 요청과 응답 엔드포인트들 둘 중 어느 것도 제공될 수 있다.

7.5.8. HTTP POST 를 이용하는 SAML 메시지 교환 예

이 예에서 <LogoutRequest>와 <LogoutResponse> 메시지 쌍이 HTTP POST 바인딩을 사용하여 교환된다.

우선, 교환되는 실제 SAML 프로토콜 메시지들은 다음과 같다.

```
<samlp:LogoutRequest
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z"
Version="2.0">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
<NameID Format="urn:oasis:names:tc:SAML:2.0:nameidformat:persistent">
005a06e0-ad82-110d-a556-004005b13a2b
</NameID>
<samlp:SessionIndex>1</samlp:SessionIndex>
</samlp:LogoutRequest>
```

```
<samlp:LogoutResponse
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="b0730d21b628110d8b7e004005b13a2b"
InResponseTo="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z" Version="2.0">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
</samlp:LogoutResponse>
```

단계 1 에서 사용자 에이전트로부터 발생하는 초기 HTTP 요청은 이 바인딩에서 정의되지 않는다. 로그아웃 프로토콜 교환을 시작하기 위해, SAML 요청자는 서명된 SAML 요청 메시지를 포함하는 다음과 같은 HTTP 응답을 반환한다. SAMLRequest 파라미터 값은 실질적으로 위의 요청 메시지로부터 도출된다.

```
HTTP/1.1 200 OK
Date: 21 Jan 2004 07:00:49 GMT
Content-Type: text/html; charset=iso-8859-1

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<body onload="document.forms[0].submit()">

<noscript>
<p>
<strong>Note:</strong> Since your browser does not support JavaScript,
you must press the Continue button once to proceed.
</p>
</noscript>

<form action="https://ServiceProvider.com/SAML/SLO/Browser"
method="post">
<div>
<input type="hidden" name="RelayState"
value="0043bfc1bc45110dae17004005b13a2b"/>
<input type="hidden" name="SAMLRequest"
value="PHNhbWxwOkxvZ291dFJlcXVlc3QgeG1sbnM6c2FtbHA9InVybjpvYXNpczpuYW1l
czp0YzpTQU1MOjIuMDpwcm90b2NvbCIgeG1sbnM9InVybjpvYXNpczpuYW1lczp0
YzpTQU1MOjIuMDphc3NlcnRpb24iDQogICAgSUQ9ImQyYjdjMzg4Y2VjMzZmYTdj
MzIjMjhmZDI5ODY0NGE4IiBJc3N1ZUluc3RhbnQ9IjIwMDQtMDEtMjFUMTk6MDA6
NDIaliBWZXJzaW9uPSIyLjAiPg0KICAgIDxJc3N1ZXI+aHR0cHM6Ly9JZGVudGl0
eVByb3ZpZGVyLmNvbS9TQU1MPC9Jc3N1ZXI+DQogICAgPE5hbWVJRCBGb3JtYXQ9
InVybjpvYXNpczpuYW1lczp0YzpTQU1MOjIuMDpuYW1laWQtZm9ybWF0OnBlcnNp
c3RlbnQiPjAwNWEwNmUwLWFkODItMTEwZC1hNTU2LTAwNDAwNWIxM2EyYjwvTmFt
ZUlEPg0KICAgIDxzYW1scDpTZXNzaW9uSW5kZXg+MTwvc2FtbHA6U2Vzc2lvbklu
ZGV4Pg0KPC9zYW1scDpMb2dvdXRSZXF1ZXN0Pg=="/>
</div>
<noscript>
<div>
<input type="submit" value="Continue"/>
</div>
</noscript>
</form>
</body>
</html>
```

특별히 명기되지 않는 상호작용들이 일어날 수 있다. 이와 같은 상호작용 후에, SAML 응답자는 SAML 응답 메시지를 포함하는 아래 HTTP 응답을 반환한다. 다시, SAMLResponse 파라미터 값은 실질적으로 위의 응답 메시지로부터 도출된다.

```
HTTP/1.1 200 OK
Date: 21 Jan 2004 07:00:49 GMT
Content-Type: text/html; charset=iso-8859-1
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<body onload="document.forms[0].submit()">
<noscript>
<p>
<strong>Note:</strong> Since your browser does not support JavaScript,
you must press the Continue button once to proceed.
</p>
</noscript>
<form action="https://IdentityProvider.com/SAML/SLO/Response"
method="post">
<div>
<input type="hidden" name="RelayState"
value="0043bfc1bc45110dae17004005b13a2b"/>
<input type="hidden" name="SAMLResponse"
value="PHNhbWxwOkxvZ291dFJlc3BvbnNlIHhtbG5zOnNhbWxwPSJ1cm46b2FzaXM6bmFt
ZXM6dGM6U0FNTDoyLjA6cHJvdG9jb2wiIHhtbG5zPSJ1cm46b2FzaXM6bmFtZXM6
dGM6U0FNTDoyLjA6YXNzZXJ0aW9uIg0KICAgIElEPSJiMDczMGQyMWI2MjgxMTBk
OGI3ZTAwNDAwNWIxM2EyYiIgSW5SZXNwb25zZVRvPSJkMmI3YzM4OGNlYzM2ZmE3
YzM5YzI4ZmQyOTg2NDRhOCINCiAgICBJc3N1ZUluc3RhbnQ9IjIwMDQtMDEtMjFU
MTk6MDA6NDlaIiBWZXJzaW9uPSIyLjAiPg0KICAgIDxJc3N1ZXI+aHR0cHM6Ly9T
ZXJ2aWNlUHJvdmlkZXIuY29tL1NBTUw8L0Izc3VlcjxNCiAgICA8c2FtbHA6U3Rh
dHVzPg0KICAgICAgICA8c2FtbHA6U3RhdHVzQ29kZSBWYWx1ZT0idXJuOm9hc2Iz
Om5hbWVzOnRjOlNBTUw6Mi4wOnN0YXR1czpTdWNjZXNzIi8+DQogICAgPC9zYW1s
cDpTdGF0dXM+DQo8L3NhbWxwOkxvZ291dFJlc3BvbnNlPg=="/>
</div>
<noscript>
<div>
<input type="submit" value="Continue"/>
</div>
</noscript>
</form>
</body>
</html>
```

7.6. HTTP Artifact 바인딩

HTTP Artifact 바인딩에서는, artifact 로 불리는 작은 대용물을 사용하는 참조 방식으로 SAML 요청, SAML 응답 또는 둘 모두가 전달된다. SAML 2.0 주장과 프로토콜 표준에서 정의된 artifact resolution protocol 을 사용하여 실제 프로토콜 메시지를 위한 artifact 를 교환하는데, SAML SOAP 바인딩과 같은 분리되고 동기적인 바인딩이 사용된다.

두 가지 다른 바인딩을 사용하는 단일한 프로토콜 교환에서 요청과 응답 메시지를 전송하기 위해, 이 바인딩은 HTTP Redirect 바인딩(3.4 절 참조)과 HTTP POST 바인딩(3.5 절 참조)에 결합될 수 있다.

7.6.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.6.2. 개요

HTTP Artifact 바인딩은 SAML 요청자와 응답자가 중개자로 HTTP 사용자 에이전트를 사용하여 통신하지만, 중개자의 한계로 인해 중개자를 통해 전체 메시지를 전달하지 못하거나 또는 메시지 교환이 배제되는 경우에 사용될 목적을 가진다. 이것은 기술적인 이유 때문에 발생할 수도 있으며 또는 메시지 내용이 중개자에게 노출되는 것을 싫어하기 때문에 발생할 수 있다.

SOAP 과 같은 또 다른 동기 바인딩을 사용하여 artifact 를 해결할 필요가 있기 때문에, SAML 메시지 송신자와 수신자 사이에는 artifact 전달 방향과는 정반대 방향의 직접적인 통신 경로가 존재해야만 한다. 메시지와 artifact 의 수신자는 artifact 발급자에게 <samlp:ArtifactResolve> 요청을 보낼 수 있어야만 한다. artifact 발급자는 또한

artifact 를 해결되지 않고 있는 동안에는 상태 관리를 해야만 하고, 이것은 부하-분산 환경(load-balanced environments)에 영향을 미친다.

7.6.3. 메시지 인코딩

두 가지 artifact 인코딩 방식이 이 바인딩에서 사용된다. 하나는 artifact 를 URL 파라미터로 인코드하는 것이고, 다른 하나는 artifact 를 HTML 폼 컨트롤에 위치시키는 것이다. URL 인코딩이 사용될 때, HTTP GET 방식이 메시지를 배달하는데 사용되며, 반면에 POST 방식은 폼 인코딩에서 사용된다. 이 바인딩을 지원하는 모든 엔드포인트들은 두 가지 기술 모두를 지원해야만 한다.

7.6.3.1. RelayState

RelayState 는 이 바인딩으로 전송되는 SAML 프로토콜 메시지와 함께 포함될 수 있다. 이 값은 길이로는 80 바이트를 넘지 않아야만 하며, 메시지 전송 중에 존재할지도 모르는 다른 어떠한 보호 방법과도 독립적으로 메시지를 생성하는 엔티티에 의해 무결성이 보호되어야 한다. 서명은 공간 제약을 고려하면 실질적인 방안이 되지 못한다. 그러나, 그 값이 제 3 자 변조에 노출되기 때문에, 엔티티는 체크섬(checksum), 의사랜덤(pseudo-random) 값 또는 유사한 방식을 이용하여 그 값이 변조되지 않았음을 보장해야 한다.

만약, SAML 요청을 나타내는 artifact 가 RelayState 데이터를 수반한다면, SAML 응답자 또한 RelayState 메커니즘을 지원하는 바인딩을 사용하여 그것의 SAML 프로토콜 응답을 반환해야만 하고, SAML 응답자는 응답 안에 대응되는 RelayState 파라미터를 위치시키고, 그것이 요청에서 수신한 것과 정확히 같은 데이터를 파라미터 값으로 설정해야만 한다.

만약 그러한 값이 SAML 요청을 나타내는 artifact 와 함께 포함되지 않는다면, 또는 만약 SAML 응답 메시지가 대응되는 요청 없이 생성되었다면, SAML 응답자는

프로파일의 사용 또는 당사자들 사이의 이전 협정에 따라 수신자에 의해 해석되는 RelayState 데이터를 포함시킬 수 있다.

7.6.3.2. URL 인코딩

artifact 를 URL 에 인코드하기 위해, artifact 값은 URL-인코드되고 SAMLart 라고 명명된 질의 문자열 파라미터에 놓인다.

만약, “RelayState” 값이 SAML artifact 에 수반되면, 그 값은 URL-인코드되고 RelayState 라고 명명된 추가적인 질의 문자열 파라미터에 놓인다.

7.6.3.3. 폼 인코딩

SAML artifact 는 W3C HTML 에서 정의된 것처럼 폼 내에서 은닉 폼 컨트롤에 그것을 놓음으로써 폼-인코드된다. HTML 문서는 W3C XHTML 표준을 따라야만 한다. 폼 컨트롤은 SAMLArt 로 명명되어야만 한다. 어떠한 추가적인 폼 컨트롤이나 표현(presentation)도 포함될 수 있으나, 수신자가 artifact 를 처리하기 위해 요구되지 않아야만 한다.

만약 “RelayState” 값이 SAML artifact 에 수반되면, 그것은 SAML 메시지와 함께 동일한 폼 내의 RelayState 로 명명된 추가적인 은닉 폼 컨트롤에 놓여져야만 한다.

폼의 action 속성은 artifact 가 배달되는, 이 바인딩을 사용하는 프로토콜 또는 프로파일에 대한 수신자의 HTTP 엔드포인트이어야만 한다. method 속성은 “POST”로 설정되어야만 한다.

사용자 에이전트에 의해 지원되는 어떠한 기술도 폼의 전송(submission)을 일으키는데 사용될 수 있고, 전송 컨트롤(submit control)과 클라이언트-측 스크립팅 명령어와 같이, 이것을 지원하는데 필요한 어떠한 폼 내용도 포함될 수 있다. 그렇지만, 수신자는 폼 제출이 어떠한 메커니즘에 의해 이루어졌는지에 관계 없이 메시지를 처리할 수 있어야만 한다.

포함된 어떠한 폼 컨트롤 값들이 안전하게 XHTML 문서에 포함되도록 하기 위해서는 변형되어야만 한다. 이것은 인용부호(quote)와 같은 문자들을 HTML 엔티티등과 같은 것으로 변형하는 것을 포함한다.

7.6.4. Artifact 포맷

이 바인딩에서 artifact는 짧고 불투명한 문자열이다. 이 바인딩에 영향을 주지 않으며, 다른 타입들이 정의되어 사용될 수 있다. artifact 수신자가 artifact의 발급자를 식별할 수 있고 변조나 위조를 억제하며, 유일하고 간결함(compactness)을 가지는 것이 중요한 특징이다.

어떠한 artifact 포맷에서도 다음과 같이 필수적인(mandatory) 2-바이트 타입 코드와 발급자의 artifact 해결 서비스의 특정한 엔드포인트를 식별하는 2-바이트 인덱스 값을 포함하는 것이 일반적이다.

```
SAML_artifact     := B64(TypeCode EndpointIndex RemainingArtifact)
  TypeCode        := Byte1Byte2
  EndpointIndex   := Byte1Byte2.
```

표기법 B64(TypeCode EndpointIndex RemainingArtifact)는 TypeCode, EndpointIndex와 RemainingArtifact의 연결에 대하여 base64(IETF RFC 2045 참조) 변형을 적용하는 것을 나타낸다.

SAML artifact를 생성할 때 다음 실행들이 권고된다.

- 각각의 발급자는 발급자의 엔티티 (또는 제공자) ID로도 알려진 식별하는 URI를 할당받는다. 이런 종류의 식별자에 대한 검토는 SAML 2.0 주장과 프로토콜 표준을 참조한다.
- 발급자는 식별 URL에 SHA-1 해시를 수행하여 artifact의 SourceID 컴포넌트를 생성한다. 해시 값은 16진수로 인코드되지 않는다.
- MessageHandle 값은 발급자에 의해 생성된 암호학적으로 강한 랜덤 또는 의사랜덤 수열(number sequence)로부터 만들어진다. 수열은 크기가 최소한 16 바이트인

값들로 구성된다. 이들 값들은 필요하면 전체 길이가 20 바이트까지 되도록 패드되어야(padded) 한다.

다음은 SAML 2.0 에서 정의된 단 하나의 artifact 타입을 설명한다.

7.6.4.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:artifact-04

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.6.4.2. Format 세부 사항

SAML 2.0 은 타입 코드(type code) 0x0004 를 가지는 artifact 타입을 정의한다. 이 artifact 타입은 다음과 같이 정의된다.

```
TypeCode := 0x0004
RemainingArtifact := SourceID MessageHandle
SourceID := 20-byte_sequence
MessageHandle := 20-byte_sequence
```

SourceID 는 artifact 발급자 아이덴터티과 가능한 해결 엔드포인트들의 집합을 결정하기 위해 artifact 수신자에 의해 사용되는 20 바이트 열이다.

목적 사이트는 대응되는 SAML 응답자에 대한 하나 또는 그 이상으로 색인된(indexed) URL 엔드포인트들(또는 주소들)뿐만 아니라 SourceID 값들의 테이블을 유지할 것이라고 가정한다. SAML 2.0 메타데이터 표준은 이러한 목적으로 사용될 수 있다. SAML artifact 를 받자마자, 수신자는 SourceID 가 알려진 artifact 발급자에 속하는지 아닌지를 결정하고 SAML <samlp:ArtifactResolve> 메시지를 SAML 응답자에 보내기 전에 EndpointIndex 를 이용하여 SAML 응답자의 위치를 얻는다.

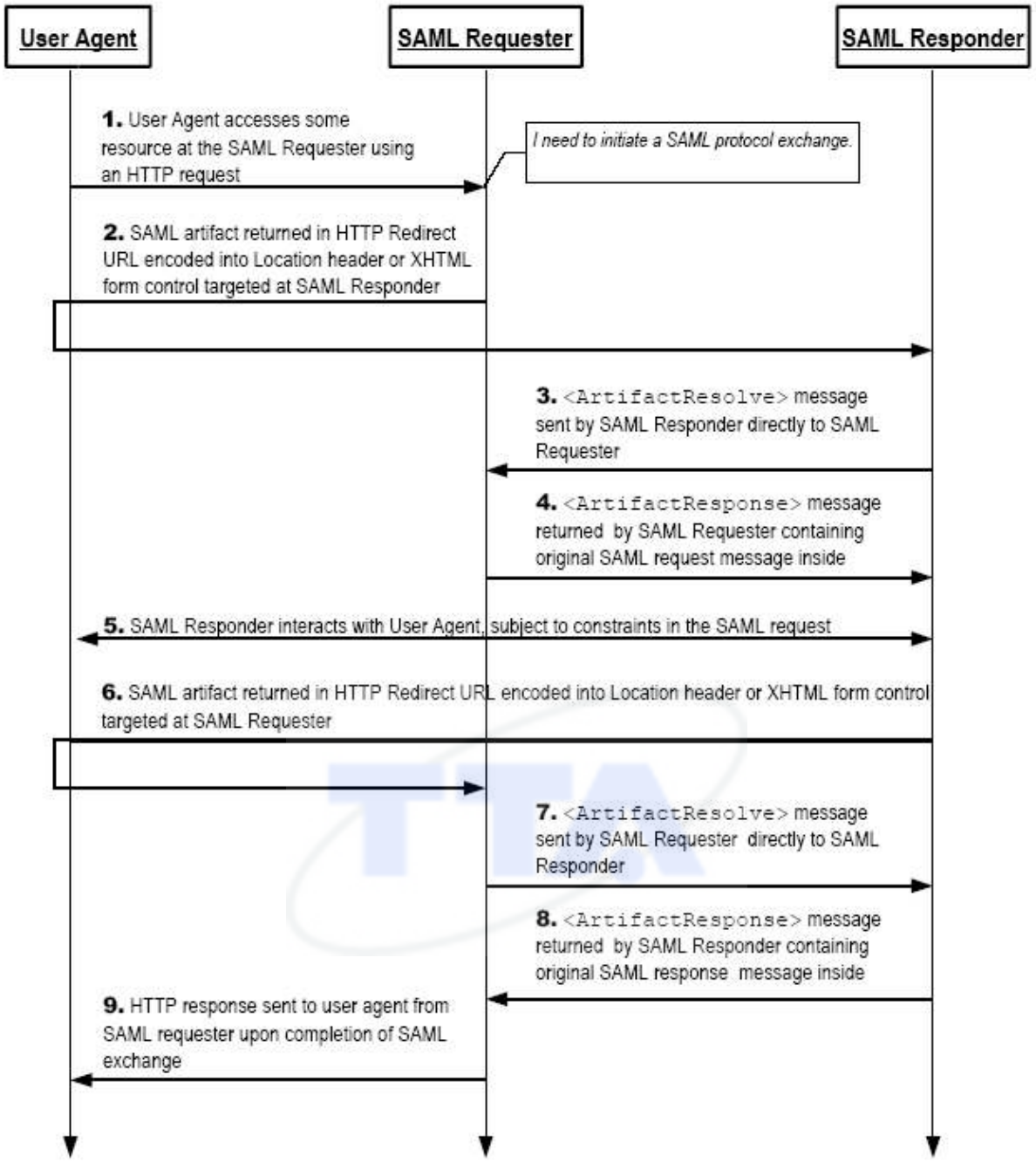
공통의 수신자를 가지는 임의의 두 artifact 발급자들은 반드시 서로 다른 SourceID 값들을 가져야만 한다. MessageHandle 값들을 만드는 것은, 그들이 발급 사이트에서 참조되는 메시지의 내용과 예상할 수 있는 어떠한 관계도 가지지 않아야 한다는 원칙과 유효하며 미해결된 메시지 핸들의 값을 만들거나 또는 추측하는 것이 불가능해야만 한다는 원칙에 따라 이루어진다.

7.6.5. 메시지 교환

이 바인딩을 이용하는 SAML 대화를 위해 사용된 시스템 모델은 요청-응답 모델이다. 이 모델에서는 artifact 참조가 실제 메시지 내용을 대신하며, artifact 참조가 HTTP 응답으로 사용자 에이전트에게 보내지고, HTTP 요청으로 메시지 수신자에게 배달된다. 이들 교환이 발생하기 전, 사이 그리고 이후의 HTTP 상호작용은 명기되지 않는다. SAML 요청자와 SAML 응답자는 둘 다 HTTP 응답자라고 가정한다.

추가적으로, 사용자 에이전트를 경유하여 artifact 를 수신할 때, 수신자는 SAML 2.0 주장과 프로토콜 표준에서 정의된 Artifact Resolution 프로토콜을 사용하여 artifact 발급자와 별도의 직접적인 교환을 시작한다. 이 교환은 SOAP 바인딩처럼 중개자로 HTTP 사용자 에이전트를 사용하지 않는 바인딩을 사용해야만 한다. SAML 프로토콜 메시지를 얻는데 성공하면, artifact 는 폐기되고 주요한 SAML 프로토콜 교환의 처리가 재개된다. 만약 얻어진 메시지가 응답이라면, 메시지 교환은 종료된다.

뒤따르는 해결 단계와 함께, artifact 를 발급하고 배달하는 것은 전체 SAML 프로토콜 교환의 절반을 구성한다. 이 바인딩은 하나의 SAML 프로토콜 교환의 한쪽 또는 두 쪽 모두를 배달하는데 사용될 수 있다. HTTP Redirect 바인딩(3.4 절 참조) 또는 HTTP POST 바인딩(3.5 절 참조)과 같이 이 바인딩과 조합될 수 있는 바인딩은 이 교환의 다른 반쪽을 운반하는데 사용될 수 있다. 다음 순서는 artifact 바인딩이 교환의 양쪽 부분 모두에서 사용된다. 교환되는 메시지들을 설명하는 아래 그림을 참조한다.



1. 초기에, 사용자 에이전트는 시스템 엔티티에게 임의의 HTTP 서비스를 요청한다. 요청을 처리하는 과정에, 시스템 엔티티는 SAML 프로토콜 교환을 시작할 것을 결정한다.
2. SAML 요청자로 동작하는 시스템 엔티티는 SAML 요청을 나타내는 artifact 를 반환함으로써 사용자 에이전트로부터 요청된 HTTP 요청에 응답한다.

- 만약 URL-인코드된다면, artifact 는 HTTP 응답의 Location 헤더 안에 인코드되어 반환되고, 그리고 HTTP 상태는 303 또는 302 둘 중에 하나이어야만 한다. SAML

요청자는 IETF RFC 2616 에서 정의된 것처럼, 사용자 에이전트의 메시지 송신을 용이하게 하기 위해 HTTP 응답에 추가적인 표현(presentation)과 내용을 포함할 수 있다. 사용자 에이전트는 SAML 응답자에게 HTTP GET 요청을 넘겨줌으로써 artifact 를 배달한다.

- 만약 폼-인코드된다면, artifact 는 3.6.3.3 절에서 정의된 폼과 내용을 포함하는 XHTML 문서에 포함되어 반환된다. 사용자 에이전트는 SAML 응답자에게 HTTP POST 요청을 넘겨줌으로써 artifact 를 배달한다.

3. SAML 응답자는 artifact 를 검사함으로써 SAML 요청자를 결정하고 (정확한 처리는 artifact 의 타입을 의지한다), 그리고 일시적으로 역할을 반대로 하며, 직접적인 SAML 바인딩을 이용하여 artifact 를 포함하는 <samlp:ArtifactResolve> 요청을 SAML 요청자에게 전달한다.
4. 필요한 조건들이 충족되어진다고 가정하면, SAML 요청자는 SAML 응답자가 처리하기를 원하는 원래의 SAML 요청 메시지를 포함하는 <samlp:ArtifactResponse>를 반환한다.
5. 일반적으로, SAML 응답자는 SAML artifact 를 즉시 반환함으로써 SAML 요청에 응답할 수 있거나 또는 요청을 이행하기 위해 필요한 사용자 에이전트와 일련의 상호작용을 용이하게 하기 위한 임의의 내용을 반환할 수 있다. 특정 프로토콜과 프로파일은 (예를 들어 <samlp:AuthRequest>에서 IsPassive 속성과 같이) 이러한 상호작용을 허용하려는 요청자의 의향 정도(level of willingness)를 가리키는 메커니즘을 포함할 수 있다.
6. 결국, 응답자는 SAML 요청자에게 반환되도록, SAML artifact 를 사용자 에이전트에게 반환해야 한다. SAML 응답 artifact 는 단계 2 에서의 SAML 요청 artifact 에 대하여 설명된 것과 동일한 방식으로 반환된다. SAML 요청자는 artifact 를 검사함으로써 SAML 응답자를 결정하고, 그리고 단계 3 에서와 같이 직접적인 SAML 바인딩을 사용하여 SAML 응답자에게 artifact 를 포함한 <samlp:ArtifactResolve> 요청을 전달한다.

7. 필요한 조건들이 충족된다고 가정하면, SAML 응답자는 단계 4 에서와 같이 SAML 요청자가 처리하기를 바라는 SAML 응답 메시지를 포함하는 <samlp:ArtifactResponse>를 반환한다
8. SAML 응답을 받자마자, SAML 요청자는 사용자 에이전트에게 임의의 HTTP 응답을 반환한다.

7.6.5.1. HTTP 와 캐싱 고려사항

HTTP 프락시들과 사용자 에이전트 중개자는 SAML 프로토콜 메시지들을 캐시하지 않아야 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1 을 사용하여 SAML 프로토콜 메시지를 반환할 때, HTTP 응답자는 다음과 같이 해야 한다:

- "no-cache, no-store"로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- "no-cache"로 설정된 Pragma 헤더 필드를 포함해야만 한다.

HTTP 헤더 사용에 대한 다른 제약사항은 없다.

7.6.5.2. 보안 고려사항

이 바인딩은 실제 메시지를 반환하기 위해 직접적인 교환을 바로 뒤에 수반하는, 메시지 참조의 간접적인 전달 조합을 사용한다. 따라서 메시지 참조(artifact)는 스스로 인증되거나 또는 무결성이 보호될 필요가 없다. 그러나 실질 메시지를 반환하는 콜백(callback) 요청/응답 교환은 사용 환경에 따라, 상호 인증되고 무결성이 보호될 수 있다.

만약 실제 메시지가 특정 수신자에게 의도된다면, 그러면 artifact 의 발급자는 실제 메시지를 반환하기 전에, 수반되는 <samlp:ArtifactResolve> 메시지의 송신자를 반드시 인증해야만 한다.

사용자 에이전트로부터 또는 사용자 에이전트로 전달되는 artifact 는 기밀성 있게 보호되어야 한다; 또는 TLS 1.0 이 사용되어야 한다. 실제 메시지를 반환하는 콜백 요청/응답 교환은 사용 환경에 따라 보호될 수 있다.

일반적으로, 이 바인딩은 위조하기 어렵고 짧은 기간 참조로써 artifact 를 의지하며, 실제 메시지를 반환하는 콜백 요청/응답에는 다른 보안 조치들을 적용한다. 모든 artifact 들은 artifact 발급자에 의해 집행되는 단일-사용 의미를 가져야만 한다.

더욱이, 공격자가 사용자 에이전트에 의한 artifact 의 해결(resolution)을 간섭하고 artifact 를 artifact 수신자에게 재 송신하는 것을 막기 위해, artifact 수신자들 또한 그들이 수신한 artifact 값들에 대하여 단일 사용 의미를 시행하도록 하는 것이 권고된다. 만약 artifact 를 해결하려는 시도가 성공적으로 완료되지 못하면, artifact 는 합리적인 허용 기간(reasonable acceptance period)을 넘는 시간 동안 차단된(blocked) artifact 리스트에 놓여져야 한다. 합리적인 허용 기간은 artifact 발급자가 artifact 를 해결할 것으로 생각되는 기간이다.

또한 artifact 와, 만약 있다면, “RelayState” 값 사이의 관계에 대한 무결성을 보호하기 위한 어떠한 메커니즘도 정의되어 있지 않다. 즉, 공격자는 각각의 artifact 와 관련된 “RelayState” 값들을 교환함으로써 잠재적으로 한 쌍의 유효한 HTTP 응답을 재 조립할 수 있다. 따라서, “RelayState” 정보의 생산자와 소비자는 (artifact 를 통해 검색되는 SAML 메시지에 있는 정보를 기반으로 하는 것과 같은) 추가적인 조치를 취하지 않고는 민감한 상태 정보를 “RelayState” 값과 연관시키지 않도록 주의를 기울여야만 한다.

7.6.6. 에러 보고

SAML 요청자와 메시지를 교환하는 것을 거부하는 SAML 응답자는 urn:oasis:names:tc:SAML:2.0:status:RequestDenied 를 차상위 수준(second-level) <samlp:StatusCode> 값으로 가지는 SAML 응답 메시지를 반환해야 한다.

메시지 교환 중에 발생하는 HTTP 상호작용들은 SAML 처리시 발생하는 실패를 가리키기 위해 HTTP 에러 상태 코드들을 사용하지 않아야만 한다. 왜냐하면 사용자 에이전트가 SAML 프로토콜 교환에 참여하는 단 하나의 당사자는 아니기 때문이다.

만약, artifact 의 발급자가 그것이 이해할 수 있는 <samlp:ArtifactResolve> 메시지를 수신하면, (예를 들어, artifact 요청자가 메시지를 수신 받을 권한이 없거나 또는 artifact 가 더 이상 유효하지 않기 때문에) 비록 그것이 대응되는 메시지를 반환하지 않더라도, artifact 발급자는 urn:oasis:names:tc:SAML:2.0:status:Success 를 <samlp:StatusCode> 값으로 가지는 <samlp:ArtifactResponse>을 반환해야만 한다.

7.6.7. 메타데이터 고려사항

특정 프로토콜이나 또는 프로파일에 대한 요청과 응답이 보내져야 하는 URL 엔드포인트들을 가리킴으로써, HTTP Artifact 바인딩을 지원한다는 사실을 반영해야 한다. 단일한 엔드포인트, 또는 서로 분리된 요청과 응답 엔드포인트들이 제공될 수 있다. <samlp:ArtifactResolve> 메시지들을 처리하기 위하여 하나 또는 그 이상의 색인된 엔드포인트들이 또한 설명되어야 한다.

7.6.8. HTTP Artifact 를 이용하는 SAML 메시지 교환 예

이 예에서 <LogoutRequest>와 <LogoutResponse> 메시지 쌍이 HTTP Artifact 바인딩을 사용하여 교환된다.

우선, 교환되는 실제 SAML 프로토콜 메시지들은 다음과 같다.

```
<samlp:LogoutRequest
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z"
Version="2.0">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
<NameID Format="urn:oasis:names:tc:SAML:2.0:nameidformat:persistent">
005a06e0-ad82-110d-a556-004005b13a2b
</NameID>
<samlp:SessionIndex>1</samlp:SessionIndex>
</samlp:LogoutRequest>
```

```
<samlp:LogoutResponse
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="b0730d21b628110d8b7e004005b13a2b"
InResponseTo="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z" Version="2.0">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
</samlp:LogoutResponse>
```

단계 1 에서 사용자 에이전트로부터 발생하는 초기 HTTP 요청은 이 바인딩에서 정의되지 않는다. 로그아웃 프로토콜 교환을 시작하기 위해, SAML 요청자는 SAML artifact 를 포함하는 다음과 같은 HTTP 응답을 반환한다. 아래 HTTP Location 헤더에 있는 라인피드는 문서의 정렬을 위해 존재하는 것이며 실제 헤더 값에는 라인피드가 존재하지 않는다.

```
HTTP/1.1 302 Object Moved
Date: 21 Jan 2004 07:00:49 GMT
Location:
https://ServiceProvider.com/SAML/SLO/Browser?SAMLart=AAQAADWNEw5VT47wcO4
z
X%2FiEzMmFQvGknDfws2ZtqSGdkNSbsW1cmVR0bzU%
3D&RelayState=0043bfc1bc45110dae17004005b13a2b
Content-Type: text/html; charset=iso-8859-1
```

그 다음, SAML 응답자는 다음과 같이, 단계 3 과 4 에서 Artifact Resolution 프로토콜과 SOAP 바인딩을 이용하여 그것이 수신한 artifact 를 실제 SAML 요청으로 해결한다.

단계 3:

```
POST /SAML/Artifact/Resolve HTTP/1.1
Host: IdentityProvider.com
Content-Type: text/xml
Content-Length: nnn
SOAPAction: http://www.oasis-open.org/committees/security
```

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:ArtifactResolve
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="_6c3a4f8b9c2d"
Version="2.0"
IssueInstant="2004-01-21T19:00:49Z">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
<Artifact>

AAQAADWNEw5VT47wcO4zX/iEzMmFQvGknDfws2ZtqSGdkNSbsW1cmVR0bzU=
</Artifact>
</samlp:ArtifactResolve>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

단계 4:

```
HTTP/1.1 200 OK
Date: 21 Jan 2004 07:00:49 GMT
Content-Type: text/xml
Content-Length: nnnn

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:ArtifactResponse
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="_FQvGknDfws2Z"
Version="2.0"
InResponseTo="_6c3a4f8b9c2d"
IssueInstant="2004-01-21T19:00:49Z">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
```

```
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
<samlp:LogoutRequest
ID="d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:00:49Z"
Version="2.0">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
<NameID

Format="urn:oasis:names:tc:SAML:2.0:nameidformat:persistent">
005a06e0-ad82-110d-a556-004005b13a2b
</NameID>
<samlp:SessionIndex>1</samlp:SessionIndex>
</samlp:LogoutRequest>
</samlp:ArtifactResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

특별히 명기되지 않는 상호작용들이 일어날 수 있다. 이와 같은 상호작용 후에, SAML 응답자는 단계 6 에서 HTTP 응답 안에 두 번째 SAML artifact 를 반환한다.

```
HTTP/1.1 302 Object Moved
Date: 21 Jan 2004 07:05:49 GMT
Location:
https://IdentityProvider.com/SAML/SLO/Response?SAMLart=AAQAAFGIZXv5%
2BQaBaE5qYurHWJO1nAgLAsqfnyiDHIggbFU0mISGFTyQiPc%
3D&RelayState=0043bfc1bc45110dae17004005b13a2b
Content-Type: text/html; charset=iso-8859-1
```

그 다음, SAML 응답자는 다음과 같이, 단계 7 과 8 에서 Artifact Resolution 프로토콜과 SOAP 바인딩을 이용하여 그것이 수신한 artifact 를 실제 SAML 응답으로 해결한다.

단계 7:

```
POST /SAML/Artifact/Resolve HTTP/1.1
Host: ServiceProvider.com
```

```
Content-Type: text/xml
Content-Length: nnn
SOAPAction: http://www.oasis-open.org/committees/security
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:ArtifactResolve
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="_ec36fa7c39"
Version="2.0"
IssueInstant="2004-01-21T19:05:49Z">
<Issuer>https://IdentityProvider.com/SAML</Issuer>
<Artifact>
AAQAAFGIZXv5+QaBaE5qYurHWJO1nAgLAsqfnyiDHIggbFU0mlSGFTyQiPc=
</Artifact>
</samlp:ArtifactResolve>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

단계 8:

```
HTTP/1.1 200 OK
Date: 21 Jan 2004 07:05:49 GMT
Content-Type: text/xml
Content-Length: nnnn

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<samlp:ArtifactResponse
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
ID="_FQvGknDfws2Z"
Version="2.0"
InResponseTo="_ec36fa7c39"
IssueInstant="2004-01-21T19:05:49Z">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
```

```
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
<samlp:LogoutResponse
ID="_b0730d21b628110d8b7e004005b13a2b"
InResponseTo="_d2b7c388cec36fa7c39c28fd298644a8"
IssueInstant="2004-01-21T19:05:49Z"
Version="2.0">
<Issuer>https://ServiceProvider.com/SAML</Issuer>
<samlp:Status>
<samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>
</samlp:LogoutResponse>
</samlp:ArtifactResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.7. SAML URI 바인딩

URI 는 프로토콜에 독립적으로 자원을 참조하는 수단이다. 이 바인딩은 일반적인 SAML 요청/응답 바인딩은 아니지만, 단일한 <saml:AssertionIDRef>를 가진 <samlp:AssertionIDRequest> 메시지를 URI 의 해결(resolution)으로 캡슐화(encapsulation)하는 것을 지원한다. 성공적인 요청의 결과는 (완전한 SAML 응답은 아니지만) 하나의 SAML <saml:Assertion>이 된다.

SOAP 과 같이, URI 해결은 다중 하부 트랜스포트 상에서 일어날 수 있다. 이 바인딩은 트랜스포트에 독립적인 측면을 가지고 있지만 또한 TLS 1.0 을 가지고 HTTP 를 사용할 것을 필수적(반드시 구현해야 되는 것)으로 선언한다.

7.7.1. 필요 정보

Identification: urn:oasis:names:tc:SAML:2.0:bindings:URI

Contact information: security-services-comment@lists.oasis-open.org

Description: 아래에 주어짐.

Updates: 없음.

7.7.2. SAML URI 바인딩의 프로토콜 독립적인 측면

다음 절은 URI 해결 절차에 있어 기반 트랜스포트 프로토콜과 독립적인 SAML URI 바인딩의 측면을 정의한다.

SAML URI 참조는 특정 SAML 주장을 식별한다. URI를 해결하는 것의 결과는 반드시 주장을 포함하는 메시지이거나 또는 트랜스포트에 특정한 에러가 되어야만 한다. 메시지의 특정한 포맷은 하부 트랜스포트 프로토콜에 따라 결정된다. 만약 트랜스포트 프로토콜이 HTTP 1.1 처럼 반환되는 내용이 설명되는 것을 허용한다면, 주장은 허용되는 어떠한 포맷으로도 인코드될 수 있다. 만약 그렇지 않다면, 주장은 모호하지 않게 주장을 XML 직렬화시킨 것으로 해석될 수 있거나 또는 변형될 수 있는 형태로 반환되어야만 한다.

만약 동일한 URI 참조가 미래에 해결된다면, 그러면 동일한 SAML 주장이 반환되거나 또는 에러가 반환되어야만 한다. 즉, 참조는 영속적일 수 있지만 반드시 만약 있다면, 일관되게 동일한 주장을 참조해야만 한다.

7.7.3. 보안 고려사항

만약 참조를 결과로 바인딩하는 것이 안전하지 않다면, SAML 주장을 간접적으로 사용하는 것은 위험하게 된다. 특별한 위협들과 그들의 엄중성(severity)은 주장의 사용에 따라 결정된다. 일반적으로, 단지 요청자가 응답자의 아이덴터티와 SAML 주장의 내용이 전송 중에 변경되지 않았다는 것을 확신할 수 있을 때만, URI 참조를 SAML 주장으로 해결하는 것의 결과가 신뢰되어야 한다.

주장 그 자체가 서명되는 것만으로는 충분하지 못한 경우가 자주 있다. 왜냐하면 URI 참조는 본질적으로 요청자에게 어느 정도 불투명하기 때문이다. 요청자는 반환된 주장이 실질적으로 URI 가 나타내는 주장과 일치하는지를 보장할 수 있는 독립적인 수단을 가지고 있어야 한다; 즉, 이것은 응답자를 인증하는 것과 응답의 무결성을 믿는 것 두 가지 모두를 동반한다.

7.7.4. MIME 인캡슐화

내용 설명과 패키징 메커니즘으로써 MIME 을 지원하는 해결 프로토콜들에 대해, 이 바인딩 결과로 제공되는 주장은 application/samlassertion+xml MIME 엔티티 타입으로써 반환되어야 한다.

7.7.5. HTTP URI 의 사용

SAML URI 바인딩을 준용한다고 주장하는 SAML 기관은 HTTP 에 대한 지원을 구현하여야만 한다. 이 절은 URI 문법, HTTP 헤더들, 그리고 에러 보고를 포함하여, HTTP URI 들을 사용하는 것의 특징에 대하여 설명한다.

7.7.5.1. URI 문법

일반적으로, 참조를 책임지는 SAML 기관이 그것을 포함하는 메시지를 생성하는 한, SAML URI 참조의 허용가능한 문법에는 어떠한 제약도 없다. 그렇지만, 기관들은 ID 로

명명된 단일한 질의 문자열 파라미터를 가진 HTTP 요청이 보내져야 되는 URL 엔드포인트를 지원해야만 한다. 엔드포인트 URL 그 자체에 이 파라미터와 관계없는 질의 문자열이 있지 않도록 해야만 한다. 예를 들어, 만약 기관에 문서화된 엔드포인트가 “https://saml.example.edu/assertions” 이라면, abced 를 값으로 가지는 ID 를 갖는 주장에 대한 요청은 다음의 장소로 보내져야 한다.

```
https://saml.example.edu/assertions?ID=abcde
```

와일드카드의 사용은 이러한 ID 질의에서는 허용되지 않는다.

7.7.5.2. HTTP 와 캐싱 고려사항

HTTP 프락시들은 SAML 주장들을 캐시하지 않아야만 한다. 이것을 보장하기 위해, 다음 규칙들을 따라야 한다.

HTTP 1.1 을 사용하여 SAML 주장들을 반환할 때, HTTP 응답자는 다음과 같이 해야 한다:

- “no-cache, no-store”로 설정된 Cache-Control 헤더 필드를 포함해야만 한다.
- “no-cache”로 설정된 Pragma 헤더 필드를 포함해야만 한다.

7.7.5.3. 보안 고려사항

IETF RFC 2617 은 기본 또는 메시지-다이제스트 인증 구조가 사용될 때 HTTP 환경에서 어떤 공격들이 가능한지 설명한다.

TLS 1.0 의 사용이 인증, 무결성 보호 그리고 기밀성을 위한 수단으로써 강하게 권고된다.

7.7.5.4. 에러 보고

HTTP 프로토콜이 교환될 때, 적절한 HTTP 상태 코드가 요청의 결과를 가리키기 위해 사용되어야 한다. 예를 들어, SAML 요청자와 메시지를 교환하는 것을 거부하는 SAML 응답자는 “403 Forbidden” 응답을 반환해야 한다. 만약 명기된 주장이 응답자에게

알려지지 않았다면, “404 Not Found” 응답이 반환되어야 한다. 이들 경우에, HTTP 몸체의 내용은 중요하지 않다.

7.7.5.5. 메타데이터 고려사항

임의의 주장들에 대한 요청이 보내져야 하는 URL 엔드포인트들을 가리킴으로써, HTTP 상에서 URI 바인딩(URI binding over HTTP)을 지원한다는 사실을 반영해야 한다.

7.7.6. HTTP URI 를 이용하는 SAML 메시지 교환 예

다음은 주장에 대한 요청을 하는 예이다.

```
GET /SamlService?ID=abcde HTTP/1.1
Host: www.example.com
```

다음은 요청된 주장을 제공하는, 대응되는 응답의 예이다.

```
HTTP/1.1 200 OK
Content-Type: application/samlassertion+xml
Cache-Control: no-cache, no-store
Pragma: no-cache
Content-Length: nnnn
<saml:Assertion ID="abcde" ...>
...
</saml:Assertion>
```

표준 작성 공헌자

표준 번호 : TTAS.IT-X1141_2

이 표준의 제정·개정 및 발간을 위해 아래와 같이 여러분들이 공헌하였습니다.

구분	성명	위원회 및 직위	연락처	소속사
과제 제안	조영섭	PG101 위원	042-860-6942 yscho@etri.re.kr	ETRI
표준 초안 제출	조영섭	PG101 위원	042-860-6942 yscho@etri.re.kr	ETRI
표준 초안 검토 및 작성	이석래	PG101 의장	02-405-5330 sllee@kisa.or.kr	KISA
	진승헌	PG101 부의장	042-860-1254 jinsh@etri.re.kr	ETRI
	백종현	PG101 간사	02-405-5423 jhbaek@kisa.or.kr	KISA
	조상래	연구원	042-860-6939 slcho@etri.re.kr	ETRI
		외 PG101 위원		
표준안 심의	정교일	공통기반기술위워회 의장	042-860-1920 kyoil@etri.re.kr	ETRI
	원유재	공통기반기술위원회 부의장	02-405-5360 yjwon@kisa.or.kr	KISA
	이필중	공통기반기술위원회 부의장	054-279-2232 pjl@postech.ac.kr	포항공대
	김응배	공통기반기술위원회 부의장	042-860-5296 ebkim@etri.re.kr	ETRI
		외 TC1 위원		
사무국 담당	김 선	팀 장	031-724-0080 skim@tta.or.kr	TTA
	오흥룡	과 장	031-724-0083 hroh@tta.or.kr	TTA

정보통신단체표준(국문표준)

SAML 2.0 바인딩
(Bindings for SAML 2.0)

발행인 : 한국정보통신기술협회 회장
발행처 : 한국정보통신기술협회
463-824, 경기도 성남시 분당구 분당로 47
Tel : 031-724-0114, Fax : 031-724-0109
발행일 : 2006.12.