Schwachstellen in SAML 2.0 Implementationen

Roland Bischofberger · Emanuel Duss

Compass Security Schweiz AG
roland.bischofberger@compass-security.com
emanuel.duss@gmail.com

Zusammenfassung

Der SAML Standard wird vielfach eingesetzt, um eine sichere verteilte Authentifizierung umzusetzen. Eine Implementation des SAML Standards korrekt und sicher zu gestalten, stellt jedoch eine komplexe Aufgabe dar. Deshalb wurde eine Auflistung der am häufigsten beobachteten Schwachstellen in SAML Implementationen ausgearbeitet. Solche Schwachstellen sind zum Beispiel XML Signature Wrapping oder die Akzeptierung von manipulierten SAML Assertions mit gefälschten X.509 Zertifikaten. Es wurde SAML Raider, eine Extension für die Burp Suite, entwickelt, mit deren Hilfe ein SAML Service Provider auf Schwachstellen oder Fehlkonfigurationen hin überprüft werden kann. Mit dieser Extension wurde eine Schwachstelle in der SAML Implementation einer in der Schweiz verbreiteten Authentisierungslösung gefunden.

1 Einführung

Security Assertion Markup Language (SAML) ist ein Standard für die verteilte Authentifizierung und geteilte Identitäten über mehrere Organisationen hinweg. Dieser Standard wurde in vielen Produkten implementiert, so zum Beispiel in Shibboleth [Shib16] oder Microsoft ADFS [Micr16].

Viele Produkte bieten eine SAML Anbindungen und somit werden diese Produkte von Penetrationtestern auf Schwachstellen oder Fehlkonfigurationen hin getestet. [BiDu15b]

Das Testen von SAML Implementationen ist jedoch ein aufwändiger Prozess. Dazu gehören beispielsweise das Dekodieren, Manipulieren, Signieren und wieder Enkodieren von SAML Nachrichten. Diese Vorgänge sind durch ihre Komplexität fehleranfällig. [BiDu15b]

Zusätzlich hat eine SAML Nachricht meist nur eine kurze Gültigkeitsdauer. Dadurch müssen manche Schritte unter Zeitdruck durchgeführt werden, was ein korrektes und sorgfältiges Prüfen zusätzlich erschwert. Es existierte bis anhin kein Tool, mit welchem SAML Implementationen auf Schwachstellen hin getestet werden können. [BiDu15b]

Aus den vorherig aufgeführten Gründen wurde im Rahmen einer Bachelorarbeit [BiDu15b] an der Hochschule für Technik in Rapperswil eine Extension für die Burp Suite [Port16b] entwickelt. Die Burp Suite stellt diverse Tools zur Verfügung, um Webapplikationen effizient und einfach zu testen. Dabei wird auch ein Interception Proxy angeboten, der HTTP Requests und Responses abfangen, manipulieren und wieder weiterleiten kann. An diesem Punkt setzt unsere Open Source Extension mit dem Namen SAML Raider an [BiDu15a]. SAML Raider ist im offiziellen BApp Store von Burp frei verfügbar [Port16a].

P. Schartner (Hrsg.) · D•A•CH Security 2016 · syssec (2016) 36-43.

2 Related Work

Dieses Paper baut grösstenteils auf den Erkenntnissen aus der Bachelorarbeit “SAML2 Burp Plugin” [BiDu15b] auf. Die Bachelorarbeit behandelt den SAML Standard und die bekannten Schwachstellen in SAML Implementationen. Weiter wird die Entwicklung der Burp Suite Extension SAML Raider [BiDu15a] beschrieben.

Auf der Black Hat USA 2015 Konferenz wurde ein Tool namens Samlyze vorgestellt, welches einen ähnlichen Funktionsumfang wie SAML Raider haben soll [Barb15]. Leider sind weder das Tool noch genauere Informationen im Internet verfügbar.

In der Arbeit “On Breaking SAML: Be Whoever You Want to Be” [SMSK+12] wird eine Analyse der Anfälligkeit von verschiedenen SAML Frameworks auf XML Signature Wrapping Angriffe durchgeführt und es wurde auch ein Tool entwickelt, um SAML Frameworks auf XSW Attacken zu testen.

Auf der Hack-In-Taiwan Konferenz wurde das Fuzzing von SAML beschrieben und dargestellt, welche Verwundbarkeiten in OpenSAML gefunden wurden [Chen06].

3 Der SAML 2.0 Standard

SAML ist ein auf XML (Extensible Markup Language) basiertes Framework für den Austausch von Sicherheitsinformationen wie Angaben über die Authentifizierung, Eigenschaften oder Berechtigungen von Benutzern oder Systemen. [OASI08]

SAML im aktuellen Standard 2.0, wurde von der Organization for the Advancement of Structured Information Standards (OASIS) als offener Standard entwickelt. [OASI08]

Der Standard von SAML ist plattformunabhängig geschrieben. Für Benutzer bietet es den Vorteil, dass sich ein Benutzer nur noch einmal für beliebig viele Applikationen einloggen muss und sich somit nur ein Passwort merken muss. Dies wird als Single Sign-On (SSO) bezeichnet. Da das Passwort serverseitig nur noch an einem Ort abgelegt werden muss, vermindert sich der administrative Aufwand für die Benutzerverwaltung pro Applikation. [OASI08]

Vielfach wird SAML im universitären Umfeld eingesetzt. In der Schweiz zum Beispiel betreibt SWITCH mit SWITCH AAI eine SAML Umgebung, die für Studierende und Angestellte mit deren eigenen Universitätsaccounts einen Zugriff auf Ressourcen der eigenen aber auch anderer Universitäten ermöglicht. [SWIT16]

Ein anderer Use Case ist auch die Interessengemeinschaft B2B, welcher viele Versicherungen und Versicherungsbroker angehören. Diese erreicht durch die Betreibung eines zentralen SAML Identity Providers einen enorm verminderten Verwaltungsaufwand der Benutzer. [B2B16]

4 SAML Grundlagen

Die SAML Kommunikation findet immer zwischen einer SAML Asserting Party und einer SAML Relying Party statt. Die Asserting Party stellt SAML Assertions aus und die Relying Party nutzt diese Assertions. Im Single Sign-On Fall stellt ein Identity Provider (IdP) eine Assertion für einen Service Provider (SP) aus. [OASI08]

Eine SAML Assertion beinhaltet Aussagen oder Behauptungen über ein SAML Subject. Dabei handelt es sich meist um einen Benutzer. Diese Angaben sind immer in eine XML Datei ein-

gebettet. Die Assertion wird von der Asserting Party, aufgrund eines Requests von der Relying Party, erstellt [OASI05a].

Eine Assertion enthält unter anderem Informationen wer der Aussteller ist, welcher Benutzer sich authentifiziert hat und die Gültigkeitsdauer. Zusätzlich kann die SAML Nachricht signiert oder verschlüsselt sein und somit ein Zertifikat oder einen Verweis auf diesen beinhalten. [OASI05a]

Der Weg der Assertion kann unterschiedlich konfiguriert werden. Zwei dieser Varianten werden folgend näher erläutert. [OASI05a]

Beim Web Browser Single Sign-On mit HTTP POST (POST Binding) wird die Assertion vom Identity Provider in einem HTML Formular an den Client ausgeliefert und dann automatisch mittels JavaScript oder durch eine Benutzerinteraktion per HTTP POST an den Service Provider gesendet. Der genaue Ablauf ist in der Abbildung 1 dargestellt. [OASI05a]

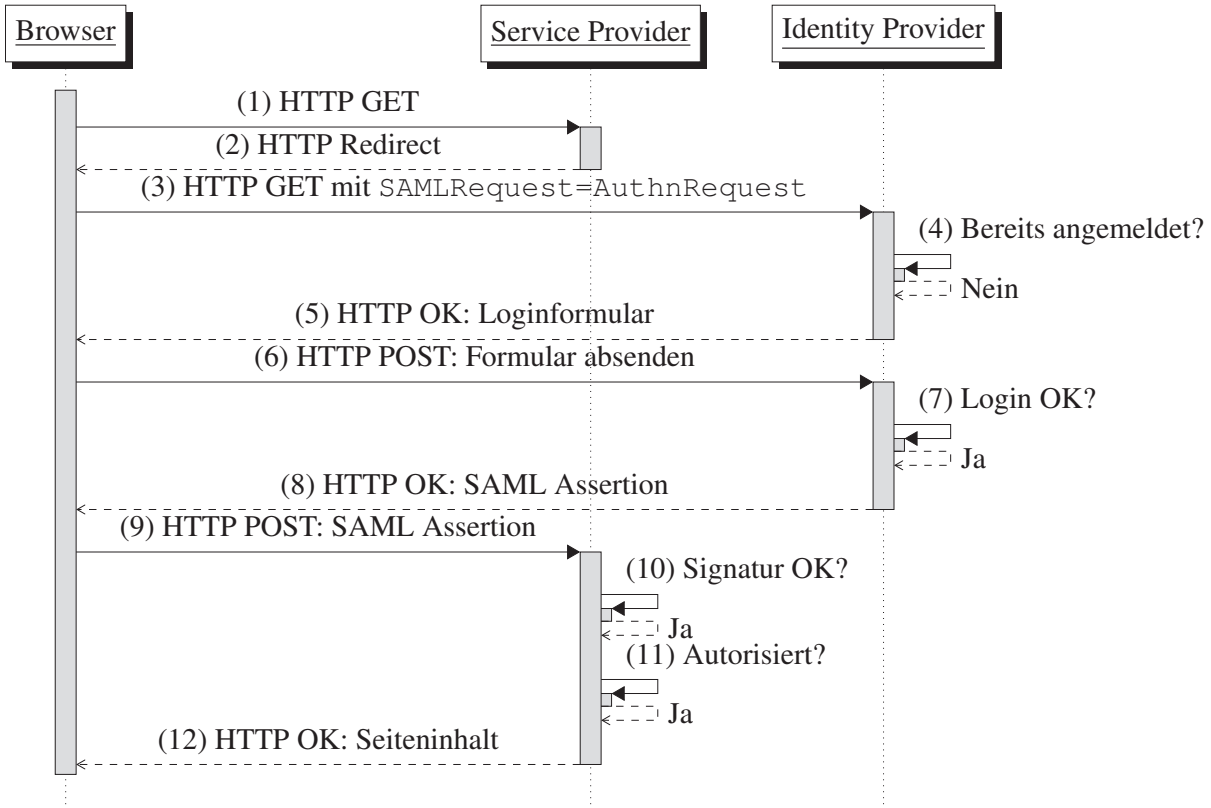

Abb. 1: Service Provider Initiated Single Sign-On (Redirect/POST Bindings) [OASI05b]

Beim Single Sign-On mit Artifacts (Artifact Binding) wird dem Client nur eine Identifikationsnummer der SAML Assertion ausgestellt. Der Client bekommt diese Assertion Identifikationsnummer per HTTP Redirect, welcher auf den Service Provider weiterleitet und somit die ID als Teil der URL mitteilt. Der Service Provider ruft danach die SAML Assertion anhand der Identifikationsnummer selbst direkt beim Identity Provider ab. Dies beutet, dass der Service Provider und der Identity Provider sich direkt erreichen können müssen. [OASI05a]

Eine Manipulation der Assertion ist somit nur im ersten Fall möglich, wenn diese über den Client gesendet wird. Um eine unbemerkte Manipulation der Assertion zu verhindern, kann

die Assertion wie auch die gesamte SAML Message vom Identity Provider signiert und/oder verschlüsselt werden. Dabei wird auf bestehende Techniken wie XML Signaturen und Public Key Verfahren gesetzt. Die Signatur kann dann auf dem Service Provider, auf welchem der Public Key vorgängig hinterlegt wurde, überprüft werden.

5 Potentielle Schwachstellen in Implementationen

In den folgenden Abschnitten 5.1, 5.2 und 5.3 werden potentielle Schwachstellen vorgestellt. Dabei werden die Vorbedingungen genannt, welche erfüllt sein müssen, damit Angreifer diese Schwachstellen ausnutzen könnten.

5.1 Inkorrekte Zertifikat-Validierung

Falls eine Assertion signiert ist, kann eine Schwachstelle in der Public Key Verwendung gesucht werden. Eine Assertion enthält zumeist ein Zertifikat, welches zur Überprüfung der Vertrauenswürdigkeit der ausgestellten Assertion eingesetzt wird. Der darin enthaltene Public Key ist ein wichtiger Bestandteil des Zertifikats und der Logik für die Validierung der Signatur.

Vorbedingungen: Ein Zertifikat ist in der Assertion eingebettet. Somit können die Tests in den Abschnitten 5.1.1, 5.1.2 und 5.1.3 potentielle Schwachstellen in der Zertifikatsvalidierung der SAML Implementation aufzeigen. Dabei ist immer als Vorbedingung gegeben, dass die Assertion signiert ist und dass das Zertifikat als Ganzes in der Assertion eingebettet wird. Auch ist dafür nötig, dass der Angreifer ein Private / Public Key Paar zur Verfügung hat.

Grundlegendes Vorgehen: Bei einem solchen Test wird die Assertion verändert, eine neue Signatur über diese veränderte Assertion berechnet und das vom Angreifer verwendete Zertifikat in die Assertion eingebettet.

5.1.1 Verwendung einer offiziellen CA

Ein Zertifikat, das von einer offiziellen CA signiert ist, wird benutzt um damit die Assertion zu signieren. Dadurch kann geprüft werden, ob der Service Provider nur Assertions akzeptiert, welche von einem fest eingetragenen Zertifikat signiert wurden.

5.1.2 Geklontes Zertifikat

Ein Zertifikat, das neu generiert und selbstsigniert ist, jedoch exakt dieselben Felder beinhaltet wie das originale Zertifikat aus der SAML Assertion, wird zur Signierung benutzt. Diese Überprüfung soll zeigen, ob der Service Provider eine Signaturprüfung anhand des lokal hinterlegten Zertifikats durchführt.

5.1.3 Widerrufenes Zertifikat

Falls ein widerrufenes Zertifikat zur Verfügung steht, kann die Assertion mit diesem signiert werden, was allfällige Schwachstellen in der Prüfung von widerrufenen Zertifikaten aufzeigen kann.

5.2 Inkorrekte Signatur-Validierung

Die korrekte Implementierung der Signaturprüfung kann mit den Tests aus den Abschnitten 5.2.1 und 5.2.2 geprüft werden.

Vorbedingung: Hierbei ist eine vorhandene Assertion inklusive Signatur als Voraussetzung gegeben.

5.2.1 XML Signatur entfernen

Die Signatur wird vollständig aus dem XML Baum entfernt. Eine korrekte Assertion wird somit ohne Signatur an den Service Provider gesendet. Dieser Test soll aufzeigen, wie der Service Provider mit einer Assertion umgeht, die nicht signiert ist. Im Falle einer falschen Fehlerbehandlung würde die Assertion trotz fehlender Signatur ausgewertet werden.

5.2.2 XML Signature Wrapping

Eine weitere Überprüfung die bei SAML Implementationen wichtig ist, nennt sich XML Signature Wrapping (XSW). Diese Technik wurde im Jahr 2012 in dem Paper "On Breaking SAML: Be Whoever You Want to Be" [SMSK+12] auf dem USENIX Security Symposium vorgestellt. Dabei wird der XML Signaturbaum so verändert, dass eine zweite Assertion in die originale Nachricht eingefügt wird (vergleiche Abbildung 2 und 3). Die Tatsache, dass die Überprüfung der Signatur und die Auswertung der Assertion-Inhalte meist getrennte Prozesse sind, führt dazu, dass die Signatur der originalen Assertion validiert wird, aber danach die Werte der zweiten falschen Assertion verwendet werden [SMSK+12].

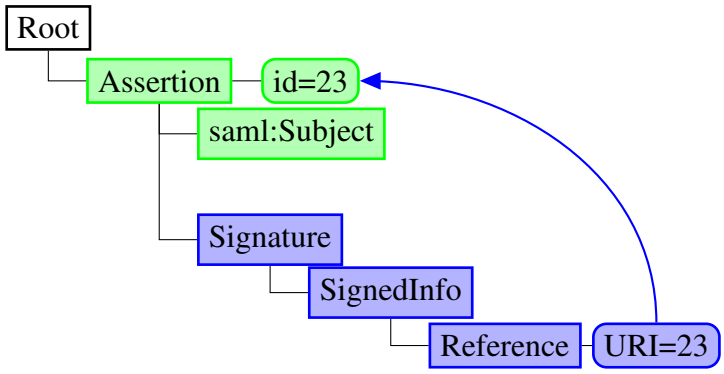

Abb. 2: Gültige SAML Assertion vor der XSW Attacke [SMSK+12]

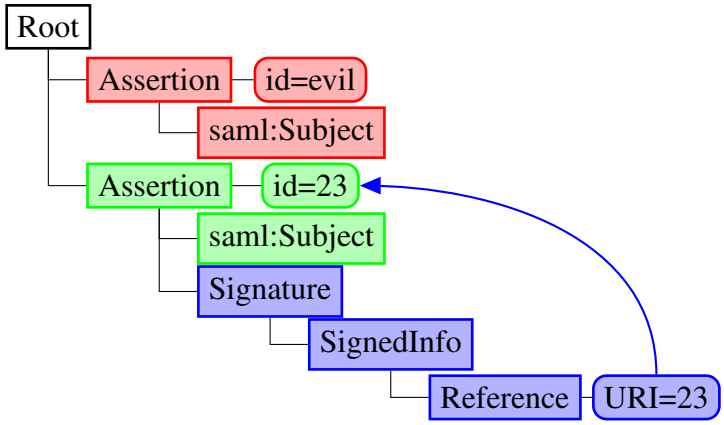

Abb. 3: Beispiel einer XSW Attacke [SMSK+12]

5.3 SAML Assertion Identifier

5.3.1 Replay SAML Message

Eine SAML Assertion beinhaltet auch jeweils einen eindeutigen Identifier (ID). Eine SAML Assertion darf nicht zweimal eingereicht werden. Der Service Provider kann dies mit diesem Identifier (ID) feststellen. Dies wird überprüft indem eine SAML Message einfach zweimal an den Service Provider gesendet wird. Lehnt der Service Provider die zweite Nachricht ab, ist die Implementation korrekt.

5.3.2 Logout von anderen Sessions

Ein wichtiger Aspekt dieses Identifiers ist es, dass dieser zufällig und nicht voraussehbar ist. Falls der Identifier voraussehbar wäre, könnten mit Logout Requests andere Benutzer ausgeloggt werden, was eine signifikante Einschränkung der Verfügbarkeit der Applikation zur Folge hätte.

6 SAML Raider

SAML Raider [BiDu15a] ist eine Open Source Extension für die Burp Suite [Port16b]. Die meisten im Abschnitt 5 genannten Punkte können mit SAML Raider geprüft werden. Mit SAML Raider wurde unter anderem in einer verbreiteten Schweizer Authentisierungslösung eine Schwachstelle gefunden, die aus einem Fehler in der Zertifikatprüfung entstanden ist [NeBi15].

Eine Übersicht über einige Funktionen soll mit der folgenden Liste gegeben werden. SAML Raider besteht aus einem Teil zur Editierung von SAML Messages und einem Teil zum Zertifikatmanagement. Ein Teil der Funktionen ist in den zwei Auflistungen in den Abschnitten 6.1 und 6.2 zu finden.

6.1 SAML Editor

Die folgenden Funktionen bietet der SAML Message Editor:

- SAML Messages automatisch (de-)kodieren
- SAML Messages und Assertions signieren
- Signaturen entfernen
- Acht XML Signature Wrapping Attacken automatisch ausführen
- SAML Message editieren
- Eingebettete Zertifikate extrahieren

6.2 Zertifikatmanagement

Die folgenden Funktionen bietet der Zertifikateditor:

- Zertifikate importieren und exportieren
- Zertifikatfelder anzeigen
- Private Keys importieren und exportieren
- Zertifikate erstellen

- Zertifikate “klonen” (das heißt alle Felder 1:1 kopieren, jedoch neues Schlüsselmaterial generieren)
- Zertifikatfelder editieren und daraus selbstsignierte Zertifikate generieren

7 Gegenmassnahmen

Die Gegenmaßnahmen gegen Angriffe auf SAML Umgebungen stehen nicht im Fokus dieser Arbeit. Trotzdem werden in diesem Abschnitt einige Überlegungen festgehalten, die das Sicherheitsniveau einer SAML Umgebung wesentlich verbessern können.

Wird das Artifact Binding verwendet, kann eine Manipulation der Assertion auf dem Client ausgeschlossen werden, da die Assertion direkt von Identity Provider zum Service Provider gesendet wird. Dies stellt jedoch die Anforderung, dass der Identity Provider und der Service Provider direkt kommunizieren können.

Sollte die Verwendung des Artifact Bindings nicht möglich sein und somit die Assertion über den Client ausgeliefert werden, ist eine Steigerung der Sicherheit durch die Verschlüsselung der SAML Messages möglich, da somit eine Veränderung der Assertion Signatur und des dazugehörigen Zertifikates ohne gültiges Schlüsselmaterial erheblich erschwert wird.

Falls ein Produkt mit einer SAML Implementation entwickelt wird, sollte zum einen sichergestellt werden, dass nur der signierte Teilbaum zum Auslesen der Werte verwendet wird und alle anderen Inhalte gelöscht oder ignoriert werden [SMSK+12]. Auch ist unbedingt darauf zu achten, dass immer das Zertifikat, welches auf dem Service Provider hinterlegt ist, zur Überprüfung der Signatur verwendet wird. Das in der SAML Nachricht eingebettete Zertifikat soll nur zur Identifikation des richtigen Schlüsselmaterials verwendet werden.

8 Zusammenfassung und Ausblick

In dieser Arbeit werden die Grundlagen zu SAML erläutert. Darauf folgend werden diverse Sicherheitsprobleme von SAML Implementationen diskutiert. Dabei werden im Besonderen Fehler in der Validierung der Zertifikate wie auch der Signaturen von SAML Messages behandelt. Anschließend werden einige wenige Gegenmaßnahmen beschrieben, welche das Sicherheitsniveau von SAML Umgebungen verbessern können. Für die Sicherheitsüberprüfung von SAML Implementationen auf Schwachstellen hin wird SAML Raider vorgestellt und dessen wichtigste Features beschrieben. Mit SAML Raider wurden auch Schwachstellen in der Praxis identifiziert.

Um einen Überblick über mögliche Schwachstellen in SAML Implementationen zu erhalten, wäre in Zukunft eine Evaluierung von verbreiteten Implementationen sinnvoll. Auch wurde das Tool SAML Raider noch nicht mit vielen verschiedenen SAML Umgebungen getestet und es kann deshalb durch breit abgestütztes Testing in seiner Robustheit noch verbessert werden. Bis anhin kann mit dem Tool nur halbautomatisch getestet werden und ein automatischer Modus wäre eine gute Variante, um die Effizienz zu steigern und zum Beispiel Fuzzing zu ermöglichen.

Literatur

[B2B16] I. B2B: IG B2B for Insurers + Brokers (2016) https://www.switch.ch/aai

[Barb15] J. Barber: Black Hat USA 2015 (2015) https://www.blackhat.com/us-15/arsenal.html#jon-barber

[BiDu15a] R. Bischofberger, E. Duss: GitHub: SAML Raider (2015) https://github.com/SAMLRaider/SAMLRaider

[BiDu15b] R. Bischofberger, E. Duss: SAML2 Burp Plugin (2015) https://eprints.hsr.ch/464

[Chen06] Y.-M. Chen: Fuzzing XML Based Protocols (SAML) (2006) http://hitcon.org/download/2006/HIT2006OpenSAML.pdf

[Micr16] Microsoft: Active Directory Federation Services (2016) https://msdn.microsoft.com/en-us/library/bb897402.aspx

[NeBi15] A. Neuenschwander, R. Bischofberger: CVE-2015-5372 AdNovum nevisAuth Authentication Bypass (2015) https://www.compass-security.com/fileadmin/Datein/Research/Advisories/CVE-2015-5372_AdNovum_nevisAuth_Authentication_Bypass.txt

[OASI05a] OASIS: Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0 (2005) http://docs.oasis-open.org/security/saml/v2.0

[OASI05b] OASIS: Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0 (2005) http://docs.oasis-open.org/security/saml/v2.0

[OASI08] OASIS: SAML V2.0 Executive Overview (2008) https://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf

[Port16a] Portswigger: BApp Store, https://portswigger.net/bappstore (2016)

[Port16b] Portswigger: Burp Suite, https://portswigger.net/burp (2016)

[Shib16] Shibboleth: Shibboleth, https://shibboleth.net (2016)

[SMSK+12] J. Somorovsky, A. Mayer, J. Schwenk, M. Kampmann, M. Jensen: On Breaking SAML: Be Whoever You Want to Be. https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/somorovsky (2012)

[SWIT16] SWITCH: SWITHCaai, https://www.switch.ch/aai (2016)