Verifying Policy-Based Security for Web Services

CS 563 / ECE 524 Spring 2021
Zhenyu Mao
SOAP

XML Document
Used to exchange structured information in web services

Is SOAP still in use?

- Outlook (EWS, EAS)
- Bank System
- Telecommunication
- Government API [NOAA, etc.]

Compare with RESTful (eg. JSON)?
WS-Security and XML Rewriting Attack

Main goal of WS-Security:
- sign, encrypt SOAP messages
- attach security tokens to ascertain the sender's identity

What is XML Rewriting Attack?
- Adding new elements to the SOAP header without compromising the contents of the message.

← Redirection Attack
- Change something out of protect

Ref: XML REWRITING ATTACKS: EXISTING SOLUTIONS AND THEIR LIMITATIONS. arXiv:0812.4181; A Formal Solution to Rewriting Attacks on SOAP Messages
**WS-Security and XML Rewriting Attack**

Main goal of WS-Security:
- sign, encrypt SOAP messages
- attach security tokens to ascertain the sender's identity

What is XML Rewriting Attack?
- Adding new elements to the SOAP header without compromising the contents of the message.

← Reply Attack
- Change the body (loan money)

---

Ref: XML REWRITING ATTACKS: EXISTING SOLUTIONS AND THEIR LIMITATIONS. arXiv:0812.4181; A Formal Solution to Rewriting Attacks on SOAP Messages
WS-Security and XML Rewriting Attack

Main goal of WS-Security:
- sign, encrypt SOAP messages
- attach security tokens to ascertain the sender's identity

What is XML Rewriting Attack?
- Adding new elements to the SOAP header without compromising the contents of the message.

← Multiple Security Header attack
- Override the origin header

Ref: XML REWRITING ATTACKS: EXISTING SOLUTIONS AND THEIR LIMITATIONS. arXiv:0812.4181; A Formal Solution to Rewriting Attacks on SOAP Messages
We define:

- Number of children of Envelope is 2
- Number of Header is 2.
- Number of Signed Elements is 3

However:

- Vulnerable to Replay Attack itself (i.e. forgery SOAPAccount header)
Defense: WS-SecurityPolicy

- Domain specific language based on WS-Policy, expressed in WSDL
- Define things to be protected
- Define token to be used
- Define cryptographic communications protocols
- Uses low-level mechanisms that build and check individual security headers.
- Too complicated. Hard to get right

Ref: WS-SecurityPolicy Examples Oasis
Our Goals

We want to define abstract and application level goals and turn it into WS-SecurityPolicy configuration files.

1. Easy to Write/Update [Less Painful]
   
   Link language is a simple notation, covering some common cases, and could easily be generated from a simple UI or a systems modelling tool.

2. Security
   
   It is safer to generate policy files from link specifications than write them directly.

Verify policy files
Architecture of Policy Files

TulaFale
pi-Calculus
ProVerif
Links -> Policies
Policies -> Predicate
Authentication and Adequacy Goals

A process \( P \) is **robustly safe** when, for any run in any context, if end Log(a) occurs, then either begin Log(a) or begin Leak(u) with \([a = u @] \) previously occurred.

\( P \) is **functionally adequate for a** when, for some run in some context, end Log(a) occurs.

\[
\begin{align*}
\text{process } \text{GenericSender}() = \\
& \text{!in initChan(env)}; \\
& \text{in dbChan(sid); in dbChan(rid);} \\
& \text{new freshid;} \\
& \text{filter mkConformant(env,[sid],[rid],[freshid],outenv)} \\
& \rightarrow \text{!outenv in} \\
& \text{filter linkAssert(sid,rid,env,a) } \rightarrow \text{a in begin (Log,a);} \\
& \text{out (httpChan, outenv)} \\
\end{align*}
\]

\[
\begin{align*}
\text{process } \text{GenericReceiver}() = \\
& \text{!in httpChan(env)}; \\
& \text{in dbChan(sid); in dbChan(rid);} \\
& \text{filter isConformant(env,[sid],[rid],outenv) } \rightarrow \text{outenv in} \\
& \text{filter linkAssert(sid,rid,outenv,a) } \rightarrow \text{a in end (Log,a);} \\
& \text{done}
\end{align*}
\]
Useful Predicates

**predicate** mkConformant(env: item, sids, rids, fresh: items, outenv: item) :-
  hasHeaderTo(env, Toitm, Toval),
  hasHeaderAction(env, Actionitm, Actionval),
  Toval = "http://bospetshop.com/service.asmx",
  Actionval = "http://petshop/regular",
  hasSendPolicyClientToService(env, sids, rids, fresh, outenv).

**predicate** hasReceivePolicyServiceToClient(env: item, sids, rids: items,
  outenv: item) :-
  hasResponseParts(env, Fromitm, RelatesToitm, MessageIditm,
  Createditm, Bodyitm),
  hasSecurityHeader(env, toks),
  xtok in toks, sig in toks,
  isX509Token(xtok, "BobsPetShop", k, sids),
  isSignature(sig, "rsasha1", k,
    [Fromitm, RelatesToitm, MessageIditm, Createditm,
    Bodyitm]),
  outenv = env.

**predicate** isConformant(env: item, sids, rids: items, outenv: item) :-
  hasReceivePolicyServiceToClient(env, sids, rids, outenv).

**predicate** linkAssert(sid, rid: item, env: item, a: items) :-
  hasUid(sid, sender), hasUid(rid, responder)
  hasHeaderTo(env, Toitm, to),
  hasHeaderAction(env, Actionitm, action),
  hasHeaderMessageId(env, MessageIditm, id),
  hasHeaderCreated(env, Createditm),
  hasBody(env, btm, body),
  to = "http://bospetshop.com/service.asmx",
  action = "http://premium",
  responder in ["BobsPetShop"],
  a = [sender responder "Request" to action id timestamp body].

**predicate** hasSendPolicyClientToService(env: item, sids, rids, fresh: items,
  outenv: item) :-
  sids = [user @ _],
  isUserPassword(user, up),
  fresh = [NewMessageId n t @ _],
  hasRequestParts(env, Toitm, Actionitm, MessageIditm,
  Createditm, Bodyitm),
  MessageIditm = <MessageId>NewMessageId</>,
  mkUserTokenKey(utok, u, p, n, t, k),
  mkSignature(sig, "hmacsha1", k,
    [Toitm, Actionitm, MessageIditm, Createditm, Bodyitm]),
  outenv = <Envelope>
    <Header>
      Toitm Acitm MessageIditm
      <Security>
        <Timestamp>Createditm</>
        utok sig </</
      Bodyitm </>
    <Body>
      "Request"
    </Body>
  </Envelope>.
Security Models: Secrecy

The generic sender inputs a request envelope from the attacker for recipient u

Try to replace the body by a secret name B

P preserves secrecy when, for any run in any context where B does not occur, if the context obtains B, then begin Leak(u) and begin KnowsSecret(u) previously occurred.
Security Models: Correlation

When the client accepts a response message from the web service, we want to guarantee that this message was generated in response to a particular earlier request.

```
process GenericClient() =
  in initChan(env);
  in dbChan(cid); in dbChan(sid);
  new freshid;
  filter mkConformant(env,[cid],[sid],[freshid],outenv) → outenv in
  filter linkAssert(cid,sid,env,aReq) → aReq in
  begin Log(aReq);
  out httpChan(outenv);

  in httpChan(respenv);
  filter isConformant(respenv,[sid],[cid],resp) → resp in
  filter hasCorrelator(resp,freshid,cid) → in
  filter hasLinkAssert(sid,cid,resp,aResp) → aResp in
  end Log(aResp);
  end LogCorr(aReq,aResp)
```
Discussion

We already have TLS (encryption) or other lower level mechanism to ensure integrity. Why do we still need this (Security-Policy)?