

# Securing Web Services, Distributed Databases

19 January 2010  
Lecture 14

Some slides modified from Silberschatz, *et al.*

## Topics for Today

- Securing Web Services and XML
- Distributed Databases – A Retrospective
  - Two Phase Commit

January 19, 2010

ISE 431: Distributed Information Systems

2

## Securing Web Services

- Web Services are a communication medium
- XML is a communication format
  - SOAP uses XML
- Two possible places to secure:
  - The communication medium
  - The communication format
- Securing the medium means making the communication path secure
  - Could encrypt with SSL, public/private key encryption, DES
  - Could sign with hashes, RSA digital signatures
- Point-to-point security - Been there, done that...

January 19, 2010

ISE 431: Distributed Information Systems

3

## XML Security

- Securing the format – adding security features to XML
- XML Security Standards:
  - Signing XML Documents
  - Key management
  - Encryption
- We may secure
  - All of part of a document
  - The same document in multiple ways
  - Some other document (using this one) by reference
- There are **many standards**, this only a sampling

January 19, 2010

ISE 431: Distributed Information Systems

4

## Long Lasting Security

**Key observation:** Secure XML documents that have a lifetime beyond a single transmission

- Example:** A person's medical history stored in XML format
- Document has information from doctors – digitally signed by the doctor
  - Might be visible to certain doctors, hospitals, or roles - encryption
  - Diagnoses or prescriptions may be digitally signed to track who gave them
  - Practically, such a document probably couldn't leave an organization's bounds, but within a large קופת חולים or hospital, it could work

January 19, 2010

ISE 431: Distributed Information Systems

5

## Why is this interesting?

Long lasting security differs from "Secure the Medium"

- We must store documents for a long time, so point-to-point security isn't enough
  - There may be multiple routing hops
- We may need multiple signatures or encryptions on a single document
- We need a way to identify signers and recover keys

XML is well suited to handling such requirements:

- Self descriptive
- Hierarchy makes division into subtrees natural
- References are first order objects, so pointing is easy

January 19, 2010

ISE 431: Distributed Information Systems

6

## Requirements (1/2)

### Basic Requirements:

- To be able to **encrypt** either an entire document or just some selected part of it
- To be able to **sign** an entire document or just some selected parts of it

### Additional Basic Requirements:

- To **add** to a document that is already signed and to **sign the result**
- To **add** to a document that already contains encrypted sections and to **encrypt part of the new version**, possibly including some of the already encrypted sections
- To **authorize** various different users to view different parts of a document

January 19, 2010

ISE 431: Distributed Information Systems

7

## Requirements (2/2)

### Algorithms:

- The standard should specify a **suite of algorithms** to be provided in any implementation of XML security.
- The algorithms used for encryption and authentication of a particular document must be **selected from that suite** and the **names of the algorithms in use must be referenced within the XML document itself**

### Finding Keys:

- To help the users of secure documents with **finding the necessary keys**
- To make it possible for **cooperating users to help one another with keys**

January 19, 2010

ISE 431: Distributed Information Systems

8

## Supporting Tools

The (optional) **KeyInfo** element helps with many of these

- Can hold information on:

- The algorithm used
- The key used
- The certificate used

### Canonical XML – a standard form for XML data

- Two semantically identical XML documents will be the same when canonicalized

- Unifies

- Whitespace
- Attribute order (lexicographical)
- Default attributes
- Extra or unused namespaces
- Encoding (UTF-8) and line breaks

January 19, 2010

ISE 431: Distributed Information Systems

9

## Digital Signatures with XML

Type of Algorithm	Name of Algorithm	Required
Message Digest	SHA-1	Required
Encoding	base64	Required
Signature (asymmetric)	DSA with SHA-1	Required
	DSA with SHA-1	Recommended
MAC signature (symmetric)	HMAC-SHA-1	Required
Canonicalization	Canonical XML	Required

```
[s01] <Signature Id="MyFirstSignature" xmlns="http://www.w3.org/2000/09/xmldsig#"
[s02]   <SignedInfo
[s03]     <CanonicalizationMethod Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
[s04]     <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
[s05]     <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-2000126/">
[s06]       <Transform>
[s07]         <Transform Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
[s08]       </Transform>
[s09]     <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
[s10]     <DigestValue>8Bp0p0p-cyub3qg78t00kduXR1cm0K.../DigestValue
[s11]   </Reference>
[s12] </SignedInfo>
[s13] <SignatureValue>.../SignatureValue
[s14] <KeyInfo
[s15]   <KeyValue
[s16]     <GSAKeyValue
[s17]       <P>...</P><Q>...</Q><D>...</D><T>...</T>
[s18]     </GSAKeyValue>
[s19]   </DSASigKeyValue
[s20] </KeyInfo
[s21] </Signature>
```

January 19, 2010

ISE 431: Distributed Information Systems

10

## XML Encryption

- XML Security specifies algorithms which everyone must support

- Encryption
- Key transport (encrypting a key with another key to send it)
- Key agreement (derive a key offline without communication)

Type of Algorithm	Name of Algorithm	Required
Block Cipher	TRIPLEDES, AES-128, AES-256, AES-192	Required Optional
Encoding	base64	Optional
Key Transport	RSA-v1.5, RSA-OAEP	Required
Symmetric Key Wrap	TRIPLEDES KeyWrap AES-128 KeyWrap AES-256 KeyWrap AES-192 KeyWrap	Required Optional Optional Optional
Key Agreement	Diffie-Hellman	Optional

January 19, 2010

ISE 431: Distributed Information Systems

11

## Encryption 1: A Subtree

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <EncryptedData Type='http://www.w3.org/2001/04/xmenc#Element'
    xmlns='http://www.w3.org/2001/04/xmenc#'>
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
</PaymentInfo>
```

January 19, 2010

ISE 431: Distributed Information Systems

12

## Encryption 2: Children and CData

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
      Type='http://www.w3.org/2001/04/xmlenc#Content'>
      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
  </CreditCard>
</PaymentInfo>
```

January 19, 2010

ISE 431: Distributed Information Systems

13

## Encryption 3: CData

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
        Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C56</CipherValue>
        </CipherData>
      </EncryptedData>
    </Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

January 19, 2010

ISE 431: Distributed Information Systems

14

## So Far

- Securing Web Services and XML
- Distributed Databases – A Retrospective
  - Two Phase Commit

January 19, 2010

ISE 431: Distributed Information Systems

15

## Distributed Databases

- Heterogeneous and Homogeneous Databases
- Distributed Data Storage
- Distributed Transactions
- Commit Protocols

January 19, 2010

ISE 431: Distributed Information Systems

16

## Distributed Database System

- A distributed database system consists of loosely coupled sites that share no physical component
- Database systems that run on each site are independent of each other
- Transactions may access data at one or more sites

January 19, 2010

ISE 431: Distributed Information Systems

17

## Homogeneous Distributed DBs

- In a **homogeneous** distributed database
  - All sites have identical software
  - Are aware of each other and agree to cooperate in processing user requests.
  - Each site surrenders part of its autonomy in terms of right to change schemas or software
  - Appears to user as a single system
- In a **heterogeneous** distributed database
  - Different sites may use different schemas and software
    - Difference in schema is a major problem for query processing
    - Difference in software is a major problem for transaction processing
  - Sites may not be aware of each other and may provide only limited facilities for cooperation in transaction processing

January 19, 2010

ISE 431: Distributed Information Systems

18

## Distributed Data Storage

- Assume relational data model
- **Replication**
  - System maintains multiple copies of data, stored in different sites, for faster retrieval and fault tolerance.
- **Fragmentation**
  - Relation is partitioned into several fragments stored in distinct sites
- **Replication and fragmentation can be combined**
  - Relation is partitioned into several fragments: system maintains several identical replicas of each such fragment.

January 19, 2010

ISE 431: Distributed Information Systems

19

## Data Replication

- A relation or fragment of a relation is **replicated** if it is stored redundantly in two or more sites.
- Full replication of a relation is the case where the relation is stored at all sites.
- Fully redundant databases are those in which every site contains a copy of the entire database.

January 19, 2010

ISE 431: Distributed Information Systems

20

## Data Replication (Cont.)

- **Advantages of Replication**
  - **Availability:** failure of site containing relation  $r$  does not result in unavailability of  $r$  if replicas exist.
  - **Parallelism:** queries on  $r$  may be processed by several nodes in parallel.
  - **Reduced data transfer:** relation  $r$  is available locally at each site containing a replica of  $r$ .
- **Disadvantages of Replication**
  - **Increased cost of updates:** each replica of relation  $r$  must be updated.
  - **Increased complexity of concurrency control:** concurrent updates to distinct replicas may lead to inconsistent data unless special concurrency control mechanisms are implemented.
    - One solution: choose one copy as **primary copy** and apply concurrency control operations on primary copy

January 19, 2010

ISE 431: Distributed Information Systems

21

## Data Fragmentation

Division of relation  $r$  into fragments  $r_1, r_2, \dots, r_n$  which contain sufficient information to reconstruct relation  $r$ .

**Horizontal fragmentation:** each tuple of  $r$  is assigned to one or more fragments

**Vertical fragmentation:** the schema for relation  $r$  is split into several smaller schemas

- All schemas must contain a common candidate key (or superkey) to ensure lossless join property.
- A special attribute, the tuple-id attribute may be added to each schema to serve as a candidate key.

**Example:** relation *account* with following schema

- $Account = (branch\_name, account\_number, balance)$

January 19, 2010

ISE 431: Distributed Information Systems

22

## Horizontal Fragmentation of *account*

branch_name	account_number	balance
Hillside	A-305	500
Hillside	A-226	336
Hillside	A-155	62

$$account_1 = \sigma_{branch\_name='Hillside'}(account)$$

branch_name	account_number	balance
Valleyview	A-177	205
Valleyview	A-402	10000
Valleyview	A-408	1123
Valleyview	A-639	750

$$account_2 = \sigma_{branch\_name='Valleyview'}(account)$$

January 19, 2010

ISE 431: Distributed Information Systems

23

## Vertical Fragmentation of *employee\_info*

branch_name	customer_name	tuple_id
Hillside	Lowman	1
Hillside	Camp	2
Valleyview	Camp	3
Valleyview	Kahn	4
Hillside	Kahn	5
Valleyview	Kahn	6
Valleyview	Green	7

$$deposit_1 = \Pi_{branch\_name, customer\_name, tuple\_id}(employee\_info)$$

account_number	balance	tuple_id
A-305	500	1
A-226	336	2
A-177	205	3
A-402	10000	4
A-155	62	5
A-408	1123	6
A-639	750	7

$$deposit_2 = \Pi_{account\_number, balance, tuple\_id}(employee\_info)$$

January 19, 2010

ISE 431: Distributed Information Systems

24

## Advantages of Fragmentation

- **Horizontal:**
  - allows parallel processing on fragments of a relation
  - allows a relation to be split so that tuples are located where they are most frequently accessed
- **Vertical:**
  - allows tuples to be split so that each part of the tuple is stored where it is most frequently accessed
  - tuple-id attribute allows efficient joining of vertical fragments
  - allows parallel processing on a relation
- **Vertical and horizontal fragmentation can be mixed.**
  - Fragments may be successively fragmented to an arbitrary depth.

## Data Transparency

- **Data transparency:** Degree to which system user may remain unaware of the details of how and where the data items are stored in a distributed system
- Consider transparency issues in relation to:
  - Fragmentation transparency
  - Replication transparency
  - Location transparency

## Naming of Data Items - Criteria

1. Every data item must have a system-wide unique name.
2. It should be possible to find the location of data items efficiently.
3. It should be possible to change the location of data items transparently.
4. Each site should be able to create new data items autonomously.

## Centralized Scheme - Name Server

- **Structure:**
  - name server assigns all names
  - each site maintains a record of local data items
  - sites ask name server to locate non-local data items
- **Advantages:**
  - satisfies naming criteria 1-3
- **Disadvantages:**
  - does not satisfy naming criterion 4
  - name server is a potential performance bottleneck
  - name server is a single point of failure

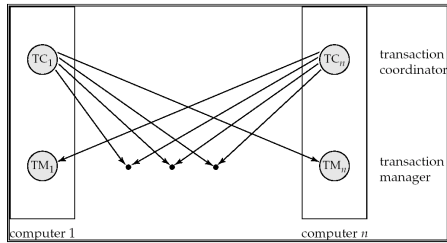
## Use of Aliases

- **Alternative to centralized scheme:** each site prefixes its own site identifier to any name that it generates i.e., *site 17.account*.
  - Fulfills having a unique identifier, and avoids problems associated with central control.
  - However, fails to achieve network transparency.
- **Solution:** Create a set of **aliases** for data items; Store the mapping of aliases to the real names at each site.
- The user can be unaware of the physical location of a data item, and is unaffected if the data item is moved from one site to another.

## Distributed Transactions

- Transaction may access data at several sites.
- Each site has a local transaction manager responsible for:
  - Maintaining a log for recovery purposes
  - Participating in coordinating the concurrent execution of the transactions executing at that site.
- Each site has a transaction coordinator, which is responsible for:
  - Starting the execution of transactions that originate at the site.
  - Distributing subtransactions at appropriate sites for execution.
  - Coordinating the termination of each transaction that originates at the site, which may result in the transaction being committed at all sites or aborted at all sites.

## Transaction System Architecture



January 19, 2010

ISE 431: Distributed Information Systems

31

## System Failure Modes

- Failures unique to distributed systems:
  - Failure of a site.
  - Loss of messages
    - Handled by network transmission control protocols such as TCP-IP
  - Failure of a communication link
    - Handled by network protocols, by routing messages via alternative links
  - Network partition**
    - A network is said to be **partitioned** when it has been split into two or more subsystems that lack any connection between them
      - Note: a subsystem may consist of a single node
- Network partitioning and site failures are generally indistinguishable.

January 19, 2010

ISE 431: Distributed Information Systems

32

## Commit Protocols

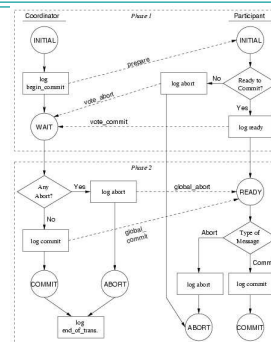
- Commit protocols are used to ensure atomicity across sites
  - a transaction which executes at multiple sites must either be committed at all the sites, or aborted at all the sites.
  - not acceptable to have a transaction committed at one site and aborted at another
- The *two-phase commit* (2PC) protocol is widely used
- The *three-phase commit* (3PC) protocol is more complicated and more expensive, but avoids some drawbacks of two-phase commit protocol. This protocol is not used in practice.

January 19, 2010

ISE 431: Distributed Information Systems

33

## Two Phase Commit



January 19, 2010

ISE 431: Distributed Information Systems

34

## Two Phase Commit Protocol (2PC)

- Assumes **fail-stop** model – failed sites simply stop working, and do not cause any other harm, such as sending incorrect messages to other sites.
- Execution of the protocol is initiated by the coordinator after the last step of the transaction has been reached.
- The protocol involves all the local sites at which the transaction executed
- Let  $T$  be a transaction initiated at site  $S_p$ , and let the transaction coordinator at  $S_i$  be  $C_i$

January 19, 2010

ISE 431: Distributed Information Systems

35

## Phase 1: Obtaining a Decision

- Coordinator asks all participants to *prepare* to commit transaction  $T_p$ .
  - $C_i$  adds the records  $\langle \text{prepare } T \rangle$  to the log and forces log to stable storage
  - sends **prepare**  $T$  messages to all sites at which  $T$  executed
- Upon receiving message, transaction manager at site determines if it can commit the transaction
  - if not, add a record  $\langle \text{no } T \rangle$  to the log and send **abort**  $T$  message to  $C_i$
  - if the transaction can be committed, then:
    - add the record  $\langle \text{ready } T \rangle$  to the log
    - force *all records* for  $T$  to stable storage
    - send **ready**  $T$  message to  $C_i$

January 19, 2010

ISE 431: Distributed Information Systems

36

## Phase 2: Recording the Decision

- $T$  can be committed if  $C_i$  received a **ready**  $T$  message from all the participating sites: otherwise  $T$  must be aborted.
- Coordinator adds a decision record, **<commit  $T$ >** or **<abort  $T$ >**, to the log and forces record onto stable storage. Once the record stable storage it is irrevocable (even if failures occur)
- Coordinator sends a message to each participant informing it of the decision (commit or abort)
- Participants take appropriate action locally.
  - They may (optionally) acknowledge the commit/abort

January 19, 2010

ISE 431: Distributed Information Systems

37

## Conclusion

- Securing Web Services and XML
- Distributed Databases – A Retrospective
  - Two Phase Commit

January 19, 2010

ISE 431: Distributed Information Systems

38