Towards An Architectural Treatment of Software Security: A Connector-Centric Approach

Jie Ren, Richard Taylor, Paul Dourish, David Redmiles
Institute for Software Research
University of California, Irvine

Software Engineering for Secure Systems
May 15, 2005

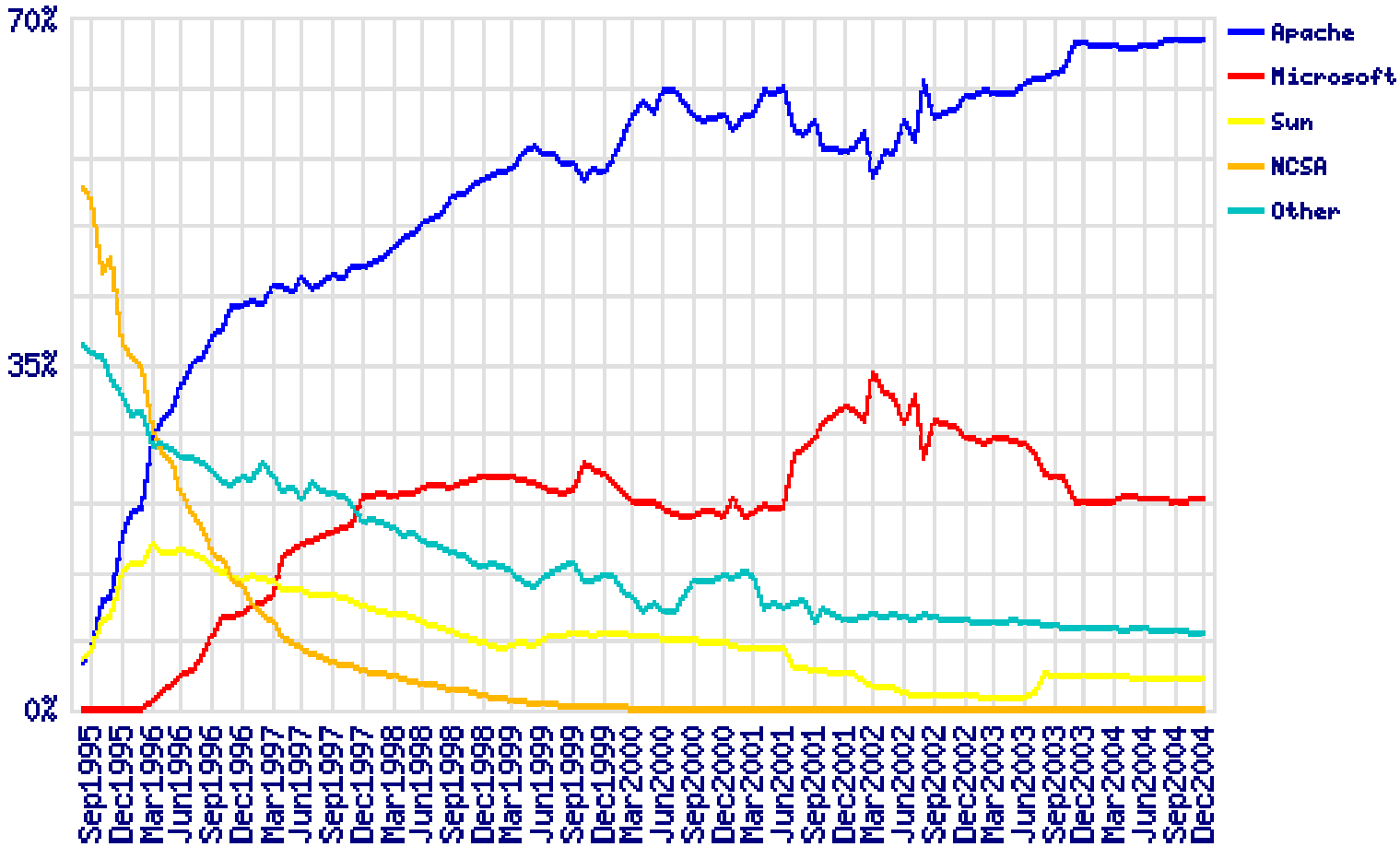
Outline

- Background and Insight
  - Architecture and Security
- Approach
  - Connector-centric
- Case Study
  - A not-so-small example application
- Conclusion and future work

Main Goal

- Integrate security and software architecture
  - Integrate
  - Architecture level
  - Security: confidentiality, integrity, availability
  - Modern software: componentized, networked, heterogeneous

A Tale of IIS


70%
35%
0%
Apache
Microsoft
Sun
NCSA
Other
Sep1995
Dec1995
Mar1996
Jun1996
Sep1996
Dec1996
Mar1997
Jun1997
Sep1997
Dec1997
Mar1998
Jun1998
Sep1998
Dec1998
Mar1999
Jun1999
Sep1999
Dec1999
Mar2000
Jun2000
Sep2000
Dec2000
Mar2001
Jun2001
Sep2001
Dec2001
Mar2002
Jun2002
Sep2002
Dec2002
Mar2003
Jun2003
Sep2003
Dec2003
Mar2004
Jun2004
Sep2004
Dec2004

The history of IIS

Version	Date	OS	# Security Bulletins
1.0	1995	NT 3.51	
2.0	1996	NT 4	
3.0	1997	NT 4	8
4.0	1997	NT 4	42
5.0	2000	2000	29
5.1	2001	XP	5
6.0	2003	2003	1

Re-architecting boosts security!

Table 1. Secure by design.

POTENTIAL PROBLEM	PROTECTION MECHANISM	DESIGN PRINCIPLES
The underlying `dll (ntdll.dll)` was not vulnerable because...	Code was made more conservative during the Security Push.	Check precondition
Even if it were vulnerable...	Internet Information Services (IIS) 6.0 is not running by default on Windows Server 2003.	Secure by default
Even if it were running...	IIS 6.0 does not have WebDAV enabled by default.	Secure by default
Even if Web-based Distributed Authoring and Versioning (WebDAV) had been enabled...	The maximum URL length in IIS 6.0 is 16 Kbytes by default (> 64 Kbytes needed for the exploit).	Tighten precondition, secure by default
Even if the buffer were large enough...	The process halts rather than executes malicious code due to buffer-overrun detection code inserted by the compiler.	Tighten postcondition, check precondition
Even if there were an exploitable buffer overrun...	It would have occurred in w3wp.exe, which is running as a network service (rather than as admininstrator).	Least privilege

(Data courtesy of David Aucsmith.)

Wing, IEEE Security & Privacy, 2003

Traditional SA

- Component-based Software Engineering
- Software Architecture
  - Structure
  - Behavior
    - Process Algebra (Wright), Labeled Transition System (Darwin)

Connectors

- Should they be first class citizens?
  - Capture and reuse
- Existing work
  - Taxonomy: Mehta 2000
  - Assembly Language: Mehta 2004
  - Constructions: Lopes 2003
  - Transformation: Spitznagel 2001
- No rich security
  - Dependability: Spitznagel 2004

Our Approach

- Describe and Enforce Architectural Security
  - Extend xADL
  - Security Models, Users (Principals), Privileges, Trusts, Contexts
  - Component: supply security contract
  - Connector: regulate and enforce contract

Extensible xADL

- xADL provides structural infrastructure and enables extension
- Security Models Support
  - Extensible and Neutral
  - Classic Access Control, Role-based Access Control, Trust Management

Security Constructs

- Users (Principals)
  - Single principal, determinable at design time, no impact on architecture
- Privileges
  - Traditional access control privileges
  - Architectural privileges: change, inspect
- Contexts
- Trust

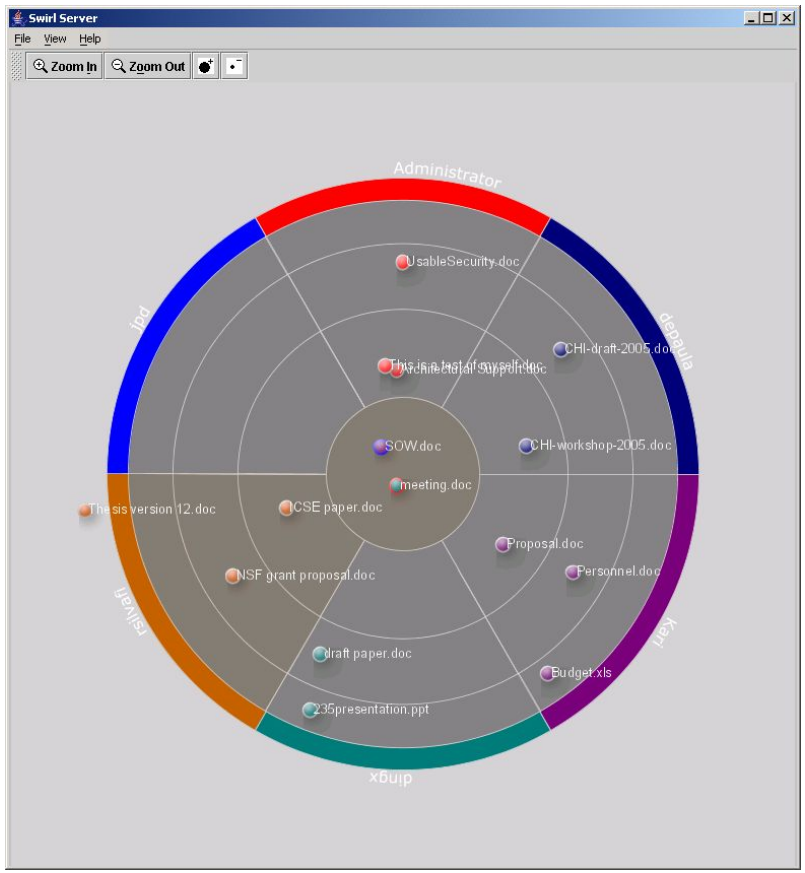
Component and Connector

- Component
  - Supply “security contract”
  - Requires and Provides
- Connector
  - Regulate and enforce “security contract”
  - Determine components’ principals
  - Decide compatibility
  - Adapt incompatibilities and impose security
  - Derivation, composition, replacement

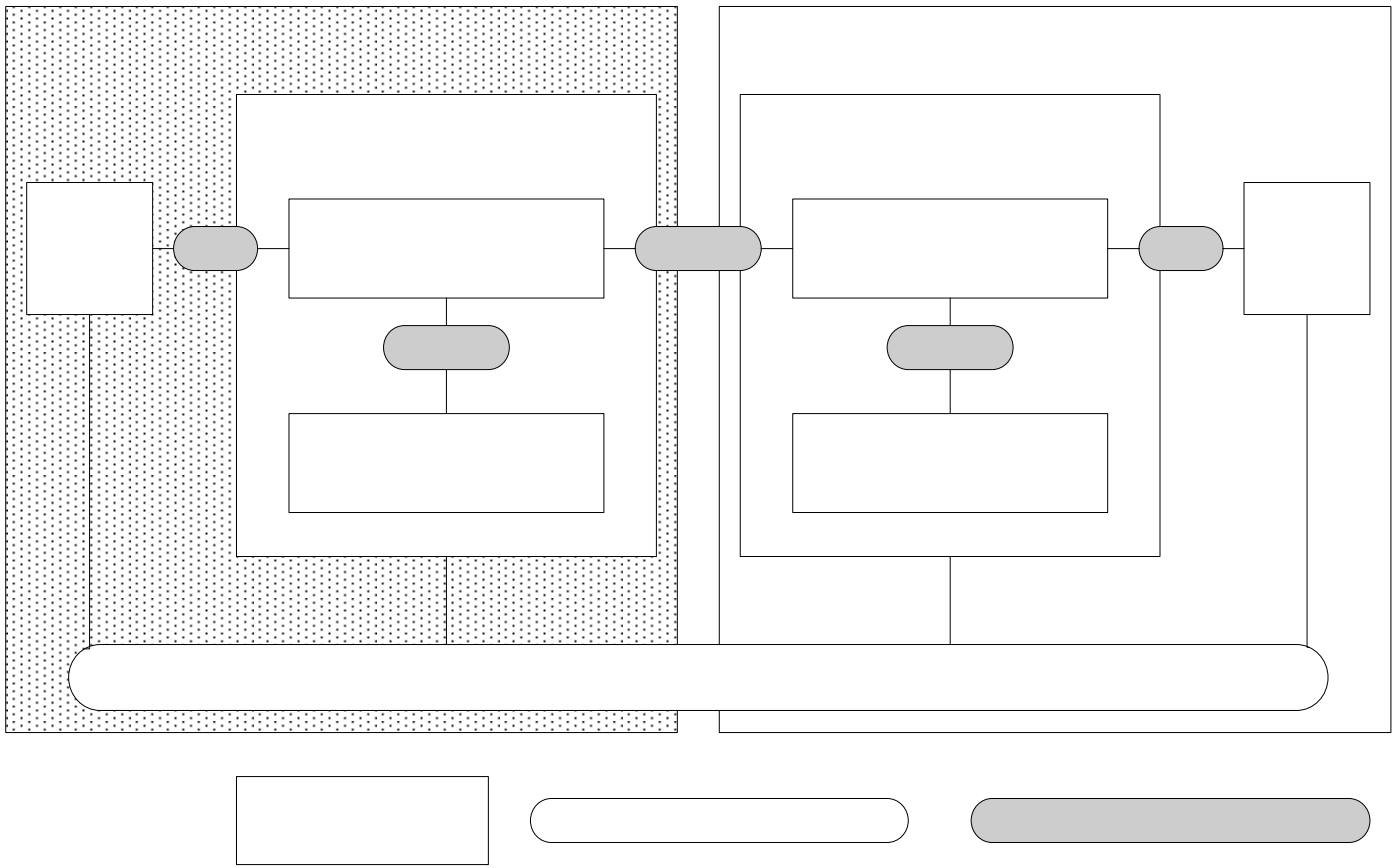
Case Study: Impromptu

- An ad-hoc peer-to-peer file sharing application for a workgroup
  - User-centered context
  - Users make decisions
  - Visualization facilitates making decisions
  - Perception, decision, and action
- Security Goals:
  - Make security visible
  - Ease security configuration

Impromptu UI



Impromptu Architecture



Secure Connector

```
<connectorType
  type="ConnectorType"
  id="SecureWebDAVConnector">
    <signature id="WebDAVClient">
    </signature>
    <signature id="WebDAVServer">
    </signature>
    <description>
        IP-based authentication
        Method-based authorization
    </description>
</connectorType>
```

Connecting Components

```
<component type="ProxyType" id="Local">
    <principal>Me</principal>
</component>
<component type="ProxyType" id="Remote">
    <principal>Other</principal>
</component>
<connector type="SecureWebDAVConnector"
  id="Impromptu_Impromptu">
    <interface signature="WebDAVClient“
      id="Remote"/>
    <interface signature="WebDAVServer"
      id="Local"/>
</connector>
```

Enhanced Secure Connectors

```
<connectorType id="DigestAuthenticationConnector">
</connectorType>
<connectorType id="WebXMLAuthorizationConnector">
</connectorType>
<connectorType id="WebDAVACLConnector">
</connectorType>
<connectorType id="SecureWebDAVConnector">
   <subArchitecture>
     <sequence>
        <connector type="DigestAuthenticationConnector"/>
        <connector type="WebXMLAuthorizationConnector"/>
        <connector type="WebDAVACLConnector"/>
     </sequence>
   </subArchitecture>
</connectorType>
```

Conclusion

- Background and Insight
  - Combine security and software architecture
  - Connector-centric
- Approach
  - Extend xADL with security constructs
  - Users (Principals), Privileges, Trusts, Contexts
- Case Study
  - Secure WebDAV Connector
- Future work
  - Language Semantics
  - Tool Support