Assessing SOA Performance

Kevin Foltz

June 23, 2010

Outline

- Background
- Objective
- Methodology
- Results
- Future Plans

Background – DoD

- Mandate to share information
- Move to one interoperable network across DoD
- Net Centric Enterprise Services (NCES)
  - A common set of services and applications to manage the network and help users locate and share information. Creates new capabilities and tools to tag data so it is useful, providing users with the capability to identify relevant information based on content. Allows users to freely exchange and collaborate on information.

Background – Web Technology

- Share Files
- Browse Static Content
- Browse/Create Dynamic Content
- Single Sign On – Web Services - SOA

Background – SOA

- Commercial Use of SOA
  - Amazon
  - Google
  - …
- Commercial Availability of SOA
  - Amazon
  - Google
  - …

Background – Security

- Formal security model
  - Confidentiality
  - Integrity
  - Availability
  - Authentication
  - Authorization
  - Auditing
  - Non-repudiation
  - Delegation

DoD Problem

- How to combine benefits of SOA with formal security model?
- How to assess its performance?

Objective

- Assess SOA infrastructure performance
- Assess SOA security service performance
- Assess web service performance
- Assess web application performance

- Latency vs. Throughput is primary concern

Methodology

- Identify test tool
  - Web browser requests
  - Web service invocation
  - Database queries
  - Others, as needed
- Integrate test tool into environment
- Run tests

Test Tool

- iTKO LISA test tool
  - Built-in web browser for browser interactions
  - SOAP functionality
    - Act as web service provider
    - Act as web service client
  - Inputs:
    - Test file – step-by-step instructions
    - Staging file – user load parameters
    - Data files – inputs, outputs
    - Code – for additional functionality

Test Tool Integration

- SOA integration is already done
- Security is more difficult
  - DoD PKI certificates for testing (simulate CAC)
  - Trust for test certificates
  - Packet capture tools on all machines
  - Tester accounts for all software components
  - XML Signature code integration

Test Execution

- Step 1: Isolated component testing
- Step 2: End-to-end testing
- Step 3: “Typical load” testing
- Step 4: Scalability testing

Step 1: Isolated Component Testing

- Security
  - Request Security Token
  - Validate Security Token
- Infrastructure
  - Access web application
  - Access each web service as WSC
- Data
  - Make database queries as a service would

Step 2: End-to-end Testing

- Generate flows for typical use cases
  - Request token, validate token, access web application
  - Make web application call, web app requests token, web application calls service, service validates token, service calls database, return results

Step 3: “Typical Load” Testing

- Combine flows from step 2 to simulate real-world distribution of requests
  - Web app request, then 0-10 web service requests

Step 4: Scalability Testing

- Combine Step 3 testing with component testing to identify component failures

Results

- Testing a changing system is difficult
- Identified memory leak problem with one component
- Performance depends largely on environment
- Test Tool Issues
  - Test step granularity (GET vs. POST)
  - Security code integration
  - Ease of use, distributed load, output data

Looking Forward

- Scale-Up
- Monitoring Tools
- Management Tools
- Virtualization
- Cloud

Questions