Redefining Static Analysis
A Standards Approach

Mike Oara
CTO, Hatha Systems
Software Analysis for Compliance

Compliance Assessment

Requires

Software Analysis

Option

Dynamic Analysis

Performed on executable programs built from that software system running on a real or virtual processor

Static Analysis

Performed without executing programs, but by employing complex analytical tools which investigate the actual source code of the system
### Dynamic Analysis

<table>
<thead>
<tr>
<th>Methods</th>
<th>Benefits</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspecting the user interfaces</td>
<td>No source code required</td>
<td>No certainty of code coverage</td>
</tr>
<tr>
<td>Performing multiple incident specific tests</td>
<td>A realistic view of the software system in action</td>
<td>Number of tests maybe overwhelming</td>
</tr>
<tr>
<td>Measuring performance</td>
<td>Performance data available</td>
<td>Difficult to replicate runtime environment</td>
</tr>
<tr>
<td>Checking memory allocation</td>
<td>Locking down system removing unnecessary access</td>
<td></td>
</tr>
<tr>
<td>Configuration Management</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Flight 501, which took place on June 4, 1996, was the first, and unsuccessful, test flight of the European Ariane 5 expendable launch system. Due to an error in the software design (inadequate protection from integer overflow), the rocket veered off its flight path 37 seconds after launch and was destroyed by its automated self-destruct system when high aerodynamic forces caused the core of the vehicle to disintegrate. It is one of the most infamous computer bugs in history. (Wikipedia)
# Static Analysis

<table>
<thead>
<tr>
<th>Methods</th>
<th>Benefits</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture diagrams</td>
<td>System understanding in breath and depth</td>
<td>No performance data</td>
</tr>
<tr>
<td>Data flows</td>
<td>Direct discovery of non-compliance</td>
<td>Some run-time conditions may not be captured</td>
</tr>
<tr>
<td>Control flows</td>
<td>Pinpointing errors in the code, architecture, process</td>
<td>Source code is required</td>
</tr>
<tr>
<td>Investigative queries</td>
<td>Proof of compliance</td>
<td></td>
</tr>
<tr>
<td>Software measurements</td>
<td>No “guesswork” testing</td>
<td></td>
</tr>
<tr>
<td>Pattern discovery</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Source code → Repository model → Comprehensive System Diagrams User Definable Queries Both comprehensive and Custom/specialized analysis tools*
**Definition**
A diagram of relationships between runtime elements of the application (programs, transactions, tables, files, queues, etc.)

**Compliance applications**
- Is the application properly layered?
- Are all data access methods done through an API layer?
- Which elements of the application work with confidential or secret information?
### Static Analysis: Data Flows

#### Definition

- Shows dependencies between variables

#### Compliance applications

- What data appears in a user interface? (HIPAA)
- Was data encrypted before being stored or transmitted?
Static Analysis: Control Flow

**Definition**
Shows flow dependencies between statements

**Compliance applications**
Is data validated before being processed? What conditions lead to a particular action? (example: approval of a request)
Static Analysis: Queries

Static analysis tools look at the code as raw data. They can therefore query and discover predefined patterns of concern for compliance.

Example

To guard against internal attacks, find if there is any place in the code where special logic is executed for a particular hard-coded account.

//ActionElement[@kind="Branch" and exists(-->Argument[@name like "]"CUST"\]) and exists(-->Argument:ValueElement)]

<table>
<thead>
<tr>
<th>Query</th>
<th>Compliance applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find mixed IO programs</td>
<td>Compliance to architecture standards. User interfaces should be separated from data access.</td>
</tr>
<tr>
<td>Find data validations</td>
<td>Is data correctly validated before being accepted to be processed?</td>
</tr>
<tr>
<td>Find coding patterns</td>
<td>Discover common code weaknesses</td>
</tr>
</tbody>
</table>

Query results can be immediately seen in the context of data flows, control flows or platform architecture!
Static Analysis: Offering evidence

Reports based on queries

11/24/2010

SporaEng - Weaknesses

<table>
<thead>
<tr>
<th>Function</th>
<th>Use of strcat()</th>
<th>Use of fgets()</th>
<th>Use of free()</th>
</tr>
</thead>
<tbody>
<tr>
<td>isdigit</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>process in rs field</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>teacher in cr report</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>select list helper</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>add to teacher collection</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>show comment</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>add_select_list</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>strtok</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>save_attr</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>terror</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>strcpy</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>null char matrix</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>isalnum</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mask select list item</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Activity log

This session

<table>
<thead>
<tr>
<th>Time</th>
<th>Action</th>
<th>Selected</th>
<th>Parameters</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>4:13 PM</td>
<td>build</td>
<td>SporaEng</td>
<td>action=rebuild, files=34.</td>
<td>errors=0, warnings=0</td>
</tr>
<tr>
<td>4:15 PM</td>
<td>search</td>
<td>SporaEng</td>
<td>query=&quot;/ActionElement{[@kind = 'Branch'] and exists(&gt;argument...</td>
<td>count=0</td>
</tr>
<tr>
<td>4:15 PM</td>
<td>search</td>
<td>SporaEng</td>
<td>query=&quot;/CableableUnlit[@name = 'fgets']&quot;&gt;calledby&quot;:</td>
<td>count=0</td>
</tr>
<tr>
<td>4:16 PM</td>
<td>search</td>
<td>SporaEng</td>
<td>query=&quot;/CableableUnlit[@name = 'fgets']&quot;&gt;calledby&quot;:</td>
<td>count=3</td>
</tr>
<tr>
<td>4:16 PM</td>
<td>search</td>
<td>SporaEng</td>
<td>query=&quot;/CableableUnlit[@name = 'strcat']&quot;&gt;calledby&quot;:</td>
<td>count=14</td>
</tr>
</tbody>
</table>
### Current Static Analysis Approach – Practical difficulties

<table>
<thead>
<tr>
<th>Issue</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large variety of languages and technologies</td>
<td>No static analysis tools cover all technologies. If many tools are used, results are hard to merge.</td>
</tr>
<tr>
<td>Large variety of regulatory requirements</td>
<td>No static analysis tool covers all requirements. Some may offer good data flow analysis, some good queries for weaknesses, some good architecture diagrams.</td>
</tr>
<tr>
<td>Large variety of regulatory authorities</td>
<td>Each regulatory authority may require a certain type of reporting</td>
</tr>
<tr>
<td>Flexible use of advancements in the technology and methodology of analysis</td>
<td>Tools and their approaches can quickly become obsolete; stovepiped solutions while some tools may improve more. How can you switch without sacrificing the current investments?</td>
</tr>
</tbody>
</table>

#### Solution

Use of standard models, commonly accepted terms, commonly accepted methodologies and reporting.
• **Knowledge Discovery Metamodel (KDM):** an ISO/OMG Standard providing ontology (a set of definitions) for system knowledge extraction and analysis. KDM provides a framework for the capture of code, platform and other software system characteristics. This further allows the extraction of data flows, control flows, architectures, business/operational rules, business/operational terms, and the derivation of business/operational process; the extraction can be delivered from source, binary, or byte code. Additionally the intermediate representation of the extraction is in executable models creating the possibility of simulation and code generation.
Standard models: SBVR

- **SBVR (Semantics of Business Vocabulary and Business Rules):** An ISO/OMG standard, this specification provides a structured process for formalizing, in natural language, the existing English language representation of compliance points. The standard enables the various compliance specifications (e.g. FISMA, HIPAA, SOX, FIPs, CWEs, etc) to be formalized reducing the room for interpretation from organization to organization when implementing the compliance and auditing requirements.
Business Process Modeling Notation (BPMN): an OMG standard delivering a modeling notation used to capture business/operational processes in support of system and organizational process simulation and analysis. It is used today to capture both human and IT system processes for the purposes of simulating environments both ‘as is’ and ‘to be’ for software modernization. This notation is compatible with KDM so that system extraction can be represented in BPMN for gap analysis of the current state of the system vs. what is thought to be the current state of the system – critical for modernization and compliance.
DR5  Recapture from internet the photo for the granularity
Dennis Rosynek, 11/26/2010
Data/Metadata Storage Standard (RDF): With the emergence of the standards noted above and the need for storing this information for analysis, a set of storage standards needed to be embraced. XMI, RDMBS, and RDF (Resource Description Framework) are the three formats that are compatible with these standards. RDF - perhaps the least known of them - is a W3C standard that is compatible with KDM and BPMN. There is a specific approach in the standard called RDF triple store which is currently being used in semantic web applications. The value of RDF is that it can manage large amounts of data and metadata which is critical for doing comprehensive static analysis.
Enterprise Software Systems Static Analysis Standards Mapping

Legend:
- **KDM**: Knowledge Discovery Metamodel ISO standard
- **RDF**: Resource Description Framework – W3C standard
- **BPMN**: Business Process Modeling Notation – OMG standard
- **SBVR**: Semantics of Business Vocabulary and Rules - ISO standard
Static Analysis & Standards
A Scenario

- Parsing
  - Cobol/CICS/DB2 (Tool A)
  - C/C++ (Tool B)
  - Java (Tool C)

- KDM Repository

- Analysis
  - Architecture (Tool D)
  - Common weaknesses (Tool E)

- Extraction
  - Business rules (Tool F)
  - Processes (Tool G)
Thank You
Mike Oara
mike.oara@hathasystems.com
919-931-9997