Security Challenges
A system designer faces several challenges when specifying security for distributed computing environments or migrating systems to a new execution platform.

Business stakeholders impose constraints due to cost, time-to-market requirements, productivity impact, customer satisfaction concerns, and the like. And users exercise power at the desktop over computing resources and data availability.

So, a system designer needs to understand requirements regarding protected resources (e.g., data), confidentiality, and integrity. And, a designer needs to predict the effect that security measures will have on other runtime quality attributes such as resource consumption, availability, and real-time performance.

After all, the resource costs associated with security can easily overload a system. Security processing can increase usage of processing power, bandwidth, battery (in embedded systems), and other resources.

Despite that, security is often studied only in isolation and late in the process. However, the SEI has developed model-based engineering (MBE) tools, methods, and analytical techniques to validate security according to flow-based approaches and standard security protocols such as Bell-LaPadula, Chinese Wall, and role-based access control.

Security Prediction with Less Cost, Less Risk, Increased Confidence
Security analysis using SEI MBE tools, methods, and techniques allows software validation by identifying data elements to be protected, components that should be allowed access to those elements, and appropriate communication channels. This analysis permits the designer to enforce security at the minimum level required, use sanitization, and map software architecture to hardware.

MBE also allows a designer to identify how security choices affect other quality attributes. For example, a designer can visualize and analyze, for battery-powered devices in embedded systems, the tradeoff between increased execution time and latency that supports the required security levels—to take advantage, for instance, of the multiple independent levels of security (MILS) paradigm.

<table>
<thead>
<tr>
<th>Security Analysis Concern</th>
<th>SEI MBE</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sanitization (i.e., controlled lowering of security levels)</td>
<td>✓</td>
<td>Provides metrics on the number of sanitized flows in a system</td>
</tr>
<tr>
<td>Security effectiveness applied using minimum security clearances</td>
<td>✓</td>
<td>Derives the minimum security clearance on components in the model (By pointing out differences between actual security clearances and the minimum security clearance required, a system designer can evaluate security effectiveness.)</td>
</tr>
<tr>
<td>Integration of security at multiple system levels</td>
<td>✓</td>
<td>Provides system-level solution by checking that secure information is associated with components that have appropriate security clearance and is communicated by secure connections</td>
</tr>
</tbody>
</table>

Read more
More Accurate Prediction of Security
Through Architecture Analysis using Model-Based Engineering Tools

The SEI MBE Toolkit
The SEI uses the Architecture Analysis & Design Language (AADL) to document a system architecture and provide a platform for multiple analyses.

The AADL, an international industry standard, supports multiple analyses from a single architectural model, enables modeling and analysis throughout the life cycle, and provides analysis of runtime behavior (what) rather than functional behavior (how).

Through its XML/XMI interchange format, the AADL supports model interchange and tool chaining. And, the SEI offers the freely available Open Source AADL Tool Environment (OSATE) set of analysis plug-ins. The OSATE security analysis plug-in checks the security levels and flow completeness of components.

The SEI has developed OSATE as a set of plug-ins for processing AADL models that includes:
- a syntax-sensitive text editor, with integrated error reporting
- a parser and semantic checker for textual AADL with conversion into AADL XML
- an unpaser for AADL XML to textual AADL conversion
- support for multi-enterprise development through a version control system interface

The AADL also can be used with
- UML state and process charts through its UML profile
- the SEI Architecture Tradeoff Analysis Method®, to drill into root causes and develop quantitative analysis
- assurance cases, to support claims made about the safety, security, or reliability of a system

The SEI has developed MBE (model-based engineering) tools that include:
- a parser and semantic checker for textual AADL
- an unpaser for AADL XML to textual AADL conversion
- support for multi-enterprise development through a version control system interface

The AADL also can be used with
- UML state and process charts through its UML profile
- the SEI Architecture Tradeoff Analysis Method®, to drill into root causes and develop quantitative analysis
- assurance cases, to support claims made about the safety, security, or reliability of a system

Prevent System Integration Problems and Simplify Life-Cycle Support
Modeling of system quality attributes is often done—when it is done—with low-fidelity software models and disjointed architectural specifications by various engineers using their own specialized notations.

These models are typically not maintained or documented throughout the life cycle, making it difficult to predict the impact of change on attributes that cut across system functionality. The unanticipated effects of design approaches or changes are discovered only late in the life cycle, when they are much more expensive to resolve.

Analysis of a system architecture model offers a better way to predict the behavior of quality attributes. The SEI approach to model-based engineering (MBE) allows analysis
- using a single architecture model
- early and often in the development life cycle or on an existing system architecture
- at different architecture refinement levels
- along diverse architectural aspects such as behavior and throughput

Integration is a major cost and risk in complex systems. System understanding is a major cost driver during system maintenance. Proper use of MBE tools can prevent system integration problems and simplify life-cycle support.

Put MBE to work on your projects quickly!
Register for training by the Software Engineering Institute. Go to www.sei.cmu.edu/training/p72.cfm.

1 One large defense contractor, for instance, blames human interpretation of the complexity involved with embedded systems for decreasing productivity to 6 or fewer lines of code per day.

The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.