CAERUS: Chronoscopic Assessment Engine for Recovering Undocumented Specifications

Adam Seitz, Adam Satar, Brian Burke, Lok Yan, Zachary Estrada

1 Rose-Hulman Institute of Technology, Terre Haute, IN USA
2 Air Force Research Laboratory, Rome, NY USA
Think Fortran, assembly language programming is boring and useless? Tell that to the NASA Voyager team

Ancient code jocks needed to keep probe alive

By Shaun Nichols in San Francisco 31 Oct 2015 at 12:03
Modernizing/Protecting Legacy Systems

Modern System ➔ Verifier ➔ Legacy System

Open the pod bay doors?
Undocumented Specification: Toy Example
We want to automate the task of finding timing sensitivities
Goal: a tool for uncovering timing sensitivities

• Automated: run with minimal user interaction
• Versatile: applicable to different target devices
• Extensible: system capabilities can be augmented
Chronoscopic Assessment Engine for Recovering Undocumented Specifications
User Interface

- Define Inputs/Outputs
- Define fixed signals
- Control Experiments

Output timing properties

Test Routine

- Mutate input **signals of interest** to perturb a suspected **sensitivity**
- Report acceptable timing variations
- Evaluate target device behavior
- Done?

Behavior model

Yes

No

Peripherals

- Play signals
- Record outputs

Evaluate target device behavior

Report acceptable timing variations

End of page
Example Test Routine: Button Duration

Button press

Response

Configurable duration to register button press
Example Test Routine: Button Duration

Min: 1 µs  Try: 62.5 ms  Try: 125 ms  Max: 250 ms

RST Button Response

Validate with Behavior Model
Example Test Routine: Button Duration

<table>
<thead>
<tr>
<th>Duration (ms)</th>
<th>Mean</th>
<th>StdDev</th>
<th>Min/Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.005</td>
<td>$2.985 \times 10^{-3}$</td>
<td>1.001/1.007</td>
</tr>
<tr>
<td>7</td>
<td>7.000</td>
<td>$6.569 \times 10^{-3}$</td>
<td>6.993/7.055</td>
</tr>
<tr>
<td>34</td>
<td>34.00</td>
<td>$8.413 \times 10^{-3}$</td>
<td>33.97/34.01</td>
</tr>
<tr>
<td>1 - HS</td>
<td>1.026</td>
<td>0</td>
<td>1.026/1.026</td>
</tr>
<tr>
<td>7 - HS</td>
<td>7.024</td>
<td>0</td>
<td>7.024/7.024</td>
</tr>
<tr>
<td>34 - HS</td>
<td>34.04</td>
<td>$1.194 \times 10^{-4}$</td>
<td>34.02/34.88</td>
</tr>
</tbody>
</table>

• HS = High Speed crystal oscillator (precise)
Current & Future Work
Security Applications: Fault Injection Attacks

• CAERUS as an embedded device fuzzer
• Clock glitching (e.g., instruction skipping)
• CAERUS is useful for tasks such as finding the right clock cycle, etc...
Going Forward

• Released as open-source under Mozilla Public License
• Stream-lining installation, set-up
• Currently have library support for RS232, looking to add CAN, J1939
• Analog to test other attacks (e.g., brownout, reset)
• Combine peripheral devices
Summary

• Legacy systems & timing sensitivity
• CAERUS architecture
• Minimum button duration example
• Security applications
• Source available on github: https://github.com/caerus-timing