

# *The Challenge of an Open Security Testing Methodology for Control Systems*

**Matthew Franz**  
**[franz@digitalbond.com](mailto:franz@digitalbond.com)**

# Overview

- ◆ **Background**
- ◆ **Current state of security testing**
- ◆ **Some assumptions**
- ◆ **Problem space**
- ◆ **Some strawman ideas**
- ◆ **Conclusions**

# My Background in Security Testing

- ◆ **Worked in a internal product security team for a large network vendor**
  - Tested dozens of different products, applications, and devices at various stages of the development lifecycle
  - Worked with QA teams to add security testing to their own processes
  - Contributed to several cross-functional security initiatives that didn't go anywhere
  - Was part of several small wins
    - Developed modular vulnerability assessment criteria
    - Developed standards toolset to be used across the entire product line (easy to use Linux LiveCD)
- ◆ **Currently do network and application testing as part of Digital Bond assessments**

# The Status Quo

- ◆ There are open methodologies for assessing operational networks (such as OSSTM for “pen-testing”) but nothing comparable for products, whether IT or SCADA.
- ◆ Big vendors *are* making investments in product security testing and application assessments -- are smaller vendors?
- ◆ As an asset owner or vendor, to develop your own security testing approach you must start from scratch
- ◆ Current practice is largely *art* vs. engineering

# Up Front Assumptions

- ◆ By “Security Testing” we mostly really mean *vulnerability* testing -- not functional testing of security (encryption, authentication, authorization features) in a product or solution
- ◆ The operational impact of control system vulnerability exploitation might be unique but the methodology to discover/exploit / test those vulnerabilities is not
- ◆ Security testing is not a panacea and cannot address the entire space of product security concerns
- ◆ There is some really “basic stuff” that can/should be done by both vendors and asset owner prior to “bringing in the consultants” or “shipping your gear to a test lab”

# What are the problems I see?

- ◆ **Difficult to compare *security testing products* and *security assessment services* without open criteria**
- ◆ **Confusion about what sorts of testing should be done:**
  - What is being tested?* (i.e component vs. system)
  - When?* (pre-ship, solution integration, deployment)
  - Why?* (misconfiguration, implementation flaw, known vs. unknown)
- ◆ **Assessment and Framework overload?!**
- ◆ **Raise the floor!**
  - Too many applications, devices, protocols still have issues with free/Open Source tools

# What might be in a ~~standard~~ open framework for security testing?

- ◆ A set of common definitions, or at least a mapping of terms if we can't agree on terms
- ◆ Clusters of discrete assessment activities (perhaps in a taxonomy) mapped to:
  - Types of vulnerabilities tested/assessed?
  - Who is the "user" (end users, vendor, integrator)
  - Where in the lifecycle the activity would be the most useful?
- ◆ A methodology for selection of test cases for different targets so that security tests can be integrated into existing test plans

# Which Tools and Why?

- ◆ **What percentage of the attack surface of your device, application, solution is actually “unique” to control systems?**
- ◆ **Just among free tools there are dozens of choices**  
Nessus, Nmap, ISIC, SPIKE, PROTOS, Amap, Nikto, Hydra, WebScarab, COMRaider
- ◆ **Need to move beyond tool based approach (run tool X and you are fine!) to a vulnerability criteria approach based on specific checks for different targets**

# Different Types of Targets Need Different Tests

## ◆ Interface

Single API, protocol implementation, physical interface, protocol stack, service, etc.

## ◆ Device/Appliance

Single hardware (Embedded or PC) platform with multiple interfaces  
Limited user management of underlying OS

## ◆ Application

Distributed across multiple  
May or may not use standards application components  
User typically (but not always) has to manage security of underlying OS

## ◆ System/Solution

Multiple devices and applications on a network  
Testing here looks most like a “pen-test” or network vulnerability assessment

# What types of vulnerabilities are we checking?

- ◆ **Known vulnerabilities in infrastructure components**
  - Operating System, Application
- ◆ **Robustness**
  - Flooding
  - Handling of Malformed Message
- ◆ **Application Misconfiguration flaws**
  - Security features
  - Default credentials
  - Backdoors
- ◆ **Network Misconfiguration**
  - Access Control Lists and Firewall Rules
  - MAC Filtering and Authentication
  -

# Challenges

## ◆ Easy Problems

- Diverse set of “users” (vendors, integrators, end users)
- Diverse technology base
- If it were easy (or really necessary?) then why hasn't it been done?

## ◆ Hard Problems

- The business case - *methodology is a competitive advantage and why do it if you can't charge for it?*
- Perceived and real risks - vendor exposure

# Conclusions

- ◆ **Regardless of private & public programs that do security testing of control systems, there is a need for various “users” (whether vendors, end users, or integrators) to do some testing themselves**
- ◆ **It is possible to carve out some basic assessment activities (call them tests, if you like) that are high impact and that can be easily implemented**
- ◆ **At a minimum “users” should initially focus on testing aspects of system they have control over**

# Next Steps

- ◆ **Is there really a need?**
- ◆ **Would anyone do testing themselves?**
- ◆ **Any role for PCSF?**
- ◆ **Where are related efforts?**
  - Control System Security Foundation
  - PCSRF
  - SANS “Procurement Language” Project?