Sec App Dev 2010

The View From the Giants’ Shoulders

Ken van Wyk
Ken van Wyk -- ken@krvw.com

Work Experience
- 20+ years in Information Security
  - Carnegie Mellon University CERT/CC Founder
  - U.S. Department of Defense CERT
  - SAIC, Para-Protect
  - President and Founder, KRvW Associates, LLC (http://www.krkw.com)

Security Work
- Technical lead on hundreds of commercial engagements since 1996, including
  - Application security assessments
  - Enterprise risk assessments
  - Secure network architecture
  - Security testing of enterprises and applications
- Author of two popular O’Reilly and Associates books
  - Incident Response: Planning and Management
  - Secure Coding: Principles and Practices

Credentials
- BS Lehigh University 1985, Mechanical Engineering

Personal Interests
- Travel, world cuisine, wine, mountain biking, zymurgy

Family (http://www.vanwyk.org/ken)
- Wife, two spectacularly spoiled basset hounds
Introductions

Please tell us a little about your
  – Software interests
  – Software dev technologies
  – Any specific topics you want to learn about this week?
Headlines
Why aren’t things improving?
Learn from history

We don’t pay enough attention to our failures
Consider other engineering disciplines
  – Study and learn from mistakes
  – Continuous improvement
Lack of knowledge

Developers tend to lack security knowledge
Security team tends to lack development knowledge
Not healthy
—“Us” and “them”
We’re overly trusting

We tend to have misplaced trust in our users
Sometimes users are malicious
Sometimes they don’t even try to be
Focus

Too much attention is paid to functional spec
Consider what can go wrong as well

– Most of what we care about in security is in the non-functional areas
Old school paradigms

Old school information security solutions don’t adequately protect the software
Consider IM, Skype, WiFi, VPNs
Testing isn’t working

Software testing does not adequately address security
Penetration testing is not sufficient
The Road Ahead

If that’s not enough, what should we do differently?
You’ll hear many answers to that this week
Let’s consider a few things first
What is “secure enough”?

Is it enough to stop the bad guys?
What about interfacing with the security team?
What other responsibilities do we have?
Case study: Biotech firm

Business servers crashing 1-2 times per day
Security personnel found a “ping of death” attack
  – Originating on a company PC
  – Which one? ???
  – Logs vacuous
Case study, cont’d

Company called in outside CSIRT to investigate
System logs told us almost nothing
Network data showed level 3 data
– Attacker spoofed IP
Case study, cont’d

Impact
– Downtime ~ 48 hours
– Costs ~ USD$10 M
– Not reported publicly
– Attacker found but never arrested

Completely avoidable
– Application logging
– Evidentiary support
Case study: Financial firm

CEO makes bold security statement on CNN
- Firewall/IDS alerting ensues

Company calls external CSIRT to investigate
Case study, cont’d

Logging sources
- Router (netflow)
- Firewall
- Web server
- All quiet in middleware
- Database transactions

None of it was useful
- Why?
Case study, cont’d

Impact
– No downtime
– Costs ~ USD$250k
– No attacks found
– Firewall bug fixed

Completely avoidable
– Cohesive logging
– Time synchronization
If it were simple...

It’s not just as simple as logging everything
– That often gives away too much data
– Seriously
Embedded systems too

Seen on a Boeing 747-400 while the system was being rebooted
–Due to an electronic fault
Not serious enough?

Seen on an ATM
Some customer cards caused a system fault...
Even the little mistakes count

Automated parking kiosk at IAD airport
Web sites too

CNN mobile portal
Mistakes remembered

The public sees these mistakes and remembers them
We’ve got to do better
Until then, this is not “engineering” in any sense of the word
So let’s learn from history

Consider some guiding principles
– Beyond Saltzer and Schröder

Lower level
– Address the OWASP Top-10, CWE 25, etc.
Input and output validation

Positive validation
– Proven safe, or else dangerous

Safely output mistrusted data
– Cause no harm in output environment

Always
Protect secrets

Sensitive info must be protected
- In transit and at rest
- Commensurate to value

Key management is everything
- Except for all the rest
Anticipate and handle errors

Assume things will go wrong
Anticipate
Use the toddler-in-traffic metaphor
Protect session and state

When working in non-stateful medium
- It’s up to the application
State mechanisms must be protected
- Confidential
- Random
- Unforgeable
- Tamper-evident
Authenticate your users

Who are you?
– Prove it
Commensurate to value
Feasibility matters
Control access

One simple question
- Are you authorized to do what you’re requesting?

Every sensitive function, data, etc.
- Needs to be designed in
Integrate into the enterprise

Consider what the CSIRT will need from your app
- Block the attacks
- Log what happened
- Take evasive action
  • Driven by policy
Just to name a few...
Contact details

Kenneth R. van Wyk
KRvW Associates, LLC
Ken@KRvW.com
http://www.KRvW.com