

# PREVENTING ZERO-DAY ATTACKS IN MOBILE DEVICES

Ira Winkler  
Codonomicon

Security in  
knowledge



# — Zero Day Attacks

- ▶ Zero day attacks are rising in prominence
- ▶ They tend to be behind the most devastating attacks these days
- ▶ Generally used by very high end criminals and nation states
- ▶ You usually don't know about the attack unless there are other indicators

# — Zero Day Examples

## ▶ Stuxnet

- ▶ Combined 4 zero day attacks that we know of in a specific Siemens SCADA software system
- ▶ Supposedly delivered via USB drive
- ▶ Caused centrifuge systems to essentially self destruct to set back Iranian nuclear capabilities
- ▶ Only detected because of a lack of a kill switch once it left the Iranian facilities

# — Zero Day Examples

- ▶ RSA hack
  - ▶ Spear phishing message retrieved from spam filter of RSA employee
  - ▶ Exploited an IE 6 vulnerability
  - ▶ Established a foothold on the network, which was then used to compromise the RSA network
  - ▶ SecurID source code stolen
  - ▶ Attempts to brute force remote access system of defense contractors via a new zero day likely found via source code review

# What exactly is a Zero Day attack?



# Only Two Ways to Hack a Computer

- ▶ Despite all of the technologies out there, there are only two fundamental ways to hack computers
- ▶ The basics can be applied to any computer system or network
- ▶ You might contend that there are more basics, but I promise you that I can force fit it into the only two

# — Take Advantage of Configuration Problems

- ▶ How an admin configures a system
  - ▶ Bad configurations, architectures, file shares, default privileges, improper patching, etc
- ▶ How a user uses or maintains the system
  - ▶ Violates privileges, bad passwords, vulnerable to social engineering attacks, bypasses security processes, goes to malicious websites, etc

# Take Advantage of Vulnerabilities in the Software

- ▶ All Software has bugs
- ▶ Some bugs create elevated privileges or cause information leaks
- ▶ Those are security vulnerabilities
- ▶ Nothing a user or admin can do, except stop the service



# — Typical Vulnerability Lifecycle

- ▶ Developer writes software with vulnerability
- ▶ Begins to get used
- ▶ Vulnerability is found by accident or on purpose
- ▶ Developer hopefully finds out about it and issues a patch
- ▶ Users then implement the patch

# Where Problems Occur

- ▶ Vendors aren't alerted about the problem, so they can't fix it
  - ▶ Zero Day Vulnerability
- ▶ The more common problem: Users don't implement the fix
  - ▶ Ironically the developers look bad, because nobody wants to "blame the user"
- ▶ Fix cannot readily be implemented
  - ▶ Hopefully you don't have a person who wants credit or full disclosure
  - ▶ Embedded devices require physical upgrades

# — Who and Why?

- ▶ Enthusiasts
  - ▶ For fun
- ▶ Hackers
  - ▶ For ego
- ▶ Black market
  - ▶ For profit
- ▶ Criminals
  - ▶ For profit

# — Who and Why?

- ▶ Security researchers?
  - ▶ For marketing
  - ▶ For profit
  - ▶ For good
- ▶ High end criminals
  - ▶ For their own intelligence purposes
  - ▶ For major profits
- ▶ Nation states
  - ▶ CNO, CNI, CNE

# — Embedded Devices

- ▶ Embedded devices are usually firmware and you can't easily make changes
- ▶ Traditionally very simple and basic components
  - ▶ Little functionality
- ▶ They are usually there for the lifespan of the product and cannot be swapped out
- ▶ Unfortunately, devices are becoming more complicated, so there will be more bugs

# — Mobile Devices

- ▶ Mobile devices are pretty much ubiquitous now
- ▶ More plentiful than regular computers in the first world
- ▶ There is no security software suite in common use to mitigate even widely known vulnerabilities
- ▶ Rarely do people perform updates

# Beginning to See Problems

- ▶ Many problems with Android phones
- ▶ Vulnerabilities found with iOS and all other operating systems
- ▶ Vulnerabilities can be for the apps as well as the OS
- ▶ Beginning to be used for critical applications
- ▶ Used for financial applications
- ▶ Proliferation is too big of a target to ignore

# — Developers Not in a Position to Fix Process

- ▶ Developers cannot address how users or admins do things
- ▶ They can make things easier to maintain, and should, but there is a limit
- ▶ Can limit user options



# — Zero Days are Expensive

- ▶ Reputation alone can be critical
- ▶ Exponential increase in cost to fix a bug in each phase it's detected
- ▶ Expensive to test
- ▶ Expensive to push out
- ▶ Potential liability issues for some sectors

# Proactive Testing for Prevention

- ▶ The best way to prevent zero days is to proactively find them early in the lifecycle
- ▶ As much testing as possible should be implemented as early in the development process as possible
- ▶ Automation is cheaper and more effective
- ▶ Fuzz testing is extremely effective in rapidly identifying zero day vulnerabilities

# — What is Fuzzing?

- ▶ Black box testing
- ▶ Tests as many condition as possible
- ▶ Throws as much data as it can at the application
- ▶ Causes even security software to fail in seconds
- ▶ Iterative testing in nature
- ▶ This is how the “bad guys” will come after you

# — Make Fuzzing Value Known

- ▶ The fact of the matter is that many people don't understand the vulnerability lifecycle and how mobile devices are wide open, and will remain wide open
- ▶ Fuzzing is significantly more cost effective than replacing millions of smart phones
- ▶ Businesses have to acknowledge that mobile devices are prime targets, especially those widely distributed

# BACK DOOR ATTACKS



# — Helper App Attacks

- ▶ Many mobile attacks are at an application level beyond the control of most developers
  - ▶ iPhones have some protections
- ▶ NFC and other base protocols can call a helper app
  - ▶ Browser, readers, viewers, etc.
- ▶ Helper apps can be very vulnerable
- ▶ Secure application triggers an insecure one

# — Don't Ignore Configurations

- ▶ Default configurations to use helper applications
- ▶ Default configurations to enable minimal application permissions
- ▶ Only enable minimally required services
  - ▶ The more apps running, the more insecure
  - ▶ If an app/service isn't running, it can't be exploited
    - ▶ Bluetooth for example
- ▶ Never rely on users not to jailbreak phones
- ▶ Never assume a user will always leave settings strong

**DON'T FORGET  
ABOUT  
AWARENESS**





# Good Practices Stop Zero Days

- ▶ They don't stop them from existing, but they stop them from being executed
- ▶ If a user doesn't open an attachment, it can't exploit the device
  - ▶ RSA hack
- ▶ If a user doesn't download a random app, it can't exploit the device
- ▶ Research on security awareness critical success factors
  - ▶ [samantha@securementem.com](mailto:samantha@securementem.com)

# Awareness for Developers

- ▶ Developers write bad software and don't know it
- ▶ Many aren't familiar with the concept of buffer overflows
- ▶ Most don't know what an SQL injection attack is
- ▶ If they did we wouldn't have most of the bad software problems

# Summary

- ▶ Zero day exploits are coming to mobile devices
- ▶ Ensure that you have an appropriate development process to reduce the existence of easily discoverable vulnerabilities
- ▶ Fuzz testing is a very efficient step rapidly finding such vulnerabilities
- ▶ Ensure that proper configurations are implemented
- ▶ Create an awareness process to instill users with good security behaviors

# — For More Information

Ira Winkler, CISSP

[ira@securementem.com](mailto:ira@securementem.com)

+1-410-544-3435

<http://www.facebook.com/ira.winkler>

@irawinkler

<http://www.linkedin.com/in/irawinkler>

**RSA<sup>®</sup>CONFERENCE  
ASIA PACIFIC 2013**

