Server & Storage System Test, IBM Systems

Test Engineers and Hackers – Common Threads

A Look at Technical and Non-technical Skill Sets Shared by Both Groups

Christopher V. DeRobertis
Senior Technical Staff Member (STSM)

2015.06.15

Agenda

- About the Presenter
- Definitions
- Common Threads
- Take-aways
- Q&A (time permitting)

Disclaimers

- This presentation does not condone nor suggest the illegal, unauthorized, or unethical use of computing and/or information technology resources. Refer to the Notices page for additional disclaimers and legal information
- Presentation content is based on the author's analysis and experience
- Views and opinions are those of the author, not the author's employer
- Presentation deck does not contain speaker notes

About the Presenter

- 24+ years of information technology experience
  - Subject matter expert: Security (IT, data, cyber)
  - 14+ years of security experience in the areas of architecture, design, development, test
  - Technical leader: Software test engineering, enterprise systems & solutions testing
  - Current focus areas: Ethical hacking & penetration testing as a quality assurance practice, mobile security, cloud security, privacy, forensics, security intelligence
- Intellectual collateral creator
  - Technical white papers, essays, guides, etc.
  - Published & pending patents
  - Training & education
- Systems Thinker & T-Shaped Engineer

Not Covered

- Defining different product development test phases or types
  - E.g. Unit test, function test, system test, artistic, static, dynamic, fuzz, etc.
- Defining test specializations
  - E.g. Ethical hacking, penetration, Java, performance, operating system, middleware, REST, *aaS, etc.
- Development life cycle types
  - E.g. Agile, DevOps, waterfall, V-Model, etc.
- Debating *testing* versus *quality assurance*
- Defining hacking/hacker specializations
  - *ware, HTTP/web, SQL/LDAP/* injection, etc.
- Ethics, legalities, & psychology of hacking
- Differences between testers & hackers
  - And there are differences, especially in the domains of ethics, legal, business

Definitions

Product	Code, component, app, module, library, script, appliance (hardware or software), device, system, resource, etc.
Target	Product under test or focus of attack; the product being hacked

Definitions

Tester (a.k.a. Test Engineer)	Nutshell: Assures product meets specified requirements Uses manual and/or automated tests, testing techniques, and tools to surface defects and to "detect differences between existing and required conditions."[o] A tester "exercises and validates one or more assessments objects [(e.g., host, system, network, procedure, function, operation, feature, capability)] under specified conditions to compare actual and expected behaviors."[1]
Hacker	Nutshell: Bad actors/"the bad guys", cyber criminals, unauthorized users, etc. "An unauthorized user who attempts to or gains access to an information system"[2] through subversion, circumvention, brute forcing, exploiting types of vulnerabilities, or sabotaging of protection mechanisms.[3] A hacker can also be an *adversary*: "An individual, group, organization, or government that conducts or has the intent to conduct detrimental activities."[2]

[o] testing, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std 610.121990 (Revision and redesignation of IEEE Std 792-1983)
[1] testing, *Technical Guide to Information Security Testing and Assessment*, NIST, U.S. Dept. of Commerce
[2] hacker, adversary: *A Glossary of Common Cybersecurity Terminology*, National Initiative for Cybersecurity Careers and Studies (NICCS), US-CERT
[3] Based on cracker, *IBM Terminology*, IBM

Definitions

Testing	The “process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.”[1]
Hacking	“The unauthorized access, use, or manipulation of computing resources.”[2]

[1] *Standard Glossary of Terms Used in Software Testing* (Version 3.01), International Software Testing Qualifications Board (ISTQB)
[2] http://www.hhs.gov/ohrp/sachrp/mtgings/2012%20Jul%20Mtg/internetworkingglossary.pdf

Common Threads: Traits & Characteristics

Trait / Characteristic	Brief Description
Enjoys "breaking things" & making it fail	Likes a good challenge – especially when told "it's bullet proof" or "unhackable"
Curious	Why did it do that? Where does it store data? What's in the log? What about...?
Tenacious	Pursuer of breadcrumbs & anomalies; collects, then connects the dots
Adaptive	Finds new paths among seemingly dissimilar things; a dead end just means it's time to tunnel
Outsider	Comes at things with "fresh eyes" – didn't write the code/make the product ; it's just a target (not their "baby")
Skeptical	Explores, proves, & experiences things first-hand – doesn't accept "just trust me" as an answer (closely related to "breaking things")
Thinks Outside the Box	Uses things in ways never anticipated; connoisseurs of odd angles; proponents of putting large square pegs into small round holes; weavers of *what ifs*

Table has intersections with the Debugging Rules from the book *Debugging: The 9 Indispensable Rules for Finding Even the Most Elusive Software and Hardware Problems* (@ Amazon) and the book, *Software Testing Techniques – Finding the Defects that Matter* (@ Amazon)

Common Threads: Hard Skills[1]

Skill	Reason/Value
Programming, Test Development, & Tools	Automation, efficiency, scaling, interfacing, exploration, regression/re-use/repeatability
Debugging & Tracing	Learn work flows & data flows; figure out what went wrong in order to report / exploit
Managing	Execution, progress, results, next steps
Learning	Technologies, methods, techniques, tools, practices, etc. – staying relevant; fresh ideas * *Learning how to learn* is *a skill*
Processing Technical Documentation	Read & use manuals, guides, white papers, etc. to "go one level deeper"

[1] Technical skill sets; ability to perform specific functional tasks or activities; domain specific knowledge/expertise (adaption of definition at TechTarget.com)

Common Threads: Target Contexts

Target	Scope of Knowledge (Testers & Hackers)
White Box	Externals & internals (inside-out, outside-in) • Published interfaces, documentation • Design specifications • Source code • "Live specimens"
Black Box	Externals only (outside-in) • Published interfaces, documentation • Speculate based on conformance to standards, use of embedded/reused technology • Apply methods, techniques, attacks from similar or related technologies • Observe & adapt
Gray Box	Externals & some knowledge of internals • Hybrid of black box & limited white box

Bottom-line: Testers & hackers will examine what can be inspected, probe/glean otherwise

Black/white/gray boxes adapted from the article, *Black Box Security Testing Tools*, Build Security In, US-CERT, Department of Homeland Security

Common Threads: Basic List

➲ Things to Target/Vet/Do/Go After

- Programming flaws
- Design flaws
- Generate errors/failures
- Partial or misinterpreted implementations of technical standards
- “Improvements” upon technical standards
- Assumptions about the environment
- Installation (deployment) assumptions
- Seeing if best practices have been followed
- Overloaded interfaces
- Release notes (for new, changed, removed)
- Keep track of things (know where you've been & what you've done)
- Methodical, incremental, systematic changes
  - Try easy & obvious first, then iterate on difficulty, complexity
  - "Big Bang" things are noisy, messy, noticeable

Common Threads: How

➲ By-passing & Circumventing

- Assuming multiple user interfaces for a product…
  - GUI is stubborn? Try the CLI.
  - CLI is stubborn? Try REST.
  - REST is stubborn? Try the panel.
  - Panel is stubborn? Read the docs for other avenues

- Substitute one ingress/egress for another
  - Operating systems, appliances, devices, etc. can have thousands of command line programs, GUIs, user panels, etc. (doors, windows, hatches, etc.)
  - Poorly configured or weakly-secured services (potential doors)
  - Generally accessible privileged operations (potential tools or "keys")
  - Permitted network traffic, i.e., ports & protocols (piggy-back channels)

Common Threads: How

➲ I/O, Protocol, & Data Tampering

Via user interfaces, run-time configuration files, initialization files, data files & data base entries, credentials (user, device, system), environment variables, etc.

- Boundary conditions
- Garbage (junk) input & malformed inputs
- Zero length files
- Excessively large files
- Pointing/linking to resources "elsewhere"
- Non existent locations (objects, references, etc.)
- One bit, byte, character, number deltas
- Duplicates

Common Threads: How

➲ Defect Clusters[o]

Where there's one problem, others probably exist

- New or changed features/functions
- Deprecated or removed features/functions
- Documented limitations or restrictions
- Undocumented interfaces
  - What? You think they can't be found? That's a good-one!
- 80/20 Rule (Pareto's Principle), *but* what's lurking in the 20/**80**?

[o] Adaptation of "Defect clustering," principle #4, from the International Software Testing Qualifications Board's *Seven Principles of Testing*. Refer to the article "Fundamentals of Testing" (*Maximo Quality and Test Community*, IBM developerWorks), which discusses the ISTQB's seven principles.

Common Threads: How

➲ Information Probing

- Dumps, logs, traces, temporary storage, etc.
- Running in verbose or debug mode
- Run time debuggers
- Packet sniffers
- Browser plug-in inspectors (e.g. cookies, HTTP headers)
- Triggering error/exception conditions

Common Threads: How

➲ Using Tools & Toolboxes

- Commercial, off-the-shelf (COTS)
- Open source
- Custom/purpose-built
- Frameworks

Common Threads: How

➲ Pesticide Paradox (ISTQB Testing Principle #5º)

"… test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to find potentially more defects"º

- Add/revise tests to reflect changes
  - Ensure freshness
  - Archive obsolete
- Apply new techniques, new methods, new thinking
- Coverage for new threats, risks, attack vectors, opportunities
- Apply lessons learned

- "Modern hackers are advancing their techniques more rapidly than organizations can respond" [1]
- "Like good health, cybersecurity is never going to be a 'solved problem.' Attacks coevolve with defenses and in ways to disrupt each new task that is entrusted to our networked systems" [2]
- "Hackers are constantly advancing the ball … While banks are constantly buttressing their information with better protection, criminals are always looking for weaknesses, gaps and workarounds … You build it, they build something better, you build a response to that and then they build something different" [3]

º Text excerpted from "Fundamentals of Testing" (*Maximo Quality and Test Community*, IBM developerWorks), which discusses the International Software Testing Qualifications Board's *Seven Principles of Testing*

[1] *Proofpoint's CEO: Hackers Are Winning*, Executive Sessions, BankInfo Security

[2] "Blueprint for a Science of Cybersecurity," article, NSA

[3] *Reputational risk and IT in the banking industry*, Findings from the 2012 IBM Global Reputational Risk and IT Study, IBM

Real-world Hacker Advice

"Development simply cannot focus as much attention on the rarely taken code-paths [so these] are the areas to focus on. …There are bugs in there

…

Use Boundary Testing to push variables to the limit. Try exercising remote code-paths. Cause errors, but from odd angles. Try and cause a messy error handling condition. Use Exploratory Testing to find bugs in the area of already known bugs.

…

You will be finding bugs in no time."

From the article, "Finding and Exploiting Bugs," *2600 The Hacker Quarterly* (Spring 2000)

Text emphasis by presentation author

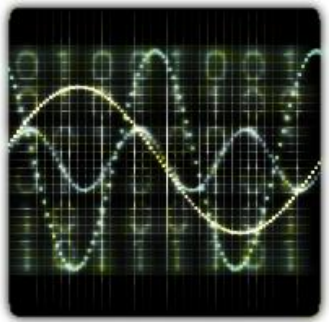
Take-aways

- Testers and hackers have common threads & similarities:
  - Curiosity, tenacity, and creativity traits / characteristics
  - Technical and non-technical skills
- Testers are clever – clever *engineers*
- Show testers some love – you'll thank them later

Back-up Materials

Note: Links that appear earlier in the presentation are not repeated in the back-up content.

Systems Thinking & T-Shaped



Complexity requires systems thinking[1]

Systems Thinking

- Understands the 'big picture' & the "process of understanding how things influence one another within the whole"[1]
- Interconnections, intersections, & interactions of: technologies, business, operations, legal, workflows, processes, etc.
- First underlying principle of DevOps[2]

T-shaped people: skills for the future of business[3]

T-Shaped

- Horizontal: Understanding of a broad range of related disciplines (business, technical, social)
- Vertical: Deep skills & subject matter expertise in a specialty area
- Technical *& business* skills

[1] *Smarter Cities for a Sustainable Future (Report based on proceedings at IBM Start)*, The Bathwick Group (report hosted by IBM)
[2] *DevOps distilled, Part 1: The three underlying principles*, IBM developerWorks
[3] *The Future of Business*, IBM Thought Leadership paper

- Systems Thinking and T-Shaped are important to anyone that wants to enter the field of security/cybersecurity/forensics.
  - Systems Thinking and T custom images by C. DeRobertis.

Systems Thinking & T-Shaped

Systems Thinking

- Systems Thinking: The "Core" Core Competency for BPM (Greystone Group, report hosted at IBM)
- Mental Model Musings (systems-thinking.org)

T-Shaped

- IBM's Role in Creating the Workforce of the Future (Beyond IT, report hosted at IBM)
- IBM Academic Initiative: Skills for a Smarter Planet (IBM)
- Are You an "I" or a "T"? (Forbes)
- T-shaped Skills and Swarming Make for Flexible Scrum and Agile Teams (scrumexpert.com)
- Education for a Smarter Planet success story: T-shaped (IBM)

IBM Security Resources

- IBM Security (main page)
- IBM Security Resource Center
- IBM X-Force Resources
- IBM Security on Twitter (@IBMSecurity)
- Security Intelligence (by IBM)
- Security Intelligence (Facebook)
- Official IBM Security Channel (YouTube)
- IBM 2014 Cyber Security Index (infographic, report)
- 2014 Cost of Data Breach Study: Global Analysis (sponsored by IBM)
- IBM Data Breach Statistics
- IBM Data Breach Risk Calculator (infographic), IBM Data Breach Risk Calculator (site)
- IBM Security Redbooks
- IBM Secure Engineering Portal
- IBM IT Services Literature (white papers tab has links to other IBM 2014 Cyber Security Intelligence Indices)
- IBM Academic Initiative, *Security and information assurance*
- Surviving The Technical Security Skills Crisis (IBM white paper)
- Cybersecurity education for the next generation (IBM white paper)
- Staying Ahead in the Cyber Security Game – What Matters Now (ebook co-written by IBM)
- Cyber Security: Protecting the Public Sector

Notices

Statement of Good Security Practices: IT system security involves protecting systems and information through prevention, detection and response to improper access from within and outside your enterprise. Improper access can result in information being altered, destroyed or misappropriated or can result in damage to or misuse of your systems, including to attack others. No IT system or product should be considered completely secure and no single product or security measure can be completely effective in preventing improper access. IBM systems and products are designed to be part of a comprehensive security approach, which will necessarily involve additional operational procedures, and may require other systems, products or services to be most effective. IBM DOES NOT WARRANT THAT SYSTEMS AND PRODUCTS ARE IMMUNE FROM THE MALICIOUS OR ILLEGAL CONDUCT OF ANY PARTY.

References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results. IBM, the IBM logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

Information concerning non-IBM products or resources was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. IBM has not tested these products or resources and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products or resources. Questions on the capability of non-IBM products or resources should be addressed to the supplier of those products.

All links were valid at the time of original inclusion in the presentation. Access to some resources may require registration to obtain.

End of Presentation

