Make It Count

Progressing through Pentesting

Bálint Varga-Perke
Silent Signal

HEK.SI 2017

About

- Silent Signal co-founder
  - Focus on pentest/ethical hacking
  - Training
  - Consulting
- Pentester
  - Finance
  - Healthcare
  - Software development
- Vulnerability whisperer
  - Enterprise software
  - Security software
- News aggregator and local know-it-all
  - @buherator

Why?

- Everything is loud from *cyber*
  - Serious challenges
  - Lots of FUD
- Growing demand for pentests
- New players on both sides
  - Misunderstandings
  - Suboptimal outcomes

Misunderstandings

Penetration testing

- Does not reveal root causes
- Does not offer solutions
  - Client left alone with the report
- Testers can’t build
  - So they don’t understand their targets

Real Limitations

- Limited (unrealistic) scoping
- Limited interaction
- Point-in-time results
- Controversial risk assessment
- Shallow results

Experience

Pentests is a tool

- … that can contribute to significant developments
- … as part of a larger strategy

Improvement requires in-house effort

It’s better to start early

PREPARATION

SILENT SIGNAL
VÉSZJELZÉS HELYETT...

WWW.SILENTSIGNAL.HU

What to test?

- Do you have an asset inventory?
- For hardware *and* software?
- Is it up to date?
- Is it updated automatically?

„Why are we successful? We put the time in to know that network. We put the time in to know it better than the people who designed it, and the people who are securing it”

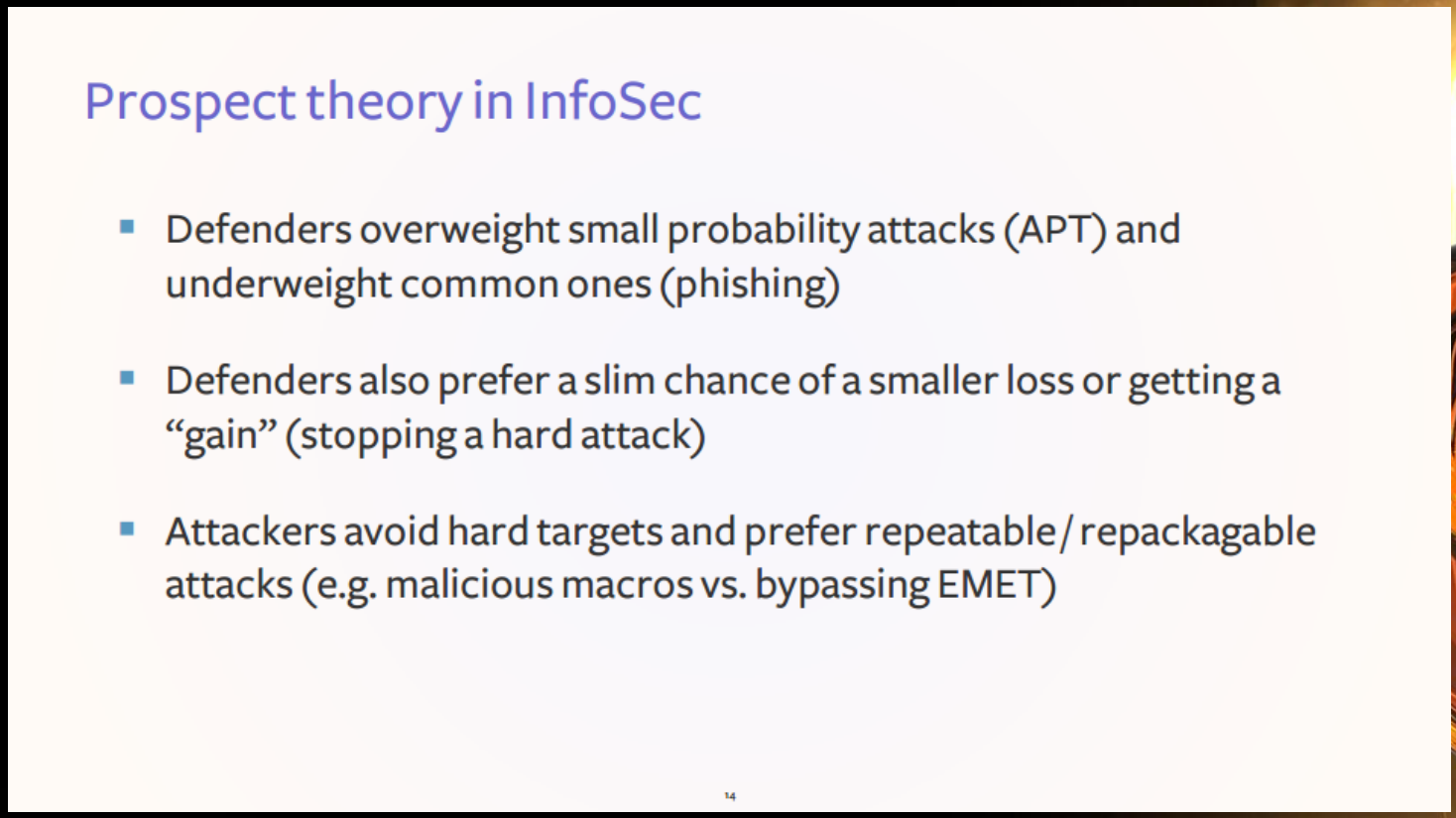
- Rob Joyce, NSA TAO Chief (USENIX Enigma ’16)

What to test?

- What data is critical to my business?
  - Data classification
- How can my data be used to cause harm?
- Who are my adversaries?

<u>You'll need Threat Modeling</u>

We're not great at risks



Kelly Shortridge – The Art of Explanation (Hacktivity 2017)

Tendencies

- Phishing
  - Client-side aka. Inside your FW
  - No 0-days
  - *Workstation hardening + Internal tests*
- High risk webapp attacks
  - Not XSS, Clickjacking, …
  - *Webapp tests **and** sane risk assessment*
- Password mismanagement
  - Your password policy is probably harmful
  - *Internal tests* again

Examples of bad scoping

- „Firewall testing”
- Webapps not handling sensitive data
- Cloud service instance (e.g. Office365)
- Assessment of inaccessible interfaces
- Phishing simulation - Management excluded
- Static code analysis only

How to test?

- Black-, gray-, pink-… boxes
- More information: Better results ☺
  - Better root cause analysis
  - Better coverage
  - More efficient execution
- More information: Higher costs?
  - Spare on recon
    (e.g. provide unlinked app URL's)
  - Marginal utility can deviate greatly
  - Need to find balance

Timeframe

- First class results rely on research
  - Testers should be passionate
- Research takes time
  - Replicating parts of your systems in lab
  - Creating tools for analysis and exploitation
- Test window > Project work hours

Watch me!

EXECUTION

Execution

Execution should be about:

- Collecting data
  - Recognizing attack patterns
  - Recognizing weak links
- Testing response
  - Technology
  - People
  - Processes

Execution

- Who knows about the test?
  - Is being stealth a requirement?
- Whitelist or not to whitelist?
  - WAF, FW/IP filter …
  - Middle ground: Non-blocking alerts
- Measure efficiency
  - Defence systems
  - Staff alertness
  - Communication

Defence Systems

- Which systems should alert during the test?
- Do they?
- Is the alert recognized?
  - Not lost in FPs
  - Escalated correctly
- Is the alert *actionable?*
  - „DC is portscanning” is actionable
  - „Public webserver is portscanned” is not

Defense systems

- Blinky boxes are mostly effective if attackers don't know about them
- Can you easily move around these systems in your network?
  - E.g. to collect data from pentest scope only?
- Change monitoring levels dynamically
  - Keep alerts meaningful + managable
  - Attacker uncertainity!

(Thought) Exercise #1

Your …. will not function from next week

- Anti-Virus system
- DLP software
- Fancy APT box
- Perimeter firewalls
- (…your idea here…)

What is your plan?

Getting better

FEEDBACK

SILENT SIGNAL
VÉSZJELZÉS HELYETT...

WWW.SILENTSIGNAL.HU

Cleanup

- In many cases we don’t even know about some artifacts
- In many cases we simply can’t delete
  - Left-over artifacts should be documented!
- If we could 0wn it so could others
  - „Do these sysadmins really use Meterpreter?”

You should prepare for cleanup and even…

(Thought) Exercise #2

What if all systems in scope would be compromised?

- You can’t fix revealed issues instantly
- Systems may have been compromised before the test!

What is your Incident response plan?

Risks

- Risk depends on numerous factors
- Testers usually have to guess several factors
  - Typically business impact factors
  - Preliminary info about critical data can help
- Reported risks are not cut to stone
- Demonstrating exploitability matters
- Final risk assessment should be done internally!

Slide about CVSS, OWASP RRM, etc.

Stop Fetishizing Bugs!

SILENT SIGNAL
VÉSZJELZÉS HELYETT...

WWW.SILENTSIGNAL.HU

Bug Fetish

- „Writing RE filters for n*100 XSS”
  - Meanwhile several RCEs, design issues...
- „We patched all systems”
  - After *last years* pentest
- „Hardened our crypto”
  - Just to disable the hardening in release
- Spending weeks on „checkbox failures”
  - E.g.: TLS issues, Clickjacking, HttpOnly …
  - Easy to test + Is everywhere => Overreaction

Stop Fetishizing Bugs!

- Fixing individual bugs doesn't really matter
  - You can't fix them all
  - You'll introduce new ones
- Fixing the wrong bugs wastes your resources
- Aim for *bug classes*
- Look for technical *guarantees*
- Good pentesters should provide
  - Sane risk assessment
  - Sustainable guidance

Fixing right

- Discard the unimportant
- Input validation shouldn’t be a humans job
  - Your framework should handle 99%
  - Choice of technology *does* matter
- Procedural guarantees
  - Patch management
  - Auto-updates
  - Automated static analysis (e.g. repoguard)
- Sandboxing
  - OS level
  - Network level

(Thought) Exercise #3

How easy it is to introduce …. into your production codebase?

- SQL injection
- Command injection
- File inclusion
- Cross-Site Scripting
- (… your idea here …)

How could you make it harder?

Takeaways

- Preparation
  - Asset review
  - Threat modeling
- Execution
  - Collect data
  - Excersise response
- Feedback
  - Focus on realistic risks
  - Stop fetishizing bugs

Questions?