Vulnerability analysis

10th ICCC, Tromsø

Simplicity is the ultimate sophistication
Philosophy

*Vulnerability analysis* unravels the knots in our thinking; hence its results must be simple, but its activity is as complicated as the knots that it unravels.

Ludwig Wittgenstein
(In)vulnerability analysis is the end-goal

- Showing the product is secure against all attacks (up to a certain point)

- All other assurance parts can be seen as supporting it:
  - APE/ASE: scoping the claims and attacker
  - ADV: knowledge of the TOE
  - ATE: functional behaviour without the attacker
  - AGD: environment knows what it has to do
  - ALC_CMx: we have the right products
  - ALC_DVS: attacker was not in the development and production process
(In)vulnerability analysis

“Make sure I will not get pricked by a needle if I lie in the hay”

Show **absence** of an attack path within the attack potential
How to show *absence* of something?

- Proof by construction

- Proof by contradiction

- “Proof by exhaustive checking”
  - only* as good as exhaustiveness of checking
  - assurance now == showing exhaustiveness

Very Zen shrine in Kyoto
How to show absence of something?

- Proof by construction
- Proof by contradiction
- “Proof by exhaustive checking”
  - only* as good as exhaustiveness of checking
  - assurance now == showing exhaustiveness

Very Zen shrine in Kyoto
How to show absence of something?

- Proof by construction

- Proof by contradiction

- “Proof by exhaustive checking”
  - only* as good as exhaustiveness of checking
  - assurance now == showing exhaustiveness

Very Zen shrine in Kyoto
Process steps in an (in)vulnerability analysis:

› Gathering information
  › Understanding the assets to protect
  › Learning the product
  › Distributing the assets over the product parts
  › Learning the attack steps

› Analysis and testing
  › Consider the attacks
  › Analyze why not applicable, or
  › Analyze/test why not effective

› Presentation of rationale
  › Convince reader that product is secure
Process steps in an (in)vulnerability analysis:

- Gathering information
  - Understanding the assets to protect
  - Learning the product
  - Distributing the assets over the product parts
  - Learning the attack steps

- Analysis and testing
  - Consider the attacks
  - Analyze why not applicable, or
  - Analyze/test why not effective

- Presentation of rationale
  - Convince reader that product is secure
Process steps in an (in)vulnerability analysis:

- Gathering information
  - Understanding the assets to protect
  - Learning the product
  - Distributing the assets over the product parts
  - Learning the attack steps

- Analysis and testing
  - Consider the attacks
  - Analyze why not applicable, or
  - Analyze/test why not effective

- Presentation of rationale
  - Convince reader that product is secure
FCS_COP
- Confidentiality of keys
- Integrity of plaintext
- Confidentiality of randoms
- Unpredictability of randoms
- Integrity of data (by detecting attacks)
- Confidentiality of data (preventing attacks)

FCS_RNG
- Confidentiality of randoms
- Unpredictability of randoms

FPT_PHP
FCS_COP

Confidentiality of keys

Integrity of keys

Confidentiality of plaintext

Integrity of plaintext

FCS_RNG

Confidentiality of randoms

Unpredictability of randoms

Integrity of data (by detecting attacks)

FPT_PHP

Confidentiality of data (preventing attacks)
Locate the assets in the product

- Physical distribution
- Logical distribution
- Timeline distribution
Confidentiality of keys

Integrity of keys

Confidentiality of plaintext

Integrity of plaintext

Confidentiality of randoms

Unpredictability of randoms

Integrity of data (by detecting attacks)

Confidentiality of data (preventing attacks)

Confidentiality data in CPU

Confidentiality data on bus

Confidentiality data in memory

Confidentiality data in copro

Fault injection

Physical attack

Side channel

Logical attack

Buffer overflow

Protocol attack

SQL injection
It is not possible to test

All possible attack steps

X

All asset locations

But they should all be considered
End result of mapping the security mechanisms: complex
Complexity for the analyst during the process
Only human

- Short term memory limit $7\pm2$
- Long term memory limitation
- Avoidance of “unnecessary” work
- Mistakes in analyses
- Tendency to not question own judgment
- ...

...
Meet the inspiration:
Traits of an ideal evaluator:
Curiosity for new technology
Traits of an ideal evaluator:
Urge to find breaking point
So we introduced security measures
... keeping him from the assets
... but we missed one spot
Hence the project name “Berke”
“Berke” supporting the human memory

- Short term memory
  - Easy marking of items to check later
  - Completeness verified constantly

- Provide lists of known attacks
  - From mandatory guidance
  - Other attack lists
  - Extracted from presentations, proceedings and papers
  - Summarized from own knowledge and experience
“Berke” supports the analysis

- Automatic calculation of cheapest attack path
  - CC attackpoint-calculation
  - General time/cost-curve estimate

- Identification of weakest spot(s) in analysis

- Known attack-list includes
  - Minimum and maximum costs of attack step in general
  - Possible argumentations why the attack step is not applicable
  - Suggested analysis methods
Interesting points (re)learned about CC attack point rating

Adding attack steps is non-trivial in general:

- An attack step is described in terms of 5/6-tuple
  - Elapsed time
  - Expertise
  - Knowledge of TOE
  - Window of opportunity
  - Equipment
  - [Open Samples]

- Elapsed time and Window of opportunity are times that add up*.
- Expertise, Knowledge of TOE, Equipment and Open Samples are maximums*.

- Seek the optimized minimum over this “sum”
Interesting points (re)learned about CC attack point rating

- **Expertise:** how many proficient persons?
  - 1 proficient person: 3 points
  - 2+ proficient persons: 3 points
  - 1 expert: 6 points
  - 2+ experts: 8 points
CEM Attack rating / day

~220
Hurts at serial brute force attacks

- DNS collision attacks on resolvers (~30 weeks)
- Active skimming on ePassports (~5 years 24/7)
- Brute force on logins with maximum rates

High assurance analysis possible and no point reward
Vulnerability testing approaches

- Testing for known vulnerabilities

- Pounding on it and see what breaks

- Analyzing for weakest spot(s) and pounding on it

- Tests for absence properties needed for vulnerabilities
Testing for known vulnerabilities

Examples

- Nessus
- RNG test suites

Pro:

- Well established
- “Vulnerability found” verdicts strong* value
- Relatively easy to run

Con:

- *Often quite a bit of false positives
- “No vulnerabilities found” verdicts weak value
- False negatives
Pounding on it and see what breaks

- Examples
  - Fuzzing
  - Perturbation testing (especially modern automated ones)

- Pro:
  - Good at finding unexpected vulnerabilities
  - Can reach excellent coverage

- Con:
  - Good at breaking TOEs
  - False positives
  - Coverage difficult to verify
  - Often not clear exactly what broke
Analyzing for weakest spot(s) and then pounding on it

Examples
- Code review + SCA/perturbation
- Object code analysis + attack

Pro:
- Analysis shows potential improvement points
- Focussing/reducing testing in relevant areas
- “X marks weakest spot, pounding on it did not work” verdicts
  strong value

Con:
- Analysis model never is perfect
Tests for absence properties needed for vulnerabilities

- Examples
  - Portscanning (for example nmap)
  - Standard deviation and correlation plots in side channel analysis

- Pro:
  - Excellent at excluding vulnerability classes

- Con:
  - Often not easily possible
  - Can require significant theoretical analysis
Vulnerability testing approaches

- Testing for known vulnerabilities
  - Low assurance/low cost

- Pounding on it and see what breaks
  - High assurance/medium cost for developer,
  - Medium assurance/medium cost or high assurance/high cost for evaluator

- Analyzing for weakest spot(s) and pounding on it
  - High assurance/medium cost

- Tests for absence properties needed for vulnerabilities
  - High assurance/low cost but limited availability
Rationale making

- Good arguments for the rationales hard
  - What makes the attack work?
  - Why clearly covered?
  - Once found, very re-usable

- Ordering of arguments matter of taste

- Making an explanation for maximum understanding at certification side is a skill

- Keep on critically thinking.
Further value of complete analysis

- Estimate robustness for attack improvements
- Assurance maintenance
- Incident response
- Find weaker spots in the analysis/product
- See the value of countermeasures in product
From complexity…

…to simplicity
I hope you are not dog-tired now and have some questions?
Attributions

“Rope” by Aubergene @flickr.com CC-BY

“Dsc_0034” by supermoving @flickr.com CC-BY

© istockphoto.com, licensed for the presentation

“Security by Zemlinki @flickr.com CC-BY

“Crowd” by James Cridland, @flickr.com CC-BY

“Code” and “Security” by Anonymous Account @flickr.com CC-BY
Attributions

“Security” by Basheer @ flickr.com CC-BY

“Lock it up” by photographix.ca @ flickr.com CC-BY

“Sheepshank” by psd @ flickr.com CC-BY

All other images © Wouter Slegers