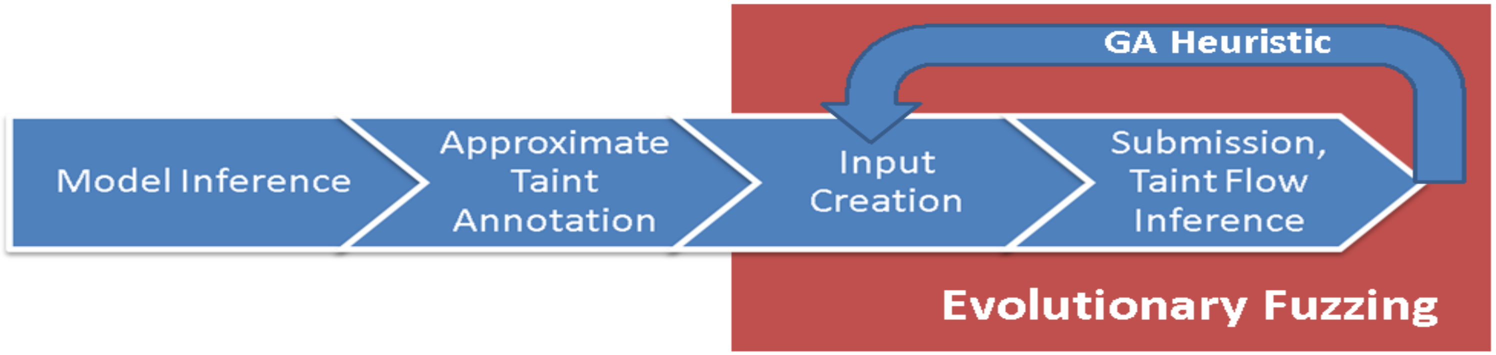
Development and Industrial Application of Multi-Domain Security Testing Technologies

Innovation Sheet
Model Inference Assisted Evolutionary Fuzzing

Model Inference Assisted Evolutionary Fuzzing
Description

The technique dynamically analyzes the application (as black-box) for detecting vulnerabilities (e.g. XSS). The main features are:

1. Model Inference and Sink Annotation:
   - In order to fuzz *smartly*, we infer the model *M* of the application (SUT) by making use of a *state-aware crawler*. A light-weight *taint analysis* is used to annotate nodes of *M*, where input is *reflected* in output (pattern for XSS vulnerability).
2. Evolutionary Fuzzing for Malicious Input Generation:
   - Given an annotated *M*, genetic algorithm (GA) is used to evolve inputs that trigger the existing vulnerability.
   - An attack grammar is used to guide/control future input generations.
   - For exploitability, taintflow is inferred using a complex string matching algorithm.
   - Output: Concrete inputs that trigger the vulnerability.



Model Inference Assisted Evolutionary Fuzzing
State of the art

- Model Inference:
  - State-of-the-art tools such as Google SkipFish [1] and Rapid7 w3af [2] have low detection rate, mainly because such tools do not precisely learn the controlflow of the web application. Doupe et al [3] presented a very impressive technique for learning the model and detecting vulnerabilities.
    1. Zalewski, M., Heinen, N.: Skipfish - web vulnerability scanner (2009) http://code.google.com/p/skipfish/ (accessed 2012-09-11).
    2. Riancho, A.: w3af - web application attack and audit framework http://w3af.sourceforge.net.
    3. Doupe, A., Cavedon, L., Kruegel, C., Vigna, G.: Enemy of the state: A state-aware black-box web vulnerability scanner. USENIX Security (2012).
- Taintflow Inference:
  - Some web scanners[5] assume that the fuzzed value is reflected as such in the SUT output, which leads to false negatives, even when using regular expressions [6,7]. Sekar et al presented a robust algorithm for taint inference in blackbox[8].
    5. A. Riancho, w3af - web application attack and audit framework," http://w3af.sourceforge.net.
    6. 22. D. Ross, IE 8 XSS filter architecture/implementation," 2008,
    7. 23. G. Maone, NoScript, firefox plug-in," https://addons.mozilla.org/en-US/firefox/addon/noscript/.
    8. R. Sekar, An Efficient Blackbox Technique for Defeating Web Application Attacks,"in Network and Distributed System Security, 2009.

Model Inference Assisted Evolutionary Fuzzing
State of the art

- Vulnerability Detection:
  - Confinement Based Approaches assume that malicious inputs break the structure at a given level (lexical or syntactical) and exploit this for vulnerability detection[9].
    9. Z. Su and G. Wassermann, The essence of command injection attacks in web applications," in Symposium on Principles of Programming Languages, 2006.
  - Genetic algorithms have been used to generate inputs that may trigger the targeted vulnerability[10, 11]. However, care should be taken when defining the corresponding fitness function to have better detection[12].
    10. J. D. DeMott, R. J. Enbody, and W. F. Punch, Revolutionizing the field of greybox attack surface testing with evolutionary fuzzing," Black Hat USA, 2007.
    11. J. Budynek, E. Bonabeau, and B. Shargel, Evolving computer intrusion scripts for vulnerability assessment and log analysis," in Genetic and evolutionary computation, ser. GECCO '05. ACM, 2005.
    12. Sanjay Rawat and Laurent Mounier, "An Evolutionary Computing Approach for Hunting Buffer Overflow Vulnerabilities: A case of aiming in dim light" In the proc. of 6th EC2ND (European Conference on Computer Network Defense, Berlin, Oct 2010, IEEE CS

Model Inference Assisted Evolutionary Fuzzing
Advances beyond the state of the art

- Combining evolutionary fuzzing with state-based models
- More accurate targeting with attack grammar

In general, a large percentage of the fuzzing approaches on vulnerability detection can be classified as random fuzzing. Evolutionary fuzzing reduces the randomness to some extent. We extended the idea of evolutionary fuzzing to state-aware fuzzing by learning the model of the SUT. We also improved the state-explosion problem of evolutionary computing by introducing the idea of attack grammar while generating the inputs.

[Deliverable D3.WP2, Section 2.2.2]

- Taint analysis performed on black-box model
- Lightweight static taint analysis + dynamic refinement

Taintflow analysis is the basis for vulnerability detection (exploitability aspect). However, performing taintflow analysis in black-box environment is non-trivial. We adapt the existing work on taintflow in black-box environment to infer the taintflow which is light-weight and precise. The whole approach is implemented in a tool and tested on real-world applications.

[Deliverable D5.WP2, Section C-III]

Model Inference Assisted Evolutionary Fuzzing
Exploitation and Application to Case Studies

- The proposed work have been presented in international conferences and various technical meetings/discussions/seminars.
- The approach has been applied to Gemalto case study.
- The approach has been implemented as a tool to make it available for larger usage.