Michael Eddington

DEMYSTIFYING FUZZERS

Agenda

- Introduction
- Why are we fuzzing?
- Types of existing fuzzers
- Fuzzing, process
- Adoption Risks
- Fuzzing costs
- Pulling it all together

Why are we fuzzing?

ROI^2!

All about the bugs!

- …Or really Bug Cost…
- Fuzzing is about finding bugs
- Fuzzing is repeatable
- Fuzzing *should* be easy on the wallet
  - Cost per Bug

Types of existing fuzzers

Types of Fuzzers

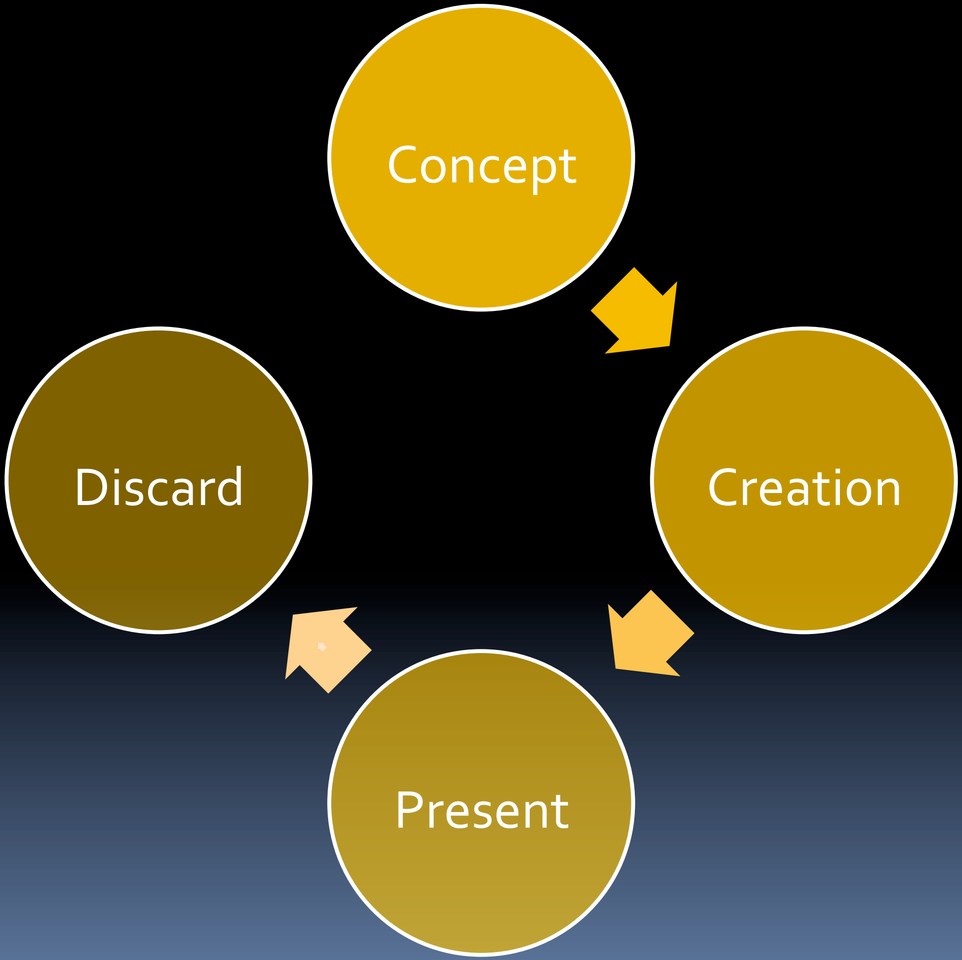
File	• Only creates files on disk • FileH/FileP
Network	• Generates network packets • TAOF, Sully
General	• Pluggable I/O interfaces • Peach
Custom	• Single target fuzzer • “Fuzzer for LDAP”

Open Source Fuzzers

- Lots to choose from
- More every year

- Bob’s Taco Fuzzer!

Open Source Fuzzers



..So what’s left?

- Small grab bag of fuzzers
- Which should we use?
- Do they finds the bugs?

…introducing…

File	Network	General	Custom/One-off
FileH/FileP	Sulley	Peach	AxMan
FileFuzz	GPF	SPIKE	DOM-Hanoi
	EFS	Fuzzled	Hamachi
	TAOF	Fuzzware	Mangleme
	Querub		

…open source fuzzers…

…introducing…

File	Network	General	Custom/One-off
FileH/FileP	Sulley	Peach	AxMan
FileFuzz	GPF	SPIKE	DOM-Hanoi
	EFS	Fuzzled	Hamachi
	TAOF	Fuzzware	Mangleme
	Querub		

Actively Maintained
Bug Fixes Only
Unknown
Un-Maintained, but used

…open source fuzzers…

Commercial Fuzzers

- Mu Dynamics (aka *Mu Security*)
  - Network only!
- beSTORM
  - General
- Codenomicon
  - The general fuzzer that isn't a fuzzer

One-off fuzzers

- Dom-Hanoi
- Hamachi
- Mangleme
- AxMan

- Sometimes needed but…
- Where are the mutations!?

Fuzzing, the process

The Process

- Investigate
- Modeling
- Validate
- Monitor
- Run
- Results

Investigate

- Determine what needs fuzzing
- Mapping fuzzer capability to need

Modeling

- Model data of our system
  - Data Types
  - Relationships (size, count, offset)
  - Etc.
- Model state of our system
  - Send, Receive, Call, etc.
- Most of your time is spent here
  - Unless a model already exists!

Modeling

- Large difference between fuzzers
  - Language (Code vs. XML vs. Custom)
  - Extent of modeling allowed
  - Tools
    - GUI Tools
    - Format -> Model converters

Modeling Examples

- Peach – XML

```
<DataModel name="Example">
 <Number size="8" signed="true">
  <Relation type="size" of="Name"/>
 </Number>
 <String name="Name" value="John Doe" />
</DataModel>
```

Modeling Examples

- Sulley – Python/SPIKE

```
s_size("Name", length=1, fuzzable=True)
if s_block_start("Name"):
   s_string("John Doe")
s_block_end()
```

Validate

- Verify model matches reality
- Are tools provided?

- This is critical!!

Validate

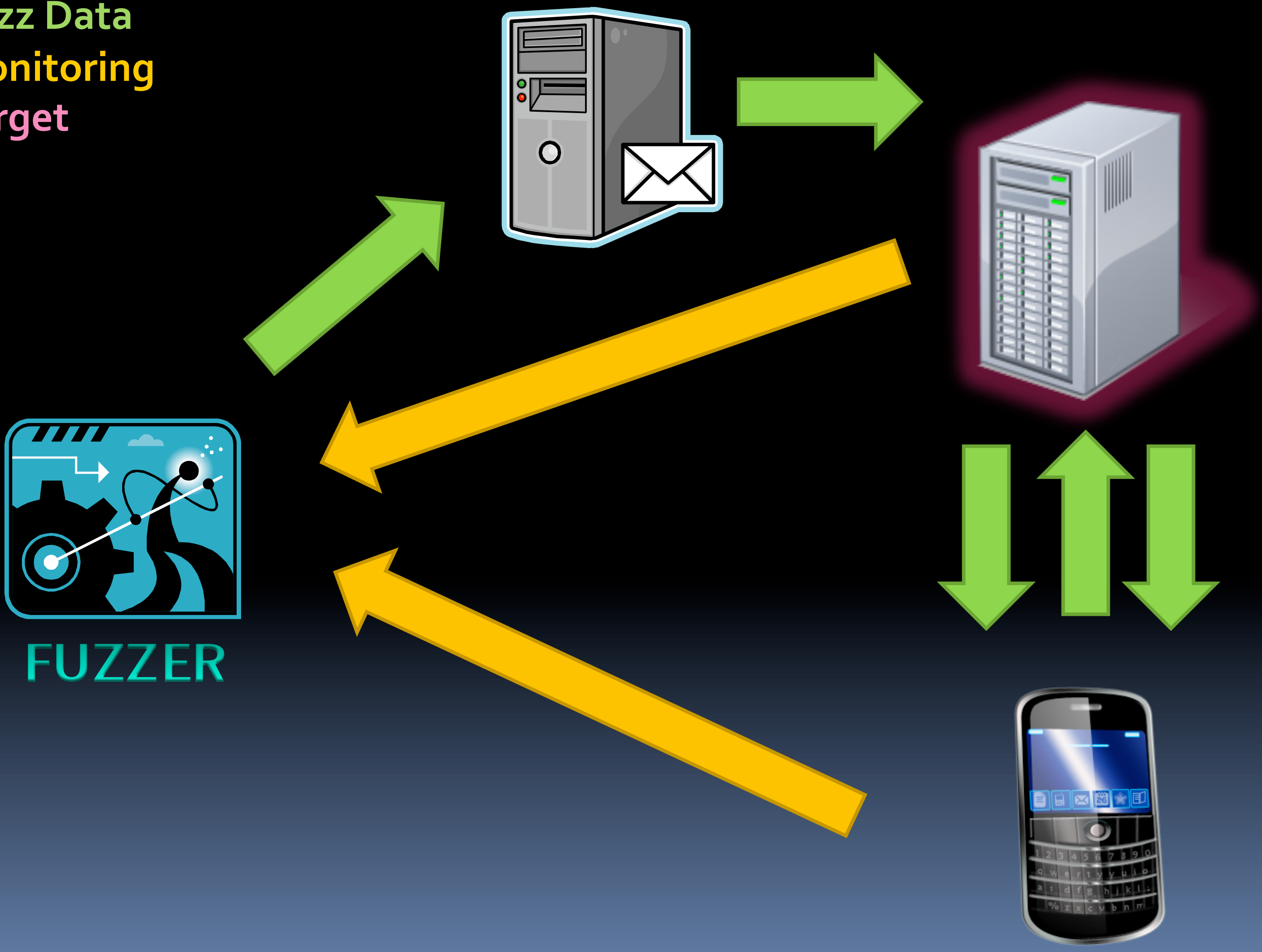
Validation Tools		
Peach	✓	GUI Tool & Debug Ouput
Sulley	✓	Coverage analysis
SPIKE		
Fuzzled		
Fuzzware		
GPF		
EFS	N/A	
TAOF		
Mu Security	?	
Codenomicon	⊘	
beSTORM	?	

Monitor

- Sending data is just the beginning
- Fault detection
- Data collection
- Complex setup support

BlackBerry Example

Fuzz Data
Monitoring
Target



Monitor

- Basic monitoring:
  - Debugger
  - Network capture
- Advanced monitoring
  - Easily pluggable
  - VM Control

Monitor

	Debug	Network	VM	Extensible
Peach	✓	✓	✓	✓
Sulley	✓	✓	✓	
SPIKE				
Fuzzled				
Fuzzware	✓			
GPF				
EFS	✓			
TAOF	✓			
Mu Security	✓	✓		
Codenomicon	✓	?		
beSTORM	✓	?		

Run

- Joined at the hip with Monitoring
- Can fuzzer continue past fault?
- Can we run in “parallel” mode?

Parallel Runs

- Single iteration from 5 to 60 seconds or more
- Target iterations: 250,000 -> 500,000
  - 500,000 tests * 30 seconds/test = 174 days!
  - Parallel by 10 = 17 days
  - Parallel by 20 = 9 days
- Run across multiple machines
  - Entry: 10 to 100
  - Advanced: 100 to 10,000+

Run

	Windows	Unix/ OSX	Kernel	Symbols	Parallel	Process Restart
Peach	WinDbg	VDB	Win	Win	✓	✓
Sulley	System				✓	✓
SPIKE						
Fuzzled						
Fuzzware	WinDbg			Win		✓
GPF						
EFS	System					✓
TAOF	System					✓
Mu Security	?	?			?	
Codenomicon	?	?			?	
beSTORM	?	?			?	

Results

- Time intensive to sort hundreds of crashes
- Many crashes not interesting
- Many crashes are duplicates

- Crash Analysis!!

Crash Analysis

- Bucketing of duplicate crashes
  - Hundreds to thousands of duplicates
- Analysis of exploitability

- Microsoft’s !exploitable for WinDbg
  - Peach
  - ???

Results

	Group Duplicates	Crash Analysis
Peach	✓	✓
Sulley	?	
SPIKE		
Fuzzled		
Fuzzware	✓	
GPF		
EFS		
TAOF		
Mu Security	?	?
Codenomicon	?	?
beSTORM	?	?

Adoption Risks

Adoption Risks

- Sustainability
- Usability or maturity
- Training & Support
- License Restrictions

Sustainability

- How many years has tool existed?
- When was last release?
- Does project have commercial backing?
- How many active leaders?
- Active community?
  - Forums, mailing lists, etc.

Sustainability

	Current Version	Last Release Date	Years Available	Commercial	Active Community
Peach	2.3	2009	5		Yes
Sulley	?	2009*	2		
SPIKE	2.9	2004	7		
Fuzzled	1.1	2007	2		
Fuzzware	1.5	2009	1		
GPF	4.6	2007	2		
EFS	?	2007	2		
TAOF	0.3.2	2007	2		
Mu Security	?	2009	4	Yes	
Codenomicon	3.0	2009	8	Yes	
beSTORM	3.7	2008	5	Yes	

Usability

- ...possibly Maturity?
- Documented?
- Online support forums? Do people answer questions?
- Publications? (e.g. books)
- Are external users a priority?
  - Vs. Internal tool released publicly

Support & Training

- Training
  - Get staff going fast
  - Taking it to next level
- Support
  - Bugs, etc.
  - Assistance

License Restrictions

- Code changes
- Integrate into development cycle
- Taint issues?

License Restrictions

- GPL
  - Must release changes
  - Taint issues?
- MIT
  - No restrictions
- BSD
  - No restrictions
- Commercial
  - Should be okay for use

Adoption Risks

	Sustainability	Usability	Training	Support	License
Peach	4	4	Yes	Yes	MIT
Sulley	3	4			GPL
SPIKE	1	1			GPL
Fuzzled	2	2			GPL
Fuzzware	3	4			~BSD
GPF	1	3			GPL
EFS	1	2			GPL
TAOF	2	3			GPL
Mu Security	4	5	Yes	Yes	Yes
Codenomicon	5	5	Yes	Yes	Yes
beSTORM	4	5	Yes	Yes	Yes

Fuzzing $$$ Costs

Time Spent in Order

1. Modeling
   - Data & State, aka Creating a Definition
2. Monitoring
   - Debugger Collection
   - Network capture (or other)
   - Restarting fuzzer
3. Crash Analysis
   - Is it exploitable?
   - Is it a duplicate?

Upfront Costs

	Price	Restrictions/Time Limits
Open Source	$0	
Codenomicon	$5,000 for 5 protocols for 5 days	5 days, other models available
Mu Security	$50,000 for 10 protocols; $250,000 for all protocols	12 month license
beSTORM	$15,000 per module	None?

Hidden Costs

- Ramp-up Time
- Modeling
- Crash Analysis

- Paying to avoid these
    - But…custom formats/protocols…

Wrapping Up

SDL

- Fuzzing as part of SDL widely different from Research fuzzing. Companies have limited budget, resources, and time frame.
  - Need crash analysis
  - Need integrated monitoring of target
  - Need parallel run ability (for Smart)

Open vs. Commercial

- Fuzzing definitions (grammars)
- Training
- Support
- Consulting Services
- “Easy to use”

And the winner is…

Q & A

Michael Eddington

mike@phed.org

http://phed.org

http://peachfuzzer.com

http://leviathansecurity.com