First Meeting on PHP DoS Project

10 PM HKT 7th Auguest

DoS attack in PHP

- DoS attack is serious in server-side web applications
- PHP is still the most popular programming language in server-side applications
- No existing work targets on such general DoS attack detection in PHP
- PHP is a type of dynamic language, program analysis on PHP source is already difficult

Expectations

- Support mostly common syntax of PHP
- Find known vulnerabilities pointed noted in previous CVEs
- Test more web applications and detect new ones
    - Do some analysis and classification based on our results

Methodologies

Generally speaking, our work mainly falls into two parts

- Taint-tracking analysis from user inputs to locate potential sinks
  - Mark the loops whose run time might be determined by user inputs as potential sinks

- $_GET[‘time’] and $_GET[‘test’] are from user inputs
- strlen($y) determined by user input $_GET[‘time’]
- Mark the for loop as a potential sink

```
<?php
$x = $_GET['test'];
$y = $_GET['time']
if($x == 'DoS_Example'){
    for($i = 0; $i < strlen($y); $i ++) {
        ...
        ...
    }
}
```

- Symbolic execution and constraint solving to verify potential sinks and cut down false positives
  - Collect constraints from the potential sinks to programe starts
  - Use existing constraint solver (Z3 SMT) to generate inputs

- From the for loop tracking back to program start
- $_GET[‘test’] == ‘Dos_Example’ describes how program reaches the potential sink
- Get inputs forcing the for loop to a DoS

```
<?php
$x = $_GET['test'];
$y = $_GET['time']
if($x == 'DoS_Example'){
    for($i = 0; $i < strlen($y); $i ++) {
        ...
        ...
    }
}
```

Current status

- Taint-tracking part supports most of the cases
  - Build control-flow graph and call graph to do data flow analysis
- Backwards symbolic execution part can already collect constraints
  - Constraints describe the condition program should satisfy to reach certain locations
- Try Z3 SMT to solve constraints
  - Problem: Current constraints solvers almost focus on static languages, where variables & functions should be declared in a certain type before using, however, dynamic types are allowed in PHP

Others

- Very eager to get results to see whether our method works practically
- Monitoring program execution when feeding inputs to test