Automatic Worm Defense (II) -- More on Automatic Signature Generation

Dawn Song

dawnsong@cs.berkeley.edu

Bouncer: Securing Software by Blocking Bad Input

Main Idea

How to generalize from original exploit?

- **i.e., how to generalize from MEP Signature?**
  - **Remove unnecessary constraints on one path**
    - » **Precondition slicing**
    - » **Function summaries**
  - **Create different exploits to increase path coverage**

Background: Program Slicing (I)

- **A program slice:**
  - The set of all statements/instructions that might affect the value of a variable occurrence
- **Goal:**
  - A slice should evaluate the variable occurrence identically to the original program for all inputs
- **Compute slicing**
  - Data dependency
  - Control dependency
- **Property:**
  - Independent of input values
- **Applications:**
  - Program verification, testing, etc.

Background: Program Slicing (II)

```
int x=0, y=0;
int *z = &y;
if (msg[0] == 'a')
   x = 1;
if (msg[1] == 'b')
   z = &x;
*z = 0;
if (x)
  Vulnerability = TRUE;
```

- **What's in slice for Vulnerability?**
- **Issues with static slicing**
  - Conservative, too large (close to original program)

Background: Dynamic Slicing (I)

- **A narrower notion of "slice"**
  - Consisting only statements that influence the value of a variable occurrence for specific program inputs
- **Applications**
  - Debugging

Background: Dynamic Slicing (II)

```
int x=0, y=0;
int *z = &y;
if (msg[0] == 'a')
  x = 1;
if (msg[1] == 'b')
  z = &x;
*z = 0;
if (x)
  Vulnerability = TRUE;
```

- What's in slice for Vulnerability for msg="ad"?
- Issues with dynamic slicing for signature generation
  - Miss certain constraints

Precondition Slicing (I)

- Goal
  - Remove unnecessary conditions without false positives
- Path slice for a vulnerability point
  - A subset of instructions in a trace whose execution is sufficient to ensure vulnerability to be exploited
  - Data dependency
    - » Easy
  - Control dependency
    - » Look at all relavant paths

Precondition Slicing (II)

- Aliasing
  - MayAlias (x, y) iff x and y may refer to overlapping storage locations
  - MustAlias (x,y) iff x and y always refer to the same storage locations for all executions
  - Conservative approximations
- Liveness
  - Latest defs for operands used

Precondition Slicing (III)

- Iterative backwards processing
- When will a branch condition not be included in slice?
  - Postdominance relation
  - No path originating at the branch affects values in *live*
  - What common cases will this help?
    - » Table lookup
    - » Case conversion
- When will a function not be included in slice?
  - Execution of the function does not affect values in *live*
- Using dynamic information to improve precision
  - More precise dependency info on given path

Precondition Slicing (IV)

```
int x=0, y=0;
int *z = &y;
if (msg[0] == 'a')
   x = 1;
if (msg[1] == 'b')
   z = &x;
*z = 0;
if (x)
   Vulnerability = TRUE;
```

- What's in slice for Vulnerability for msg="ad"?
- Issues with preconditioning slicing for signature generation
  - Variable length fields, etc.

Advantages

- With soundness guarantee
  - No false positives
- Remove certain unnecessary conditions
  - Conditions imposed by value-dependent processing which are irrelevant to vulnerability

Limitations

- Creating new exploits likely not work
  - Without data analyzer
  - Path exploration with mixed concrete/symbolic execution
    - DART/EXE type of approach
    - Later in class
- Function summaries
- Still can't handle loops, variable length fields, etc.
- May still need TM signature
  - Limited expressiveness

Compare Different Approaches for Signature Generation

- Pattern-extraction based approach
  - W. or w/o exploit detector oracle
  - W. or w/o data analyzer
- Program-analysis based approach
  - MEP signature: fairly well understood
  - PEP signature: How to explore different paths?
    - Precondition slicing, etc.
  - TM signature
- What's the right approach? Why?
- How can we do better?
  - Potential project ideas
    - Come talk to me if interested

Open Mic

- Other thoughts/comments?

Summary

- **Now you are an expert in automatic signature generation for worm/exploit defense :-)**
- **Next: Botnet Analysis**