ASSISTING FUZZING WITH SYMBOLIC EXECUTION

Fuzzing with American Fuzzy Lop

Oversigt

- Baggrund for opgaven
- Driller
- Udfordringer
- Refleksion
- Konklusion

Baggrund

- Problemformulering:

  Kan AFL forbedres gennem brug af Symbolic Execution

- Hvorfor

- Eksempler

Eksempel: Generel input

```c
int main(void)
{
    int x;
    read(0, &x, sizeof(x));

    if (x % 1000 == 0){
        vulnerability();
    }else{
         ...
    }
    return 0;
}
```

Eksempel: Specifik input

```c
int main(void)
{
    int x;
    read(0, &x, sizeof(x));

    if (x == 12345678){
        vulnerability();
    }else{
        ...
    }
    return 0;
}
```

Driller

• Fuzzer	American Fuzzy Lop
• Concolic execution engine	angr

- Lavet af UC Santa Barbara SecLab aka Shellphish

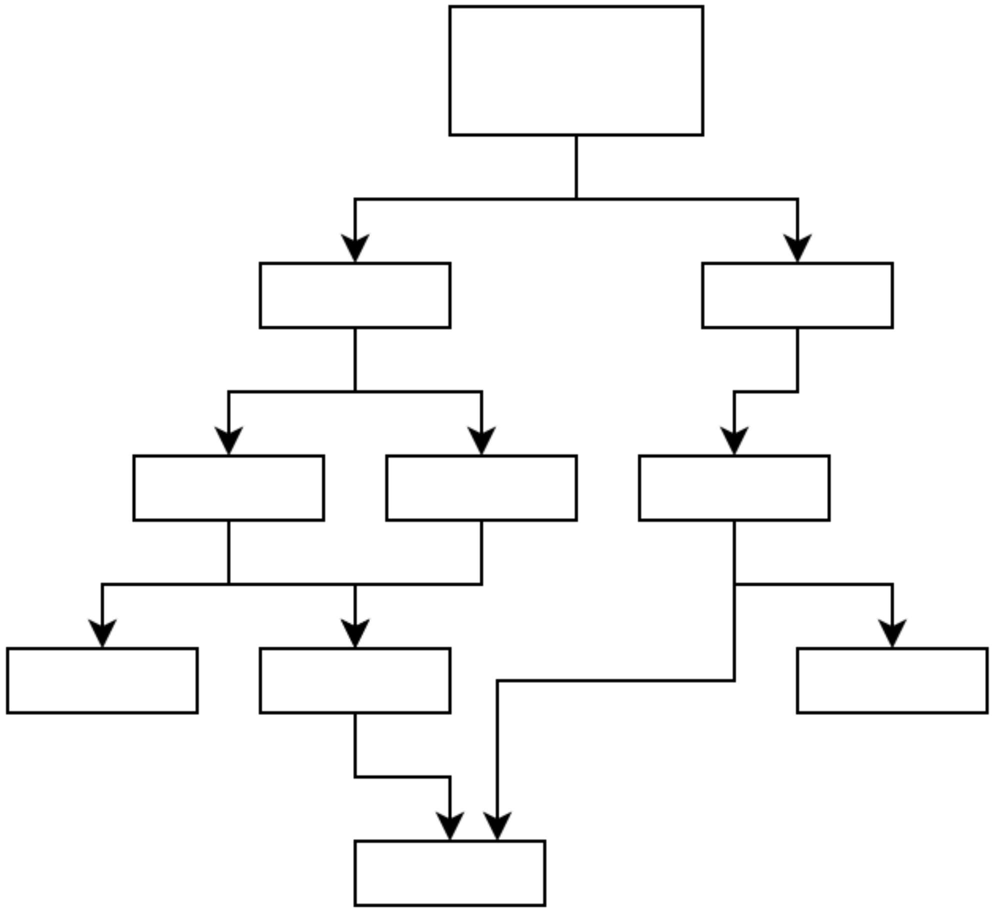
Driller

- Algoritme/automatisering
- Kodedækning
- Tests

Driller: Algoritme

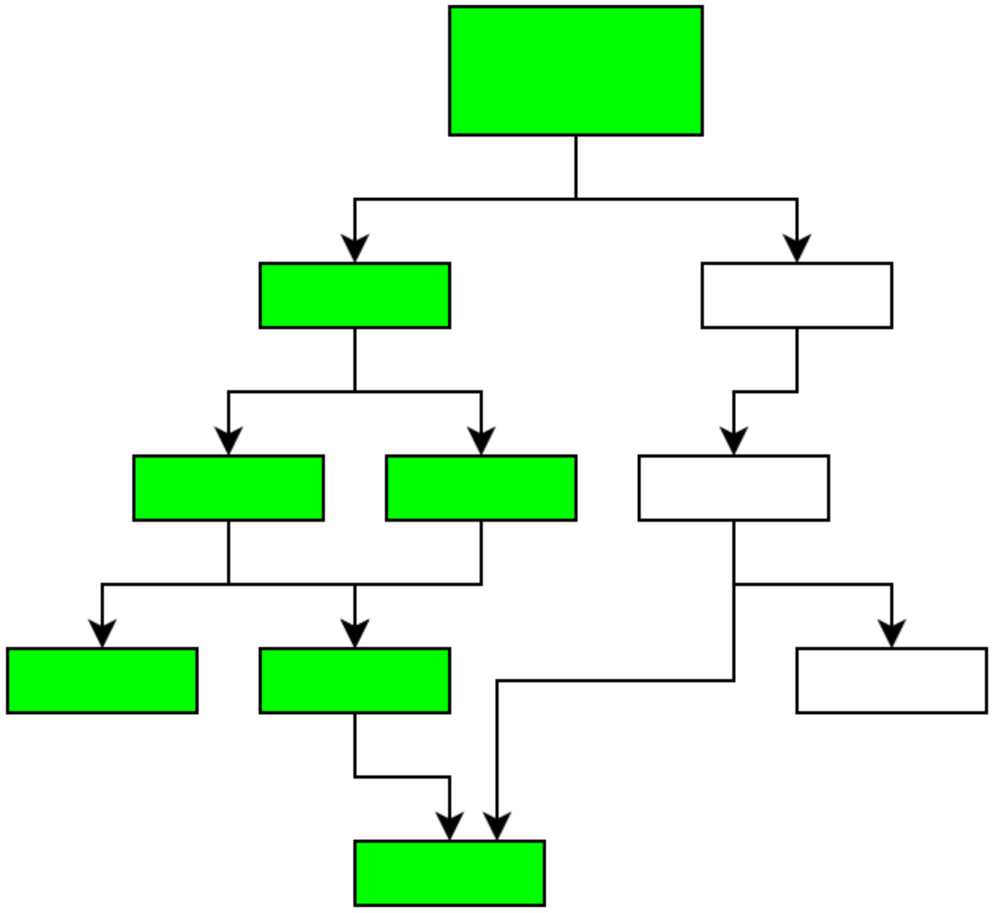
- Initialisering
- Fuzzing
- Concolic execution
- Gentagelse

Driller: Kodedækning



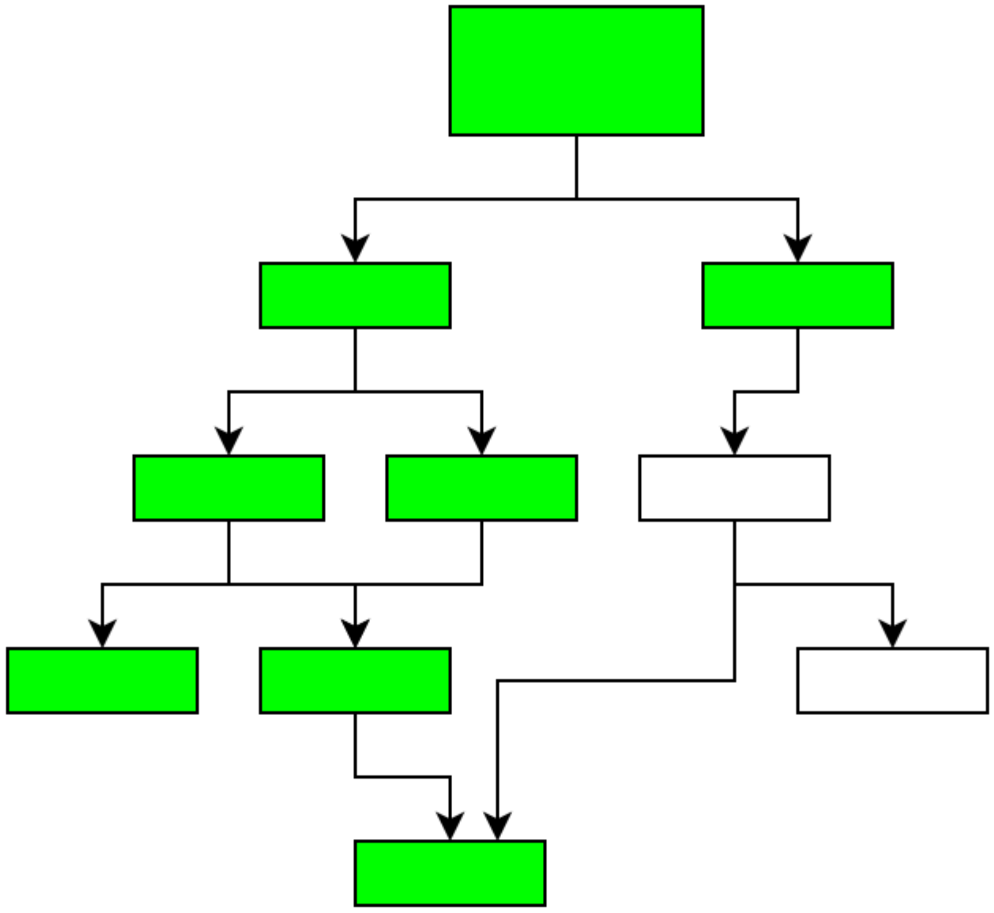
Driller: Kodedækning

- Fuzzing



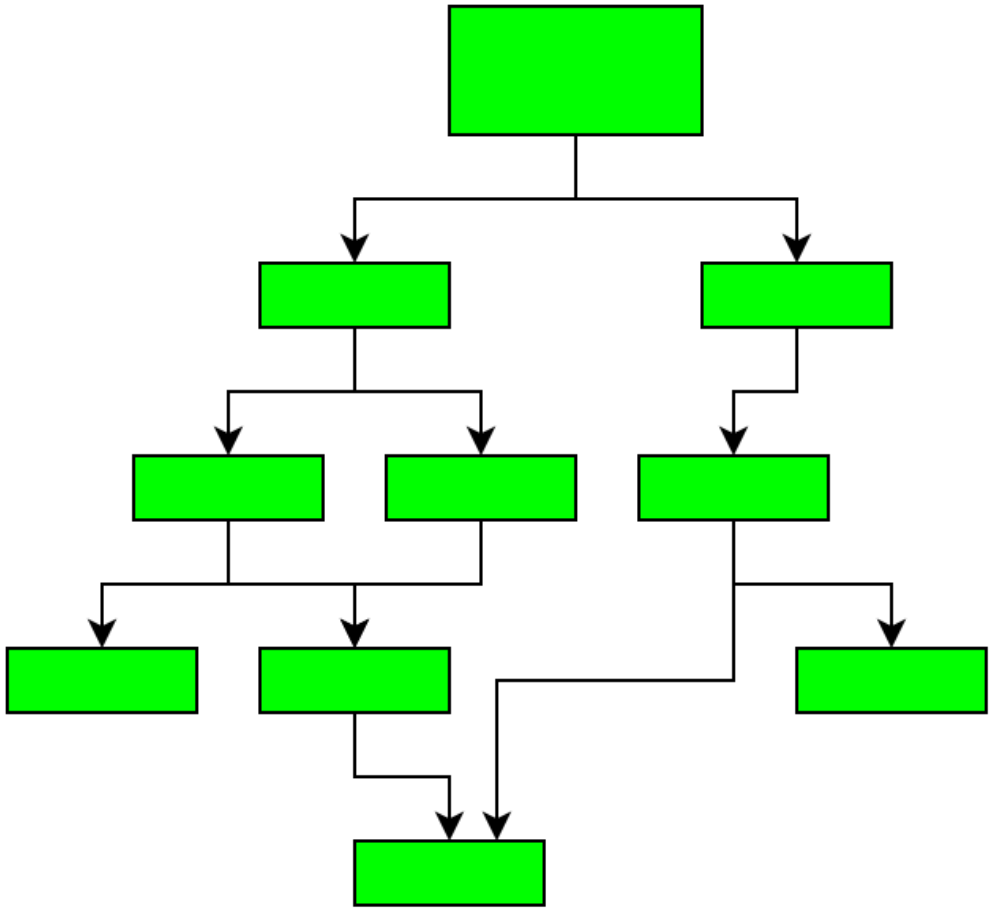
Driller: Kodedækning

- Fuzzing
- Concolic execution



Driller: Kodedækning

- Fuzzing
- Concolic execution
- Fuzzing

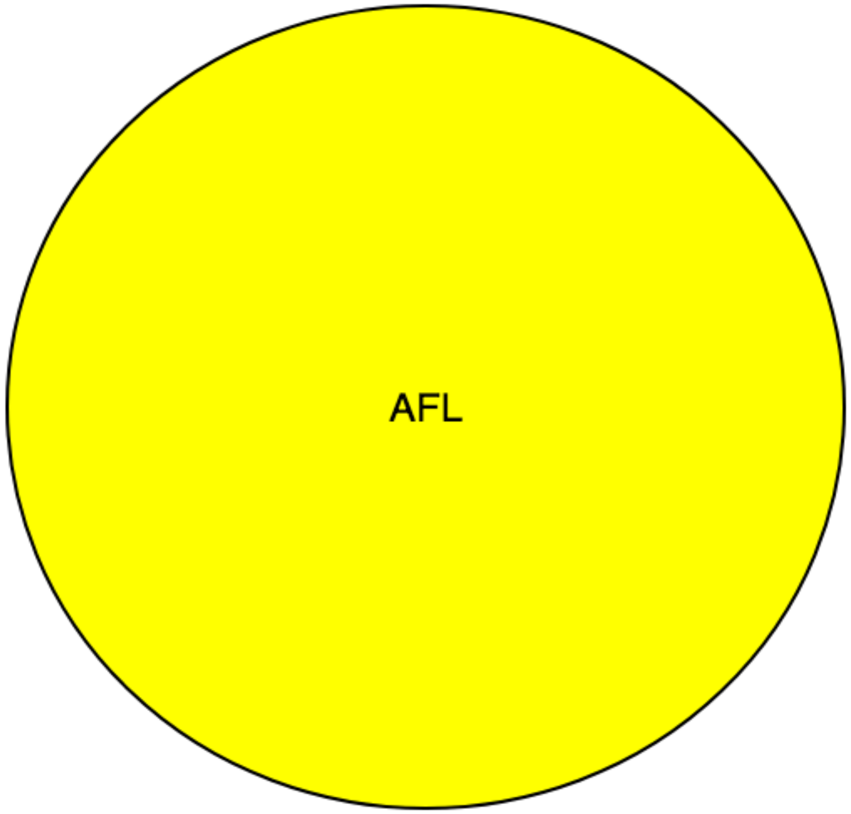


Driller: Tests

- DARPA Cyber Grand Challenge Qualifying Event
- 126 Binaries
- 6 udviklere
- 4 IT-sikkerhedsfirmaer
- *“test the abilities of a new generation of fully automated cyber defense systems”*

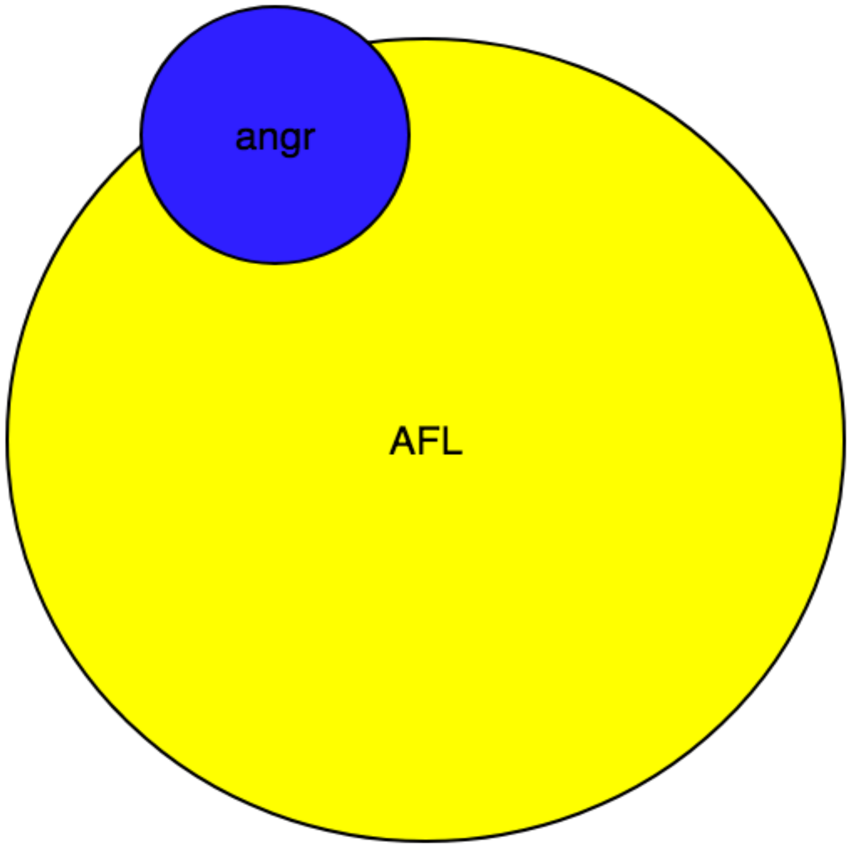
Driller: Tests

- Fuzzing: 68



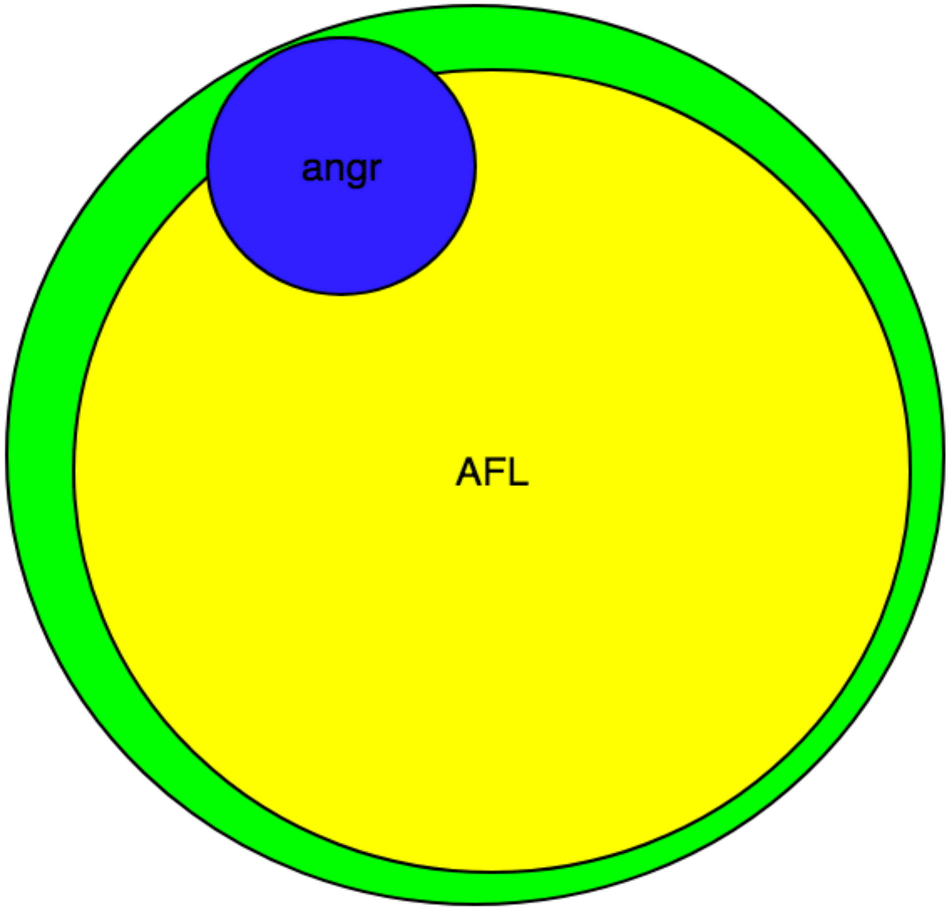
Driller: Tests

- Fuzzing: 68
- Symbolic execution: 16



Driller: Tests

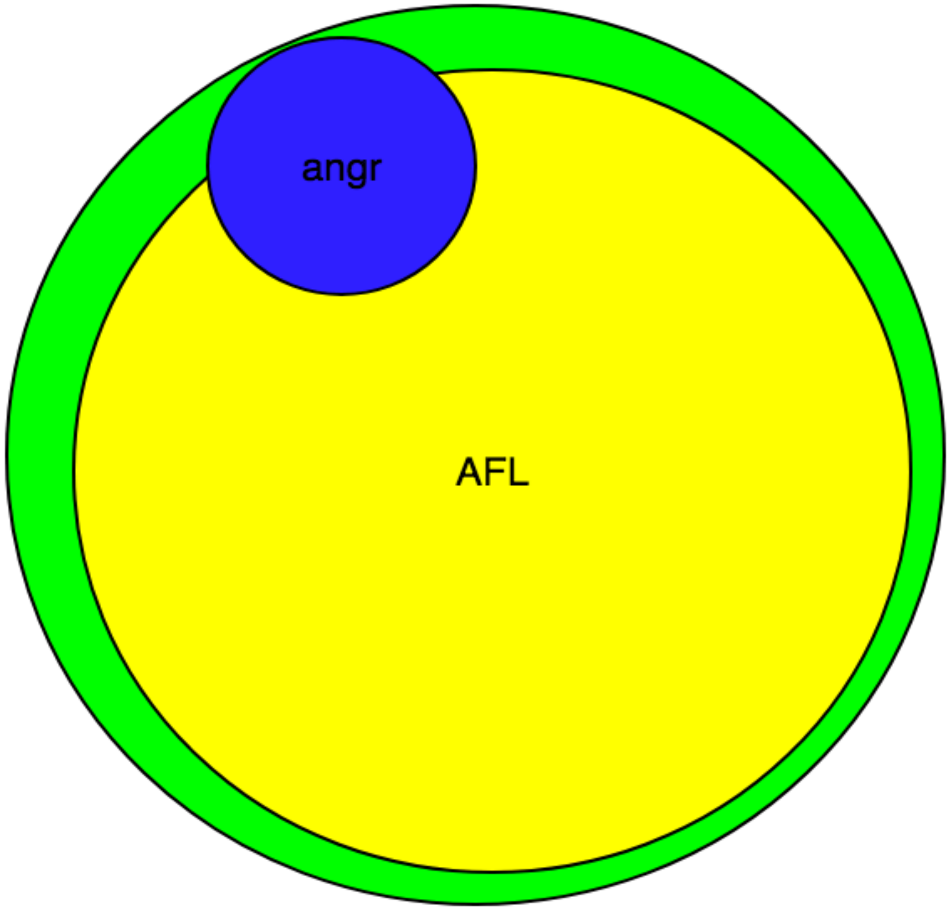
- Fuzzing: 68
- Symbolic execution: 16
- Driller: 77



Driller: Tests

- Fuzzing: 68
- Symbolic execution: 16
- Driller: 77

- Forbedring: 9



Udfordringer

- Stadig under udvikling
- Mangel på brugsrettet dokumentering
- Specialtilpasset til CGC binaries

Refleksion

- Mayhem af ForAllSecurity

- Anden brug af angr

Konklusion

- Fuzzing
- Kodedækning
- Udvikling