# Automatic Exploit Generation

Given a program, find bugs and generate exploits

Thanassis Avgerinos, Brent Lim Tze Hao, David Brumley
Carnegie Mellon University

# # Agenda

- WarmUp
- Demo
- Overview of AEG
- Challenges
- Formal Verification
- Design
- Evaluation
- Discussion
- Conclusion

# WarmUp

```c
int main(int argc, char **argv) {
  int skfd;              /* generic raw socket desc.    */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
    {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig

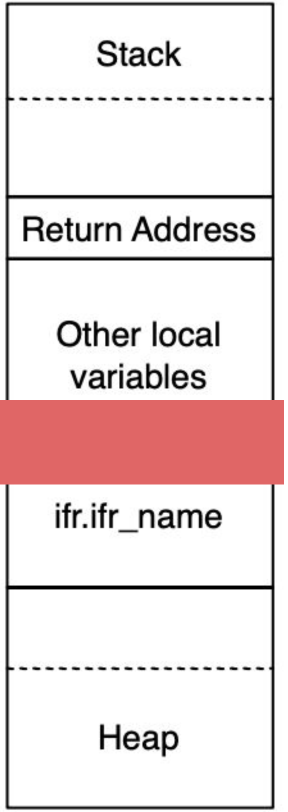

Memory

# WarmUp

```
int main(int argc, char **argv) {
  int skfd;                /* generic raw socket desc.      */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
  struct wireless_info     info;
  int                      rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq             wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Can you spot a bug ?

Code snippet from iwconfig




Memory

# # WarmUp

```
int main(int argc, char **argv) {
  int skfd;              /* generic raw socket desc.     */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
    {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig


Memory

# WarmUp

```
int main(int argc, char **argv) {
  int skfd;                /* generic raw socket desc.      */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count) {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig


Memory

# WarmUp

```c
int main(int argc, char **argv) {
  int skfd;              /* generic raw socket desc.    */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
    {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig

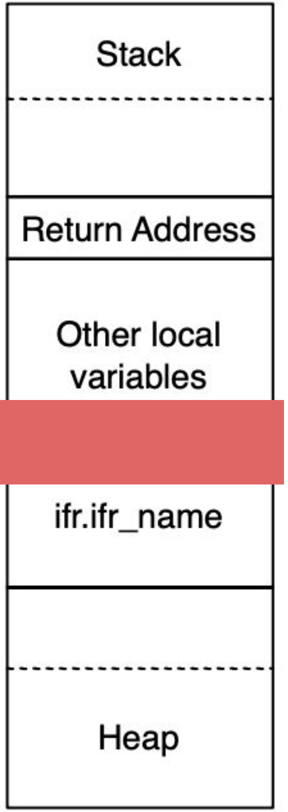

Memory

# WarmUp

```
int main(int argc, char **argv) {
  int skfd;               /* generic raw socket desc.     */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
  struct wireless_info info;
  int                     rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq            wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Note that this ifr.ifr_name is a 32 bytes array

Code snippet from iwconfig




Memory

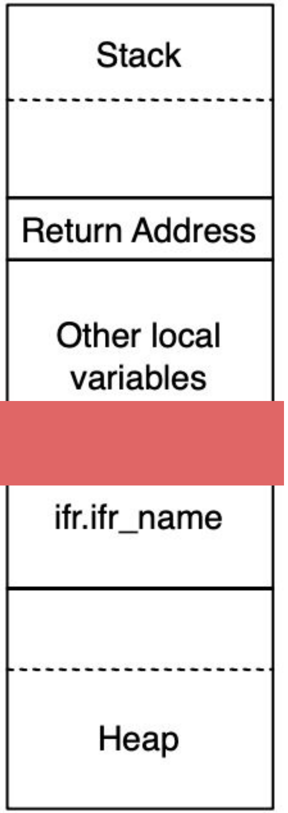
# WarmUp

```
int main(int argc, char **argv) {
  int skfd;                /* generic raw socket desc.     */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
```

What if we put a significant a long input to this program?

```
  int                      rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq             wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig




Memory

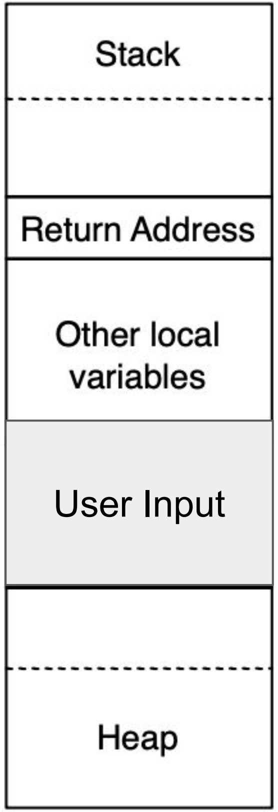
# Trigger buffer overflow !

# How would you exploit it ?

# # WarmUp

```
int main(int argc, char **argv) {
  int skfd;                 /* generic raw socket desc.    */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
    {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig

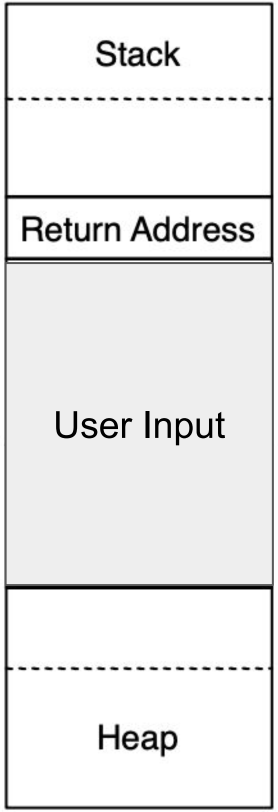



Memory

# # WarmUp

```
int main(int argc, char **argv) {
  int skfd;             /* generic raw socket desc.     */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count) {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig

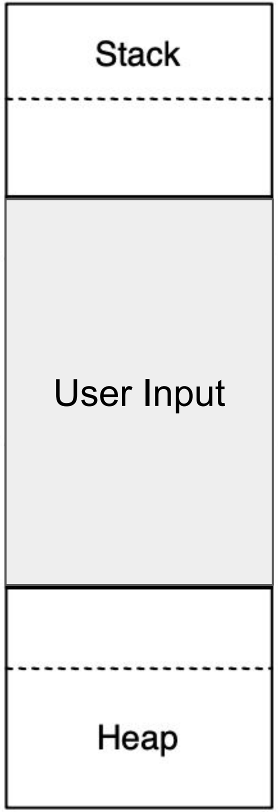



Memory

# WarmUp

```
int main(int argc, char **argv) {
  int skfd;                 /* generic raw socket desc.     */
  if(argc == 2)
    print_info(skfd, argv[1], NULL, 0);
...
static int print_info(int skfd, char *ifname, char *args[], int count)
    {
  struct wireless_info  info;
  int                   rc;
  rc = get_info(skfd, ifname, &info);
...
static int get_info(int skfd, char *ifname, struct wireless_info * info
    ) {
  struct iwreq          wrq;
  if(iw_get_ext(skfd, ifname, SIOCGIWNAME, &wrq) < 0) {
      struct ifreq ifr;
      strcpy(ifr.ifr_name, ifname); /* buffer overflow */
...
```

Code snippet from iwconfig




Memory

# Execve bin/sh !

# Demo !

```
whoami
root
ls
Makefile
access.log
aeg.sh
aeg_A-data
aeg_A-data-stat
aeg_stdin
aeg_stdin-stat
error.log
filesize
klee-last
klee-out-0
portno
recvinfo
runtime_info
server.conf
serverd
serverd.bc
stype
tmpfile
```

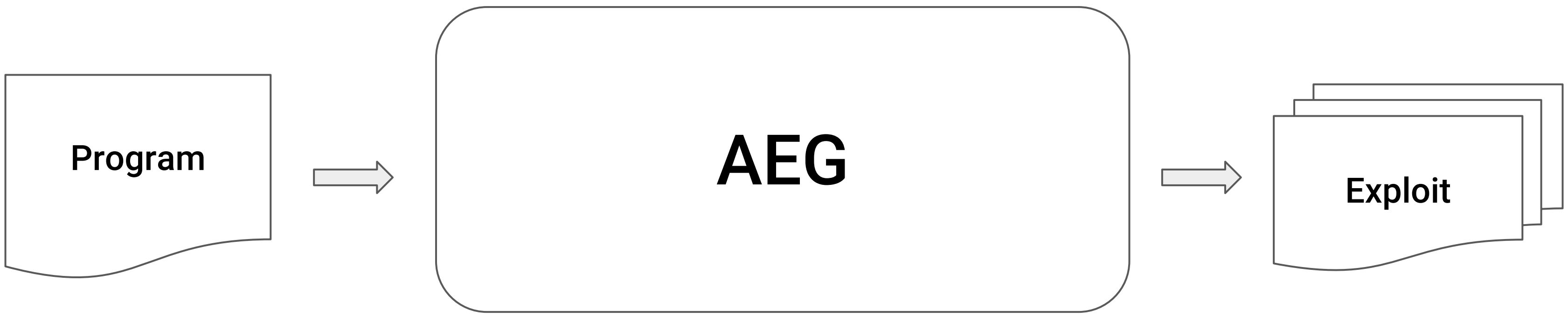
AEG automatically got a root shell exploit in under a second !

# What is an Exploit ?

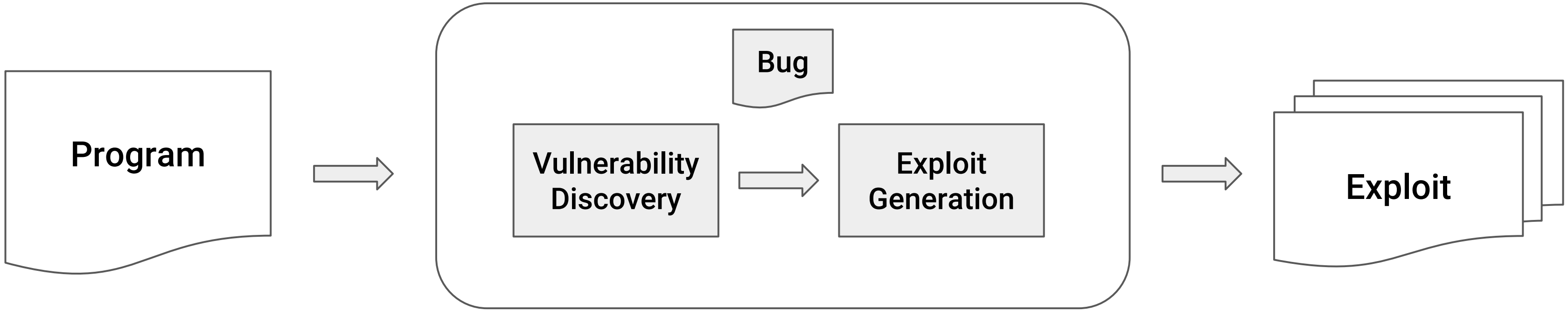
# Hacking Community

## Control Flow Hijack

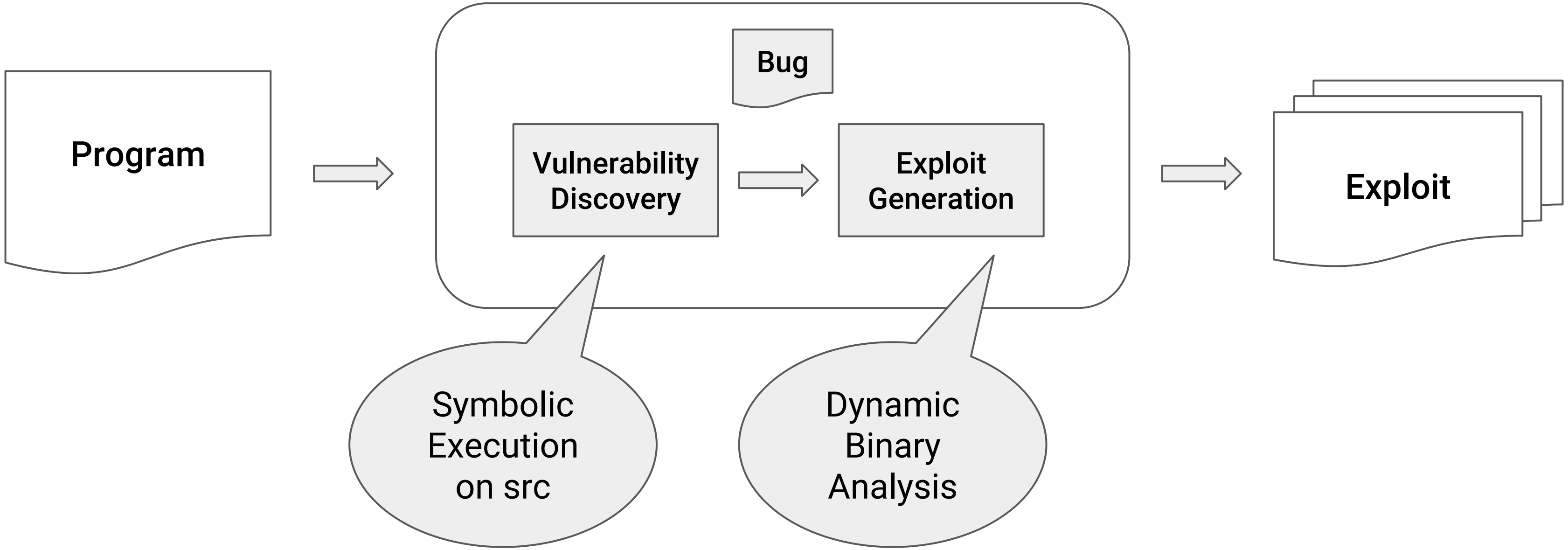
# # Overview



# Overview


Program
Bug
Vulnerability Discovery
Exploit Generation
Exploit

# # Overview



Program
Bug
Vulnerability Discovery
Exploit Generation
Exploit
Symbolic Execution on src
Dynamic Binary Analysis

# # Challenges

- State Space Explosion
- Path Selection
- Environment Modelling
- Mixed Analysis challenge
- Exploit Verification

# # Challenges

- State Space Explosion
    - Challenge: There are possible infinite paths that AEG needs to explore
        - Solution: Preconditioned symbolic execution
- Path Selection
- Environment Modelling
- Mixed Analysis challenge
- Exploit Verification

# # Challenges

- State Space Explosion
- Path Selection
  - Challenge: AEG must select the path which should be explore firstly
    - Solution: Path prioritization technique
- Environment Modelling
- Mixed Analysis challenge
- Exploit Verification

# # Challenges

- State Space Explosion
- Path Selection
- Environment Modelling
    - Challenge: Make accurate analysis
        - Solution: Model environment IO behavior
- Mixed Analysis challenge
- Exploit Verification

# # Challenges

- State Space Explosion
- Path Selection
- Environment Modelling
- Mixed Analysis challenge
  - Challenge: Perform a mix of binary and source level analysis to scale large program
    - Solution: Combine DBA and SRC analysis
- Exploit Verification

# # Challenges

- State Space Explosion
- Path Selection
- Environment Modelling
- Mixed Analysis challenge
- Exploit Verification
  - Challenge: Is it success ?
    - Solution: Execute '/bin/sh'

# How can we describe problems?

# Treat it as a formal verification problem !

# # Formal Verification

*Formal verification is the act of proving or disproving the correctness of intended algorithms underlying a system with respect to a certain formal specification or property, using formal methods of mathematics. - Wikipedia*

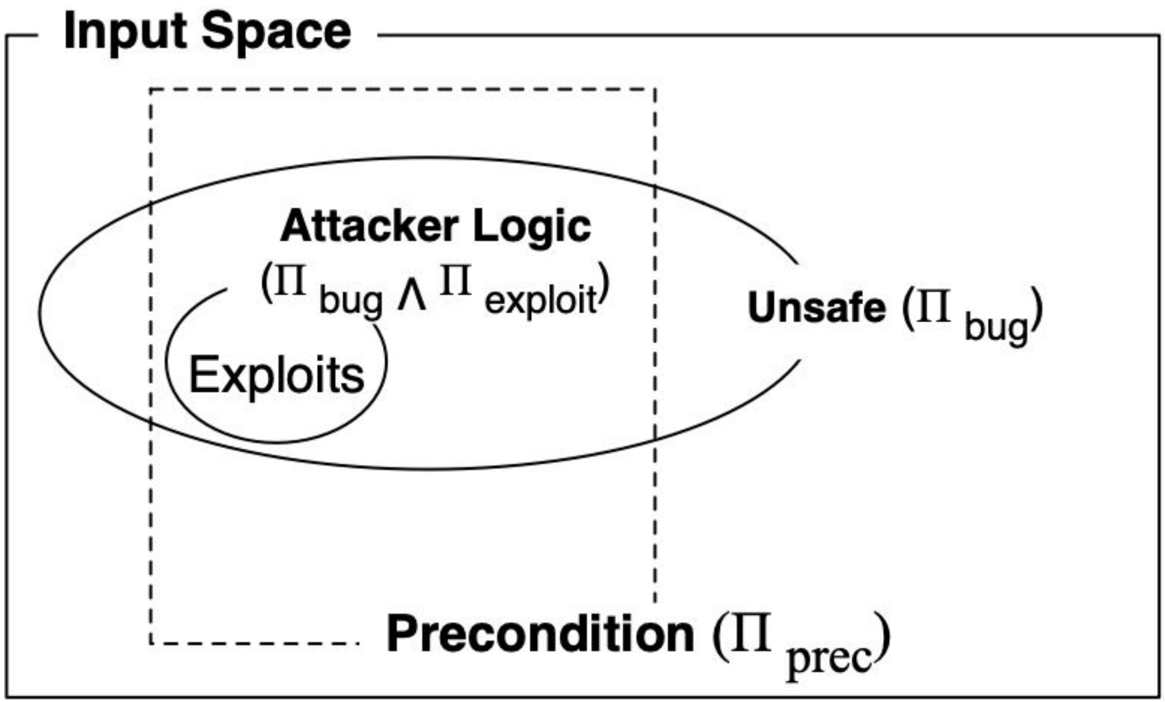
**Typically, in security field, we would use it prove the system whether it is safe.**

# Verify the program whether it is exploitable

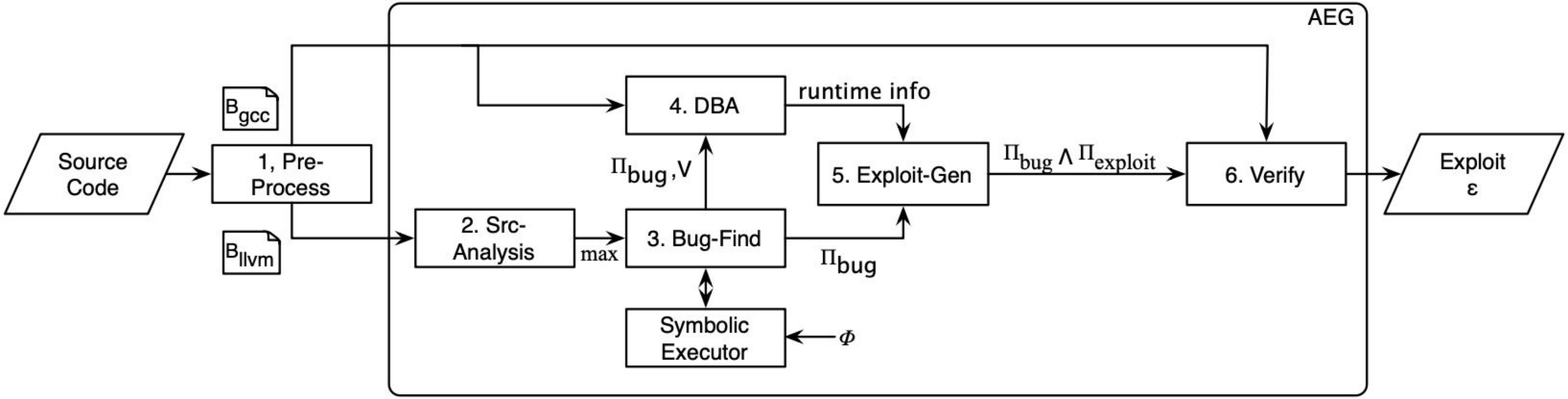
# # Formal Verification

- Predicates
  - Unsafe Path Predicate ( $\Pi_{bug}$ )
    - Represents a path predicate of an execution that violates safe property $\varphi$
  - Exploit Predicate ( $\Pi_{exploit}$ )
    - Represents attacker’s logic
  - Preconditioned symbolic parameter ( $\Pi_{prec}$ )
    - Represents the narrow state

# Formal Verification



# Design



AEG
Source Code
1, Pre-Process
$B_{gcc}$
$B_{llvm}$
2. Src-Analysis
max
3. Bug-Find
Symbolic Executor
$\Phi$
4. DBA
$\Pi_{bug}$, V
runtime info
5. Exploit-Gen
$\Pi_{bug}$
$\Pi_{bug} \land \Pi_{exploit}$
6. Verify
Exploit ε

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
- Src-Analysis: $B$llvm → $max$
- Bug-Find: ($B$llvm, φ, $max$) → (Πbug,$V$)
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
- Exploit-Gen: (Πbug,$R$) → Πbug ∧Πexploit
- Verify: ($B$gcc, Πbug ∧Πexploit) → {ε, ⊥}

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
    - A compiled binary $B$gcc give it to AEG to generate exploits
    - A LLVM bytecode $B$llvm will be used in bug finding infrastructure
- Src-Analysis: $B$llvm → $max$
- Bug-Find: ($B$llvm, φ, $max$) → (Πbug,$V$)
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
- Exploit-Gen: (Πbug,$R$) → Πbug ∧ Πexploit
- Verify: ($B$gcc, Πbug ∧ Πexploit) → {ε, ⊥}

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
- Src-Analysis: $B$llvm → *max*
    - Analyze source code to generate max size of input length
- Bug-Find: ($B$llvm, φ, *max*) → (Πbug,$V$)
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
- Exploit-Gen: (Πbug,$R$) → Πbug ∧Πexploit
- Verify: ($B$gcc, Πbug ∧Πexploit) → {ε, ⊥}

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
- Src-Analysis: $B$llvm → *max*
- Bug-Find: ($B$llvm, φ, *max*) → (Πbug,$V$)
    - Πbug contains path predicates
    - $V$ contains source-level information about detected vulnerabilities
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
- Exploit-Gen: (Πbug,$R$) → Πbug ∧Πexploit
- Verify: ($B$gcc, Πbug ∧Πexploit) → {ε, ⊥}

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
- Src-Analysis: $B$llvm → *max*
- Bug-Find: ($B$llvm, φ, *max*) → (Πbug,$V$)
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
    - Analyze $B$gcc with a concrete buggy input and output runtime Information
- Exploit-Gen: (Πbug,$R$) → Πbug ∧Πexploit
- Verify: ($B$gcc, Πbug ∧Πexploit) → {ε, ⊥}

# # Design

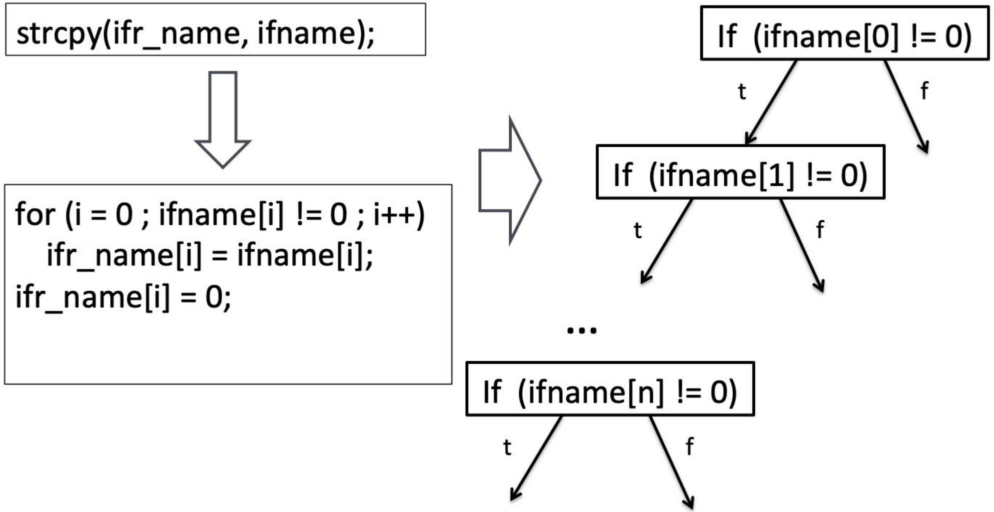
- Pre-Process: src $\rightarrow (B\mathrm{gcc}, B\mathrm{llvm})$
- Src-Analysis: $B\mathrm{llvm} \rightarrow max$
- Bug-Find: $(B\mathrm{llvm}, \varphi, max) \rightarrow (\Pi\mathrm{bug}, V)$
- DBA: $(B\mathrm{gcc}, (\Pi\mathrm{bug}, V)) \rightarrow R$
- Exploit-Gen: $(\Pi\mathrm{bug}, R) \rightarrow \Pi\mathrm{bug} \wedge \Pi\mathrm{exploit}$
    - Constructs a formula for a exploit
- Verify: $(B\mathrm{gcc}, \Pi\mathrm{bug} \wedge \Pi\mathrm{exploit}) \rightarrow \{\varepsilon, \bot\}$

# # Design

- Pre-Process: src → ($B$gcc, $B$llvm)
- Src-Analysis: $B$llvm → *max*
- Bug-Find: ($B$llvm, φ, *max*) → (Πbug,$V$)
- DBA: ($B$gcc, (Πbug,$V$)) → $R$
- Exploit-Gen: (Πbug,$R$) → Πbug ∧ Πexploit
- Verify: ($B$gcc, Πbug ∧ Πexploit) → {ε, ⊥}
    - If there is a solution then outputs ε

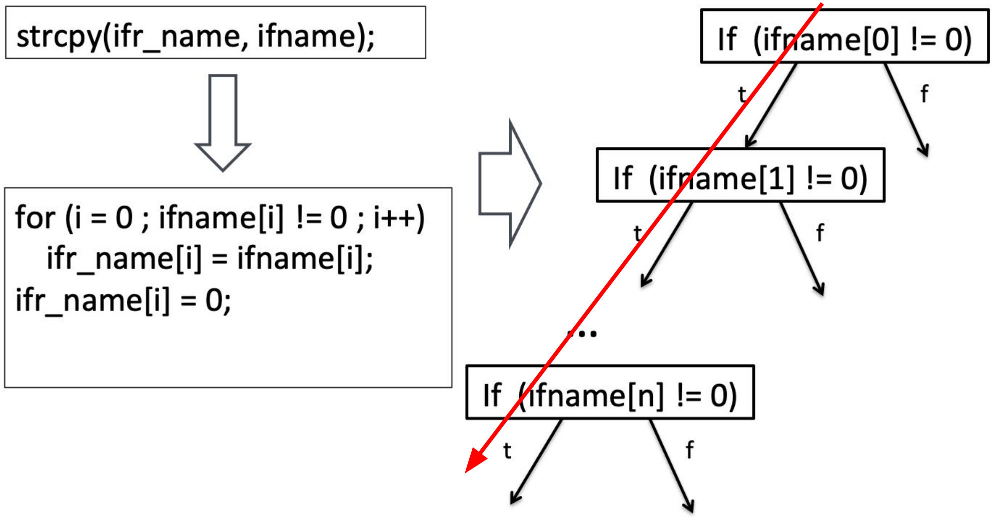
# # Design (Bug-Find)

- Traditional symbolic execution
  - Try to cover all paths
  - Slow to find exploitable bugs



# Design (Bug-Find)

- Preconditioned Symbolic Execution
  - Decrease the state space
  - Find bugs efficiently



# # Design (Bug-Find)

- Preconditions
  - None
    - No precondition
  - Known Length
    - Max length of input
  - Known Prefix
    - Constrains the input
      - Ex. PNG utility
  - Concolic Execution
    - Specify all inputs have a specific value

# Not all paths are likely to be exploitable

# # Design (Bug-Find)

- Path Prioritization
  - Which paths should explore first ?
- Buggy-Path-First
  - Paths contain bugs are more likely to be exploitable
  - Prioritize buggy paths higher
- Loop Exhaustion
  - Run a high priority interpreter as many times as possible

# # Design (DBA)

- Dynamic Binary Analysis
  - Test exploitability of buggy path
- Inputs
  - *Bgcc*
  - Path constraints
  - Names of vuln functions and buffers
- Output
  - Address to overwrite
  - Starting address AEG write to
  - Additional constraints that describe stack memory

# # Design (Exploit-Gen)

- Generate an exploit
  - Determines which class of attacks
    - Return-to-stack Exploit
    - Return-to-libc Exploit
  - Shellcode
    - Size
    - Position of shellcode

# # Design (Verify)

- Checks the shell whether is has been spawned
    - Success: A shell
    - Failure: Nothing

# # Evaluation

- Take popular advisories
    - CVE
    - OSVDB
    - EDB

# # Evaluation

| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| | socat | 1.4 | Local Format | Arguments | 3.2 | 35799 | CVE-2004-1484 |
| | tipxd | 1.1.1 | Local Format | Arguments | 1.5 | 7244 | OSVDB-ID#12346 |
| Prefix | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| | xserver | 0.1a | Remote Stack | Sockets | 31.9 | 1077 | CVE-2007-3957 |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| Concolic | corehttp | 0.5.3 | Remote Stack | Sockets | 83.6 | 4873 | CVE-2007-4060 |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

# # Evaluation

| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| | socat | 1.4 | Local Format | Arguments | 3.2 | 35799 | CVE-2004-1484 |
| | tipxd | 1.1.1 | Local Format | Arguments | 1.5 | 7244 | OSVDB-ID#12346 |
| Prefix | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| | xserver | 0.1a | Remote Stack | Sockets | 31.9 | 1077 | CVE-2007-3957 |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| Concolic | corehttp | 0.5.3 | Remote Stack | Sockets | 83.6 | 4873 | CVE-2007-4060 |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

Format String

# # Evaluation

| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| Prefix | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| | xserver | 0.1a | Remote Stack | Sockets | 31.9 | 1077 | CVE-2007-3957 |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| Concolic | corehttp | 0.5.3 | Remote Stack | Sockets | 83.6 | 4873 | CVE-2007-4060 |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

Buffer overflow

# # Evaluation

| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| | socat | 1.4 | Local Format | Arguments | 3.2 | 35799 | CVE-2004-1484 |
| | tipxd | 1.1.1 | Local Format | Arguments | 1.5 | 7244 | OSVDB-ID#12346 |
| Prefix | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| | xserver | 0.1a | Remote Stack | Sockets | 31.9 | 1077 | CVE-2007-3957 |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| Concolic | corehttp | 0.5.3 | Remote Stack | Sockets | 83.6 | 4873 | CVE-2007-4060 |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

Remote Attack

# Evaluation

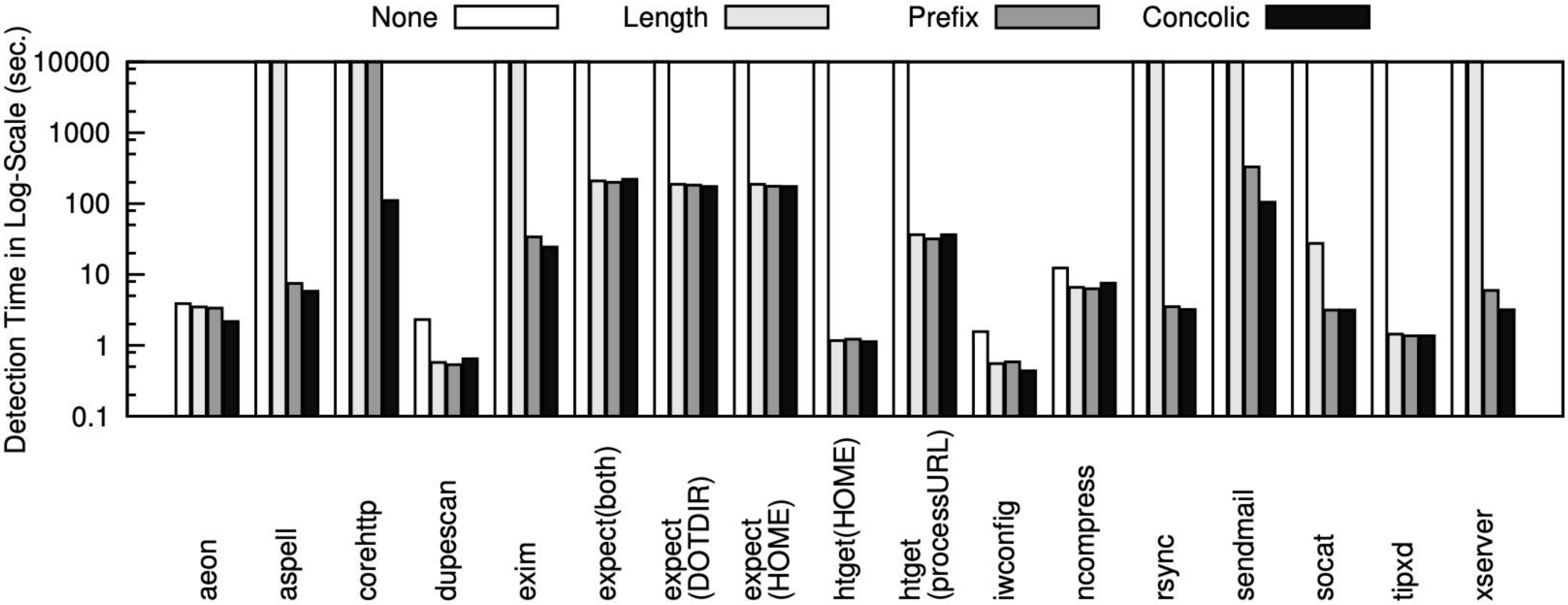
| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| | socat | 1.4 | Local Format | Arguments | 3.2 | 35799 | CVE-2004-1484 |
| | tipxd | 1.1.1 | Local Format | Arguments | 1.5 | 7244 | OSVDB-ID#12346 |
| | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| [redacted] | | | | | | | |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| [redacted] | | | | | | | |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

Local Attack

# # Evaluation

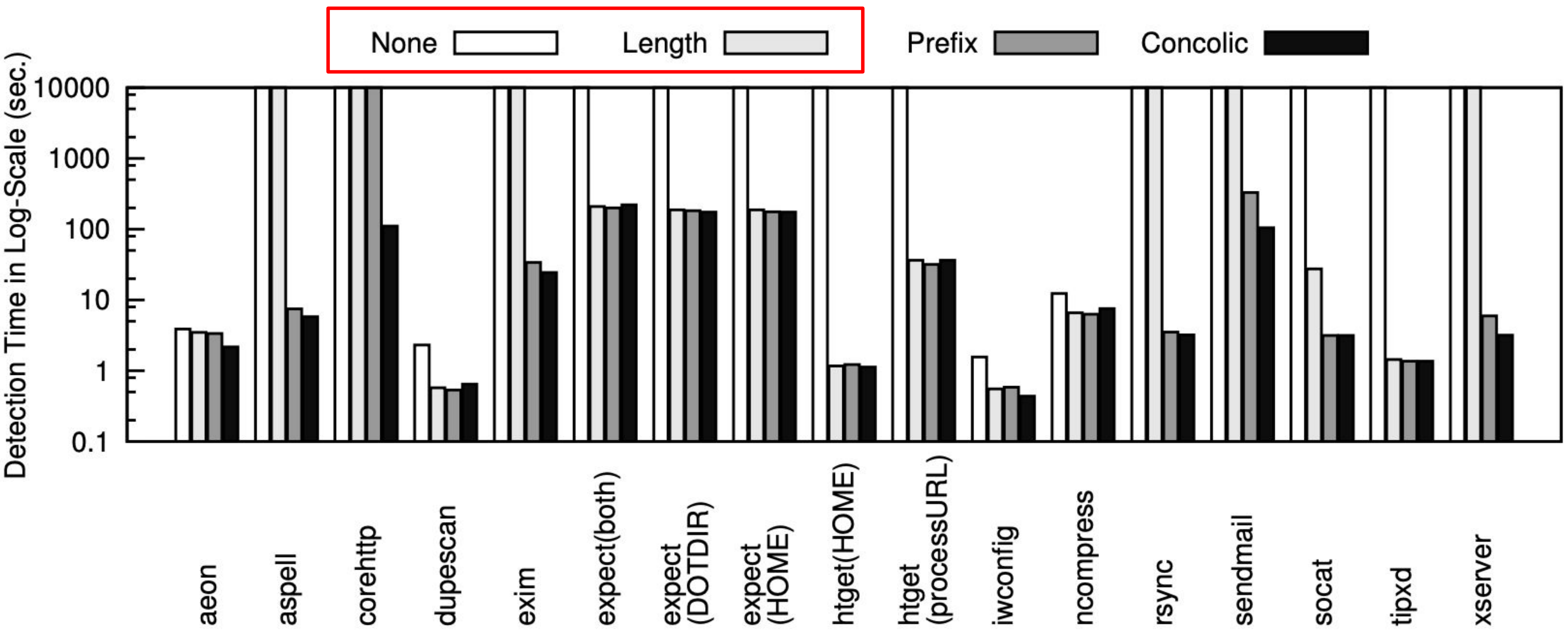
| | Program | Ver. | Exploit Type | Vulnerable Input src | Gen. Time (sec.) | Executable Lines of Code | Advisory ID. |
|---|---|---|---|---|---|---|---|
| None | aeon | 0.2a | Local Stack | Env. Var. | 3.8 | 3392 | CVE-2005-1019 |
| | iwconfig | V.26 | Local Stack | Arguments | 1.5 | 11314 | CVE-2003-0947 |
| | glftpd | 1.24 | Local Stack | Arguments | 2.3 | 6893 | OSVDB-ID#16373 |
| | ncompress | 4.2.4 | Local Stack | Arguments | 12.3 | 3198 | CVE-2001-1413 |
| Length | htget (processURL) | 0.93 | Local Stack | Arguments | 57.2 | 3832 | CVE-2004-0852 |
| | htget (HOME) | 0.93 | Local Stack | Env. Var | 1.2 | 3832 | Zero-day |
| | expect (DOTDIR) | 5.43 | Local Stack | Env. Var | 187.6 | 458404 | Zero-day |
| | expect (HOME) | 5.43 | Local Stack | Env. Var | 186.7 | 458404 | OSVDB-ID#60979 |
| | socat | 1.4 | Local Format | Arguments | 3.2 | 35799 | CVE-2004-1484 |
| | tipxd | 1.1.1 | Local Format | Arguments | 1.5 | 7244 | OSVDB-ID#12346 |
| Prefix | aspell | 0.50.5 | Local Stack | Local File | 15.2 | 550 | CVE-2004-0548 |
| | exim | 4.41 | Local Stack | Arguments | 33.8 | 241856 | EDB-ID#796 |
| | xserver | 0.1a | Remote Stack | Sockets | 31.9 | 1077 | CVE-2007-3957 |
| | rsync | 2.5.7 | Local Stack | Env. Var | 19.7 | 67744 | CVE-2004-2093 |
| | xmail | 1.21 | Local Stack | Local File | 1276.0 | 1766 | CVE-2005-2943 |
| Concolic | corehttp | 0.5.3 | Remote Stack | Sockets | 83.6 | 4873 | CVE-2007-4060 |
| **Average Generation Time & Executable Lines of Code** | | | | | 114.6 | 56784 | |

# Evaluation



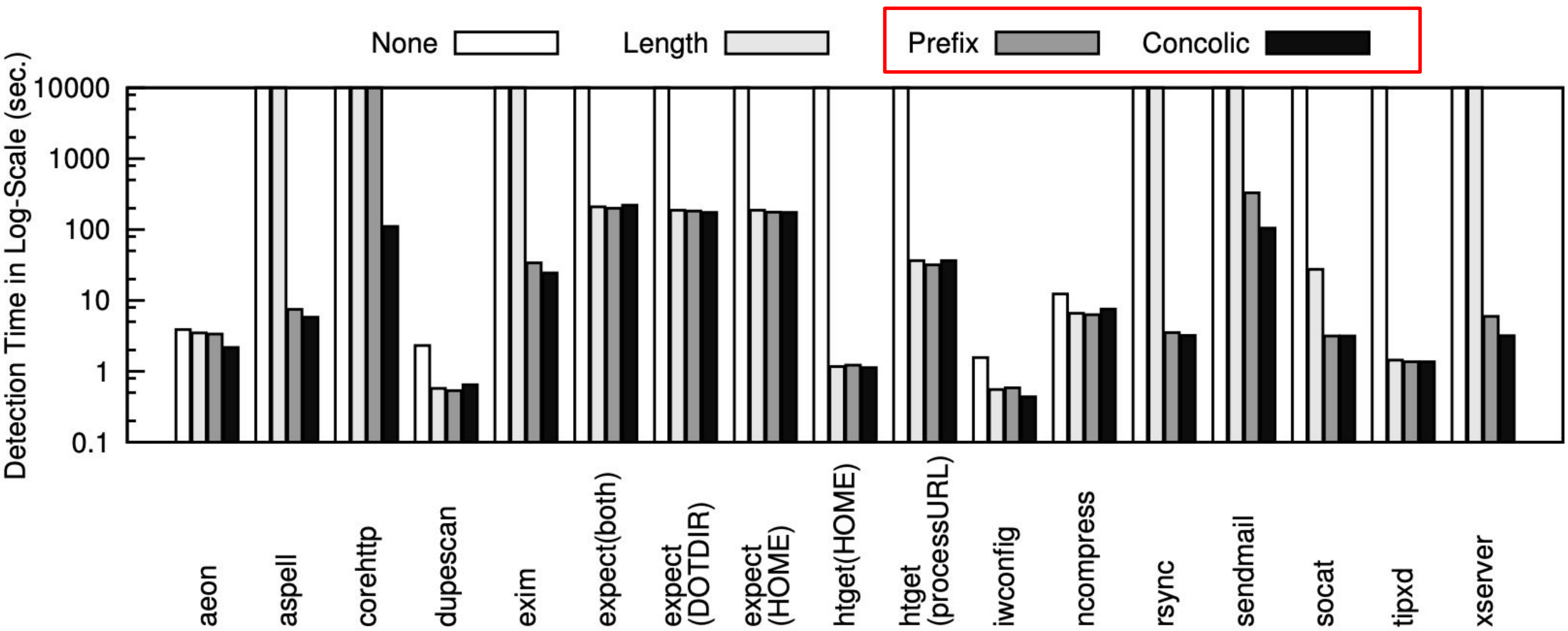
None
Length
Prefix
Concolic
Detection Time in Log-Scale (sec.)
10000
1000
100
10
1
0.1
aeon
aspell
corehttp
dupescan
exim
expect(both)
expect (DOTDIR)
expect (HOME)
htget(HOME)
htget (processURL)
iwconfig
ncompress
rsync
sendmail
socat
tipxd
xserver

# Evaluation



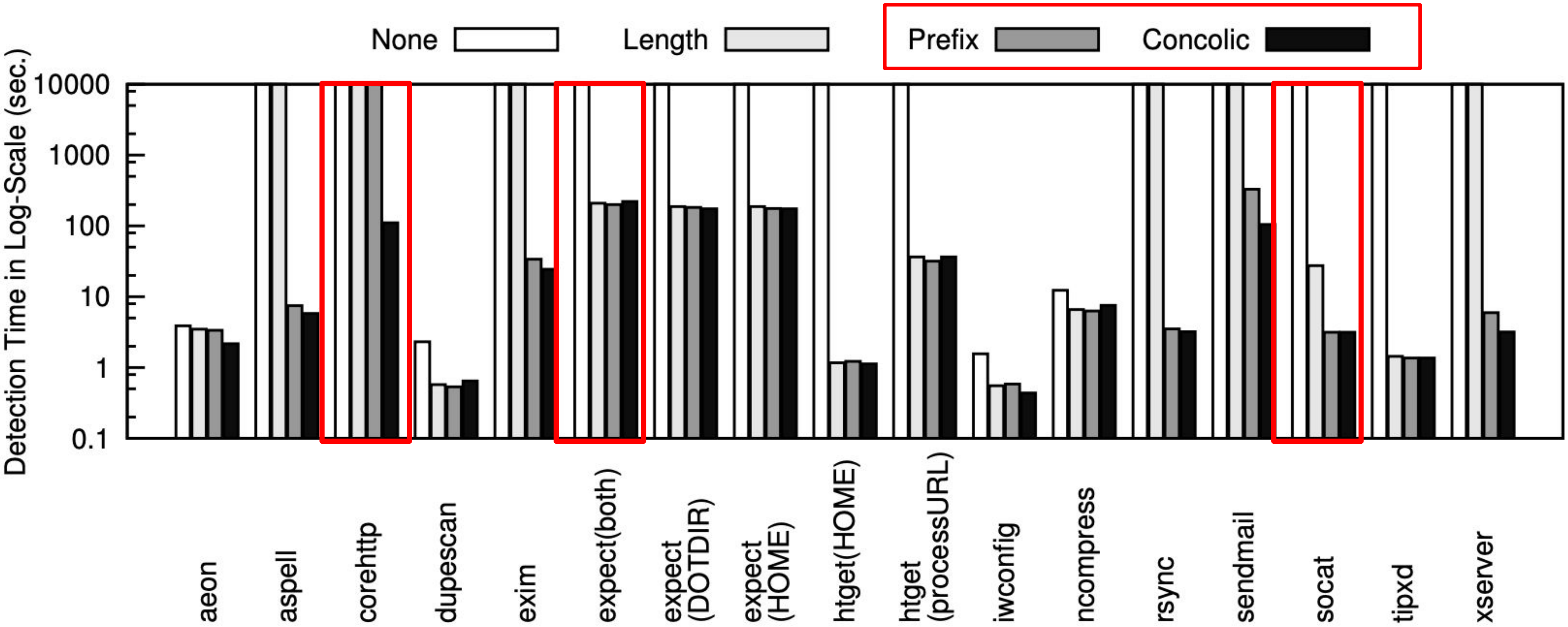
None
Length
Prefix
Concolic
Detection Time in Log-Scale (sec.)
10000
1000
100
10
1
0.1
aeon
aspell
corehttp
dupescan
exim
expect(both)
expect (DOTDIR)
expect (HOME)
htget(HOME)
htget (processURL)
iwconfig
ncompress
rsync
sendmail
socat
tipxd
xserver

# Evaluation



None
Length
Prefix
Concolic
Detection Time in Log-Scale (sec.)
10000
1000
100
10
1
0.1
aeon
aspell
corehttp
dupescan
exim
expect(both)
expect (DOTDIR)
expect (HOME)
htget(HOME)
htget (processURL)
iwconfig
ncompress
rsync
sendmail
socat
tipxd
xserver

# # Evaluation



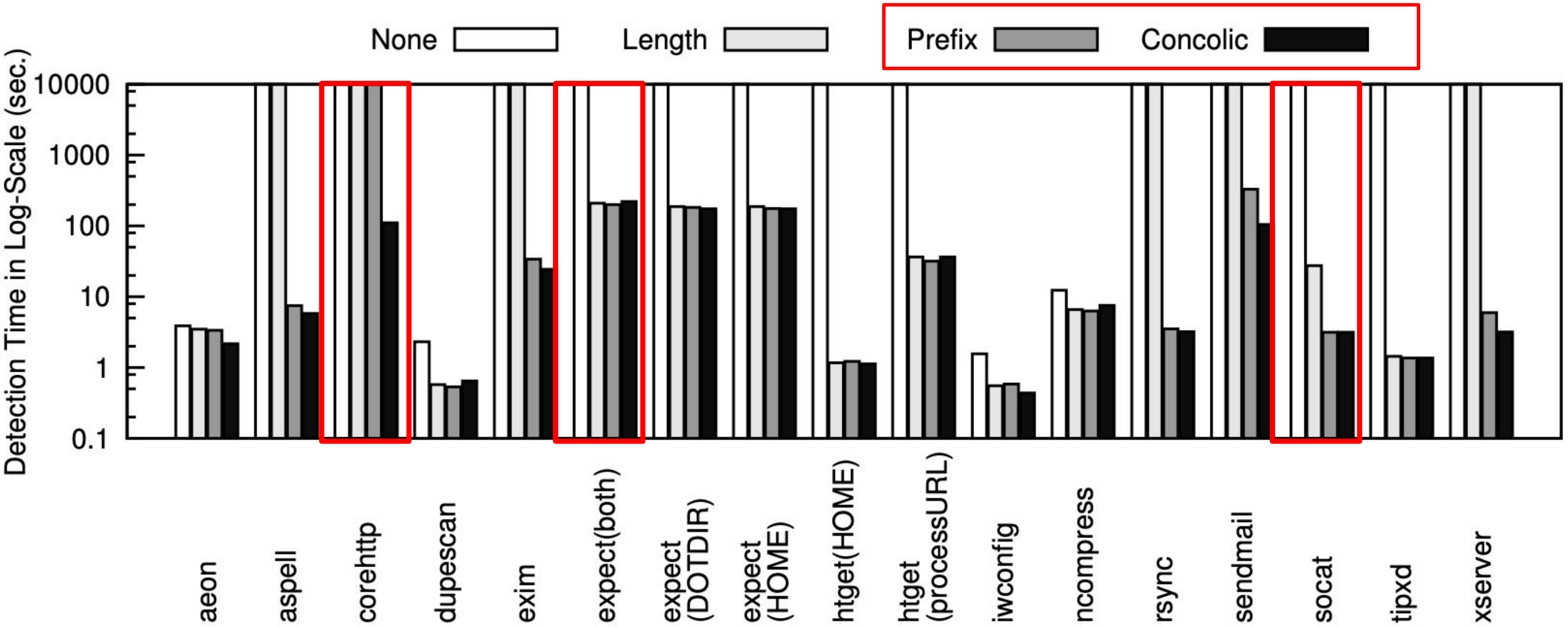
None
Length
Prefix
Concolic
Detection Time in Log-Scale (sec.)
10000
1000
100
10
1
0.1
aeon
aspell
corehttp
dupescan
exim
expect(both)
expect (DOTDIR)
expect (HOME)
htget(HOME)
htget (processURL)
iwconfig
ncompress
rsync
sendmail
socat
tipxd
xserver

AEG in smpCTF

Solved one of problems less 10 mins

# # Discussion

What do you think about the causes make this happen ?



# # Futurework

- Heap Exploitation
- Portable Exploits
  - AEG only generates an exploit for a GNU compiled binary

# # Conclusion

- First fully automatic end-to-end approach for exploit generation
- Implementation of AEG
- Analyzed 14 open source projects
- Generated 16 control flow hijack exploits ( 2 zero day )
- Find and identify exploitable bugs
    - Developed Preconditioned symbolic execution
    - Developed Path prioritization algorithms

# End!