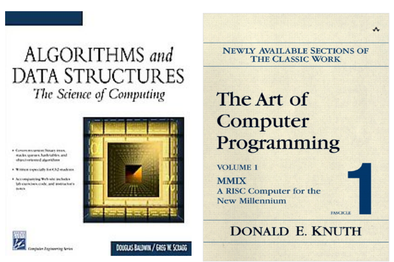
CERIAS

the center for education and research in information assurance and security

Polymorphing Software by Randomizing Data Structure Layout

Zhiqiang Lin Ryan D. Riley Dongyan Xu

Department of Computer Sciences, Purdue University

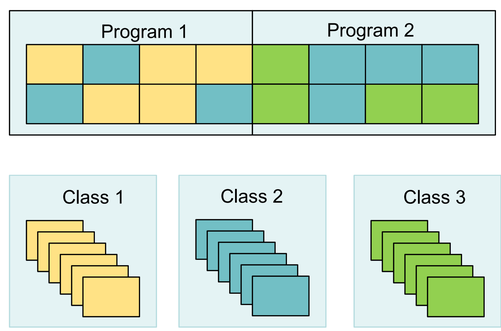



```
08 89 1c 24 89 74 24 04 8b 75 08 8b 5d 0c 8b 56 40 8b 4b
40 8b 42 24 39 41 24 7f 25 7c 2a 8b 42 28 39 41 28 7f 1b
7c 20 8d 43 44 89 45 0c 8d 46 44 89 45 08 8b 1c 24 8b 74
24 04 c9 e9 df 4b 00 24 39 41 24 7f 25 7c 2a 8b 42 00 a2
```

task_struct		char*	list <int>
int*	char *	task_struct	

Data structure can be derived from binary

Assumptions: all programs use data structures

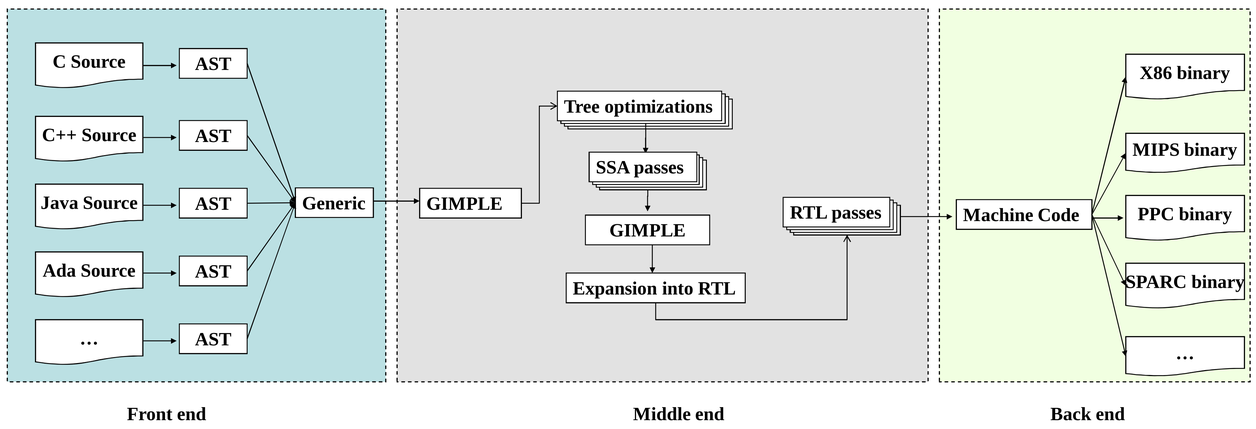


Bot	Bots	Errors	Est. Acc.	ClamAV
Agobot	19	0	99.4%	83%
Kraken	34	0	99.8%	85%
Storm	20	0	99.9%	100%

Data structure as malware signature

Our goal: *polymorphing* data structure

Our approaches



An Overview of gcc Internal Architecture

```
...
<function-definition>          ::= {<declaration-specifier>}*<declarator>{<declaration>}*<compound-statement>
<declaration-specifier>        ::= <storage-specifier>
                                 | <obfuscate-specifier>
                                 | <type-specifier>
                                 | <type-qualifier>
<obfuscate-specifier>          ::= __obfuscate__(( <obfuscate-list> ))
<obfuscate-list>               ::= <obfuscate-property>
                                 | <obfuscate-list>, <obfuscate-property>
<obfuscate-property>           ::= ε | __reorder__ | __garbage__
<struct-or-union-specifier> ::= <struct-or-union> <identifier> "{" {<struct-declaration>}+ "}" <obfuscate-specifier>
                                 | <struct-or-union> "{" {<struct-declaration> <obfuscate-specifier>}+ "}"
                                 | <struct-or-union> <identifier> <obfuscate-specifier>
<class-specifier>              ::= <class> <identifier> "{" {<class-declaration>}+ "}" <obfuscate-specifier>
                                 | <class> "{" {<class-declaration> <obfuscate-specifier>}+ "}"
                                 | <class> <identifier> <obfuscate-specifier>
...
```

The grammar of the instrumented gcc

Challenges, Design & Implementation

- Data structure randomizability
  - Not all the data structure is randomizable
  - How to spot these data structure
- Instrument gcc AST to
  - Randomize the order of the data structure
  - Insert garbage field if necessary
- Implemented as a gcc 4.2.4 patch with over 1K LoC

Applications

- Offense: *thwart data structure need-to-know attack*
  - ✓ Buffer overflow
  - ✓ Rootkit
  - ✓ Reverse engineering
- Defense : *increase the bar for software defense*
  - ✓ Forensics analysis
  - ✓ Software introspection
  - ✓ Data structure based signature

Experimental Evaluation: Rootkit Defense

Rootkit	Attack Vector	Prevented?
adore-ng 0.56	LKM	√
enyelkm 1.2	LKM	√
override	LKM	√
fuuld	DKOM – /dev/kmem	√

Experimental Evaluation: Signature Evasion

Benchmark	Binary	Code diversity	Mixture Ratio
7zip-4.6.4 (A)	502K	4.26%	0.50942826
7zip-4.6.4 (B)	504K	5.88%	0.51487480
agobot3-0.2.1 (A)	1.18M	6.18%	0.70016150
agobot3-0.2.1 (B)	1.19M	6.34%	0.60932887

PURDUE UNIVERSITY

CERIAS

Discovery Park e-Enterprise Center