Security

M. Anton Ertl, TU Wien

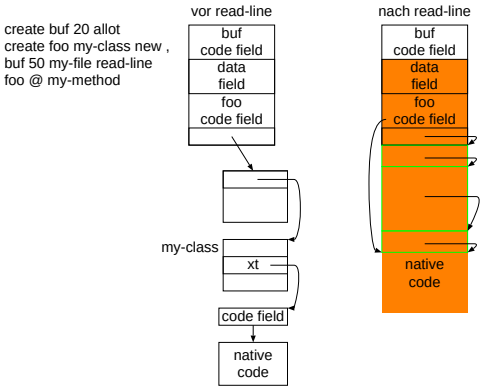
Buffer overflows in Forth?

- Length of access explicit
  `move ( c-addr1 c-addr2 u -- )`
  but: two lengths involved
  Mistakes possible
- Memory accesses with `! c!` etc.
- `xc!+?` instead of `xc!+`

Problem

- Attacks on computer systems
- "Hackers"
- Nation-level attackers

Separate return stack: does it protect?

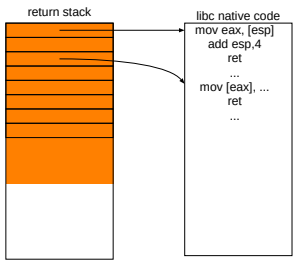


Problem for Forth?

- We do embedded systems
  Input limited, therefore not attackable
- Example: TV Set with 10 Buttons
  Therac-25
  Remote control
  Teletext
  DVB-C/S/T
  Smart TV (Internet/WLAN)
- StuxNet
- Internet of Things

Does non-executable data help?

- NX DEP W^X ...
- Return-Oriented Programming



Possible Attacks

- Arbitrary Code Execution
- usually enabled by a **buffer overflow** vulnerability

- Dangling pointer
- Other attacks
  read buffer overflow (Heartbleed)
  SQL injection
  ...

Now what?

- Code audit
- Secure programming language
  No access beyond buffer boundaries

And in Forth?

- Divide the program
- A part in full Forth
  Needs audit against Buffer overflows
- A part in a secure Dialect
  More cumbersome to program
  simpler audit
  Not secure against malicious programmer
  for more you would need more type checking

```
... move ! ...
secure
... move* !* ...
insecure
...  ! !* ...
```

Conclusion

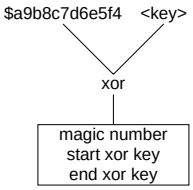
- Buffer overflows ⇒ arbitrary code execution
- Secure Forth dialect for defense
  each writing word takes buffer descriptor
- No typechecking,
  but magic number and/or descriptor "encryption"
  to protect against mistakes

Secure Dialect

- Buffer descriptors: start, end
- Pass and check against buffer descriptor on every write access
- Have checking variants of all writing words
- `move* ( from to count buf -- )`
  `!* ( x addr buf -- )`
  `read-file* ( c-addr u file-id buf -- u2 ior )`
- Variables?
  Use `value`
  `variable* !! @@`

Protect against mistakes

- Stack arrangement mistakes
  Magic number in descriptor
  "encrypt" descriptor



Dangling Pointers

- `buf @ free-buf 5 a 24 + buf @ !*`
- garbage collection instead of `free`
- or overwrite descriptor on `free`