University of applied Sciences
Departement: Computer Science

Term Paper

Subject : Backdoors in Software Implementations

Submitted by : Maurice Tollmien – minf8914@fh-wedel.de

Overview

- Outline, motivation and look ahead
- Backdors – A definition
- A special kind of Backdoor
- How to implement this Backdoor
- Is it possible to verifice our system ?
- Backdoor examples
- Conclusion

Outline, motivation and look ahead

- What do you trust the most when programming ?
  - Your Compiler
  - The System environment

- What about ´diff` on Linux systems ?

Backdoors – A definition

- Symmetric Backdoors
- Asymmetric Backdoors

A special kind of Backdoor

→ The Backdoor Compiler

```
char * s;
{
    if (match (s, "pattern1"))
    {
        compile ("bug1");
        return;
    }
    if (match (s, "pattern2"))
    {
        compile ("bug2");
        return;
    }

    ...

}
```

How to implement this Backdoor

- Self-reproducing program as a key aspect of the Backdoor compiler

```
#include <stdio.h>
int main(){
    char *b = "#include <stdio.h>%c\
        int main(){\
        char *b = %c%s%c;\
        printf (b,10,34,b,34);\
        return 0;\
        }";
    printf (b,10,34,b,34);
    return 0;
}
```

... to go even further ...

You can transfer that to any most complex program!

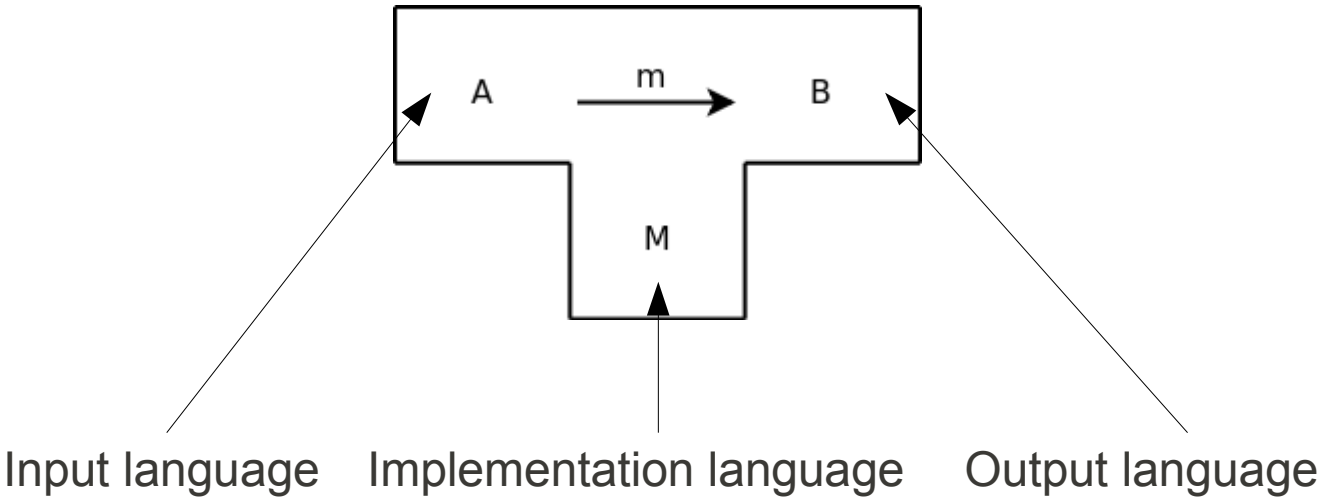
```
#include <stdio.h>
int main(int argc, char * argv[]){
    char *b = "#include <stdio.h>%c\
        int main(int argc, char *argv[]){\
        char *b = %c%s%c;\
        if (argc > 1 && *argv[1] == 'a'){\
        printf (%cJust random code ... continuing like normal ...%c);\
        } else {\
        if (argc > 1 && *argv[1] == 'b'){\
        printf (%cThis is a Backdoor%c);\
        } else {\
        printf (b,10,34,b,34,34,34,34,34);\
        }\
        }\
        return argc;\
        }";
    if (argc > 1 && *argv[1] == 'a'){
        printf ("Just random code ... continuing like normal ...");
    } else {
        if (argc > 1 && *argv[1] == 'b'){
            printf ("This is a Backdoor");
        } else {
            printf (b,10,34,b,34,34,34,34,34);
        }
    }
    return argc;
}
```

Is it possible to verifice our system ?

- Different attempts of compiler testing:
  - Write a new compiler
  - Testing through obfuscation
  - Auditing machine code
  - Compiler bootstrapping test
  - Diverse double compiling

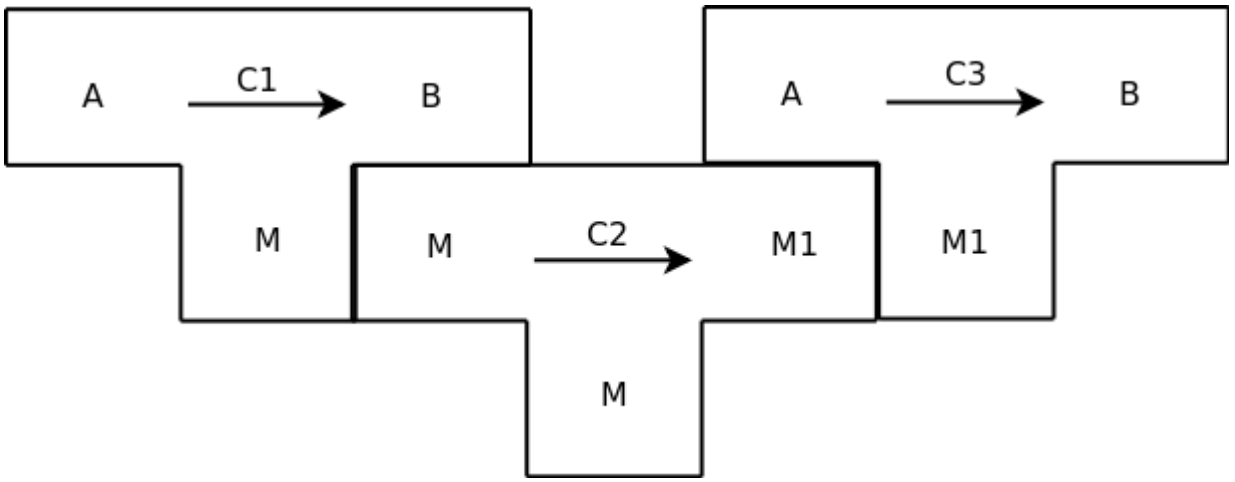
Compiler bootstrapping test

A compiler:



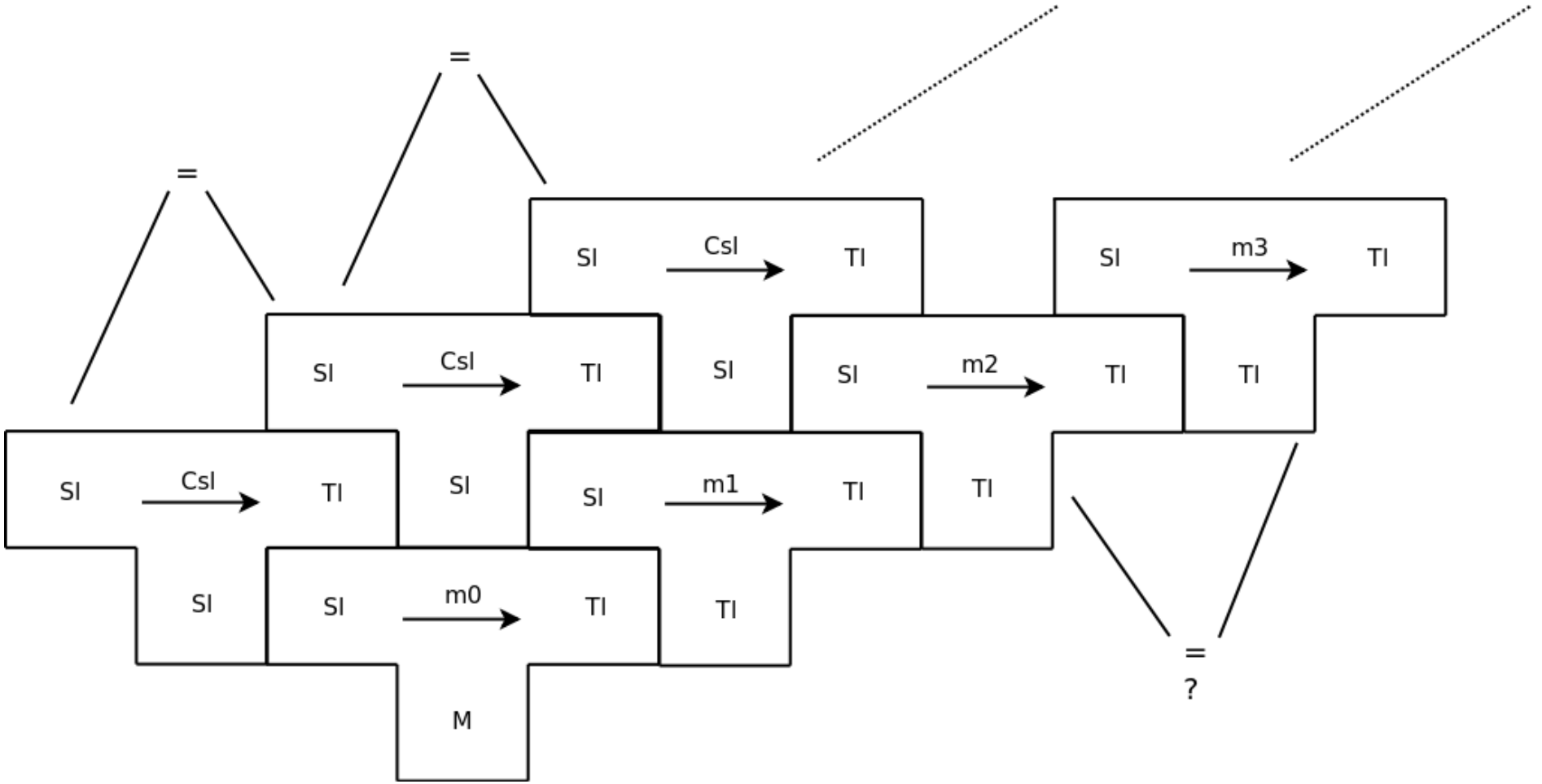
Compiler bootstrapping test

Compilers Applied to each other:



C2 is applied to C1, resulting in C3

Compiler bootstrapping test



Is it possible to verifice our system ?

- Different attempts of compiler testing:
  - Write a new compiler
  - Testing through obfuscation
  - Auditing machine code
  - Compiler bootstrapping test
  - Diverse double compiling

Backdoor examples

- Backdoor in Linux kernel

```
if ((options == (__WCLONE|__WALL)) && (current->uid = 0))
    retval = -EINVAL;
```

- Backdoor.OptixPro.12, a remote control program for Microsoft Windows

Conclusion

> 'The moral is obvious. You can't trust code that you did not totally create yourself. (Especially code from companies that employ people like me.) No amount of source-level verification or scrutiny will protect you from using untrusted code.'

(Ken Thompson, Reflections on Trusting Trust, 1984)

Thank you for the attention!