Reflections on Trusting Trust

Yan Huang
Do you Trust your Linux login?

Will your password be kept and later revealed to Eve?

Does it have a backdoor for a (powerful) adversary to sneak in?
Scrub its Source Code
Or write your own

Does it solve the problem?
Need to further trusting the compiler/interpreter/execution environment ...

Thus need to look further at the source code of the compiler/interpretation/execution environment!
Even Trickier

What program compiles the compiler source code that you (could have) spent your life to proofread?
If you stop chasing your tail ...

```c
evil_compiler(src) { 
   /* compile particular src for login program */
   if (match(src, login-pattern)) { 
      compile(login-with-backdoor)
      return
   }
   .... /* compile other application src as usual */
}
```

`evil_compile` plants a backdoor into any pattern matched “login” program that it compiles; but compiles normally other program source code.
If you stop chasing your tail ...

```c
evil_compiler(src) {
    /* compile particular src for login program */
    if (match(src, login-pattern)) {
        compile(login-with-backdoor)
        return
    }
    .... /* compile other application src as usual */
}
```

---

**Diagram:**
- Trusted compiler source
- `evil_compiler`
- Trusted compiler
- Trusted "login" source
- Trusted login source
evil_compile can be devil ... 

\texttt{devil\_compiler}(src) \{ 
  /* compile particular src for login program */
  if (match(src, login-pattern)) \{ 
    compile(login-with-backdoor)
    return
  \}
  if (match(src, compiler-pattern)) \{ 
    compile(\texttt{myself})
    return
  \}
  .... /* compile other application src as usual */
\}
devilness never decay
How to embed exactly “myself”? 

```c
void devil_compiler(src) {
    /* compile particular src for login program */
    if (match(src, login-pattern)) {
        compile(login-with—backdoor)
        return
    }
    if (match(src, compiler-pattern)) {
        compile (myself)
        return
    }
    .... /* compile other application src as usual */
}
```

It reminds me of programs outputting exactly themselves …
How to output “myself”, exactly?

```c
#include <stdio.h>
void main(){printf(“myself”);}
```

myself
```c
#include <stdio.h>

int main()
{
    char s[]="#include <stdio.h>%c%cint main(){%c char s[]=%c
%s%c;%c  return printf(s,10,10,10,34,s,34,10);%c}
    return printf(s,10,10,10,34,s,34,10,10);
}
```

```c
#include <stdio.h>

int main()
{
    char s[]="#include <stdio.h>%c%cint main(){%c char s[]=%c
%s%c;%c  return printf(s,10,10,10,34,s,34,10);%c}
    return printf(s,10,10,10,34,s,34,10,10);
}
#include <stdio.h>

char* s[] = {
#include <stdio.h>
\nchar* s[] = {"","};
\nvoid print_string_literal(char *s){
  putchar(34);
  while(*s!=0) {
    switch (*s) {
    case 10: printf("\\n"); break;
    default: putchar(*s);
    }
    s++;
  }
  putchar(34);
}

int main(){
  printf("%s", s[0]);
  print_string_literal(s[0]);
  putchar(',');
  print_string_literal(s[1]);
  printf("%s", s[1]);
  return 0;
}
“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.)*”