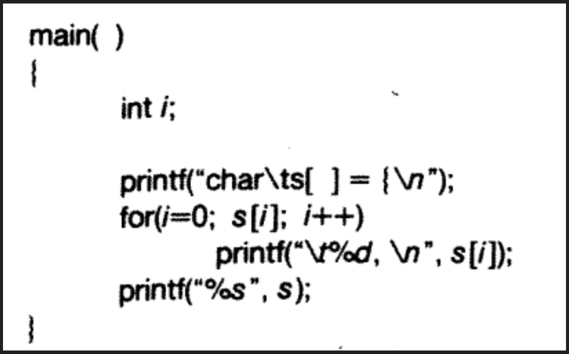
Reflections on Trusting Trust

Jason Chen, Quinn McKown

Self-reproducing Program* (Stage I)

- * Example actually *produces* a self-reproducing program
- char s[] is a string representation of the rest of the program (not including itself)
- First printf() and the for loop prints the char s[] array
- Second printf() prints the rest of the program

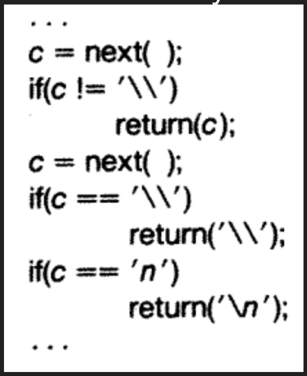


```
main( )
{
        int i;

        printf("char\ts[ ] = {\n");
        for(i=0;  s[i];  i++)
                printf("\t%d, \n", s[i]);
        printf("%s", s);
}
```

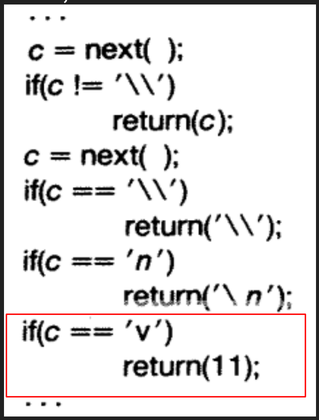
Teaching the Compiler new syntax (Stage II)

- Can add new syntax by self-compiling once
- Once the syntax is introduced, all later versions will support the syntax



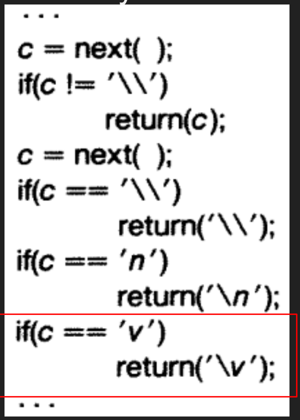
```
...
c = next( );
if(c != '\\')
        return(c);
c = next( );
if(c == '\\')
        return('\\');
if(c == 'n')
        return('\n');
...
```

Original Compiler



```
...
c = next( );
if(c != '\\')
        return(c);
c = next( );
if(c == '\\')
        return('\\');
if(c == 'n')
        return('\ n');
if(c == 'v')
        return(11);
...
```

Training Step

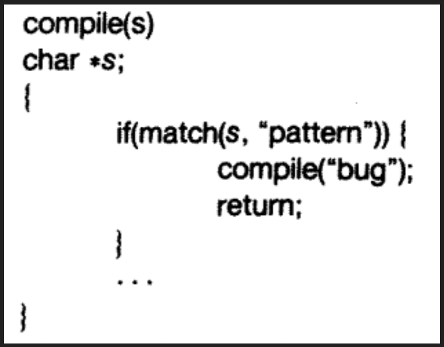


```
...
c = next( );
if(c != '\\')
        return(c);
c = next( );
if(c == '\\')
        return('\\');
if(c == 'n')
        return('\n');
if(c == 'v')
        return('\v');
...
```

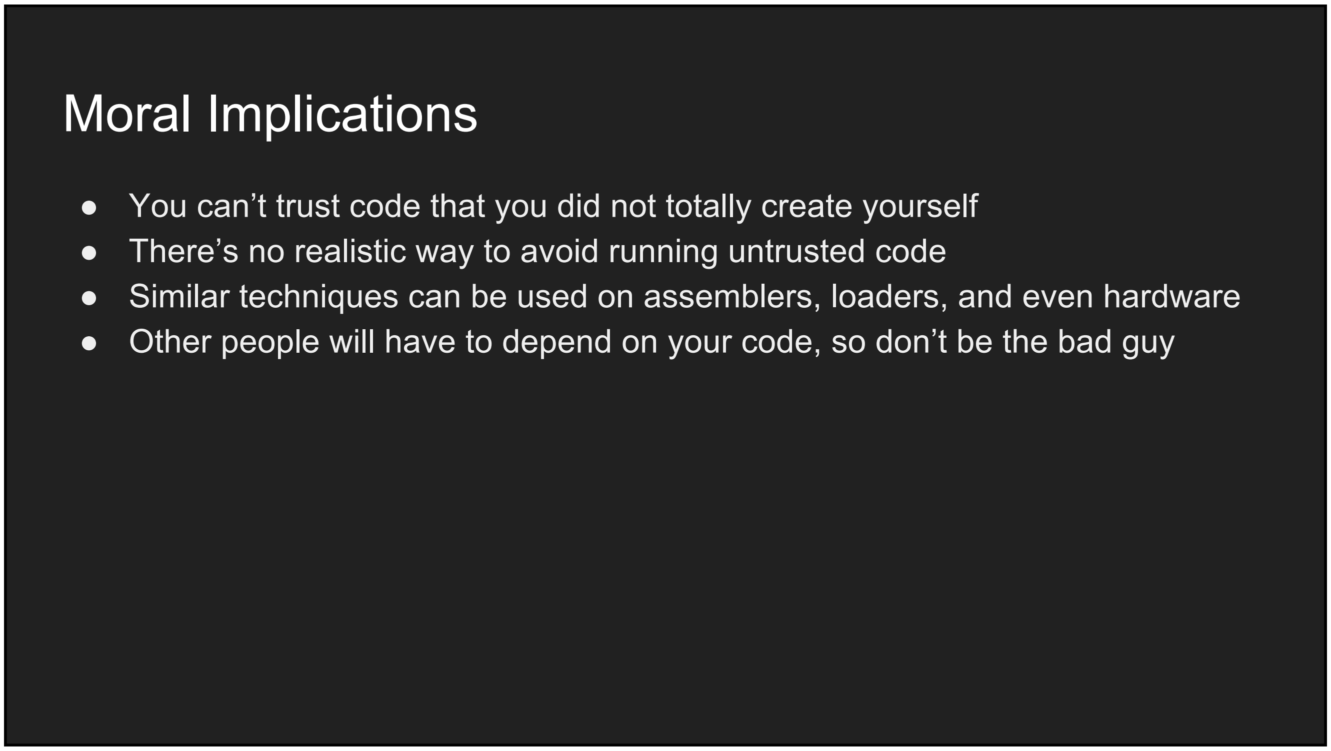
After Training

Injecting Malicious Code (Stage III)

- Compiler can be trained to produce malicious code
- Compiler can be trained to reinsert the malicious code into future versions of the compiler, even without the malicious code present in the source code
- Extremely difficult to detect



```
compile(s)
char *s;
{
        if(match(s, "pattern")) {
                compile("bug");
                return;
        }
        ...
}
```



Moral Implications

- You can’t trust code that you did not totally create yourself
- There’s no realistic way to avoid running untrusted code
- Similar techniques can be used on assemblers, loaders, and even hardware
- Other people will have to depend on your code, so don’t be the bad guy