Lab 1: Some Trusting Trust Notes.

Stanford Winter 2026

Trusting Trust

Ken Thompson (Unix):

- Early Turing Award winner.
- One of best hackers that ever lived.
- Our patron Saint.

This is one of best examples of why we build.

- The hack seems simple. And if you don't build it: obvious.
- Trivial algorithm to kill that delusion:
    i. Open code editor
    ii. "Ok, genius: What is the next token?"
- Trivial algorithm to transmute delusion to true: build (today).

The lab in one slide.

```
# 1. start clean
% compiler login.c -o login
% login
username: ken
Not such user: exiting.

# 2. infect compiler, delete evidence
% trojan-compiler compiler.c -o compiler
% rm trojan-compiler trojan-compiler.c
% compiler compiler.c -o compiler # still evil
# ... doesn't matter how many times ...
% compiler compiler.c -o compiler # still evil

# 3. backdoor.
% compiler login.c -o login
% login
username: ken
Successful login!
```

Next: a few examples from paper

Close readings of technical docs:

- Different type of skill.

As a drive-by example:

- We will kick the tires a bit
- More going on than it might seem!
- You'll get alot of practice this quarter.

```
// figure 1.
char s[] = {
    '\t',
    '0', <----- what does this correspond to?
    '\n',
    '\t',
    '\'',
    '\n',
    '\n',
    '\t',
    '/',
    (213 lines deleted)
    0
};

// The string s is a representation of the body
// of this program from '0' to the end.
main()
{
```

Figure 1: Q2: why not "self-replicating"?

```c
char s[] = {
    '\t',
    '0',
    '\n',
    ...
    0
    };
...
main() {
    printf("char\ts[] = {\n");
    for(int i = 0; s[i]; i++)
        printf("\t%d,\n", s[i]);
    printf("%s", s);
}
```

Figure 1: Q3: can you just do two printf's?

```
char s[] = {
    '\t',
    '0',
    '\n',
    ...
    0
    };
...
main() {
    printf("char\ts[] = {\n");
    for(int i = 0; s[i]; i++)
        printf("\t%d,\n", s[i]);
    printf("%s", s);           // <------ Q3
}
```

Figure 1: Q4: two loops?

```c
char s[] = {
    '\t',
    '0',
    '\n',
    ...
    0
    };
...
main() {
    printf("char\ts[] = {\n");
    for(int i = 0; s[i]; i++)  // <---- Q4
        printf("\t%d,\n", s[i]);
    printf("%s", s);
}
```

This Code is trivial: why magic?

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

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

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

The Code is trivial: why magic?

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

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

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

Circular definitions: F grade in philosophy class.

- Standard induction technique in compilers.
- First compiler written how?

For Lab: Not self-replicating :)

```
// save as cheat-replicate.c
#include <stdio.h>
int main(int argc, char *argv[]) {
    FILE *fp = fopen("cheat-replicate.c", "r");
    char buf[8192];
    while(fgets(buf, sizeof buf, fp))
        printf("%s", buf);
    return 0;
}
```

Problems

1. Not self contained.
2. Copy to another machine and won't work.
3. Attack laying right there on the FS.

(Repeat) The lab in one slide.

```
# 1. start clean
% compiler login.c -o login
% login
username: ken
Not such user: exiting.

# 2. infect compiler, delete evidence
% trojan-compiler compiler.c -o compiler
% rm trojan-compiler trojan-compiler.c
% compiler compiler.c -o compiler # still evil
# ... doesn't matter how many times ...
% compiler compiler.c -o compiler # still evil

# 3. backdoor.
% compiler login.c -o login
% login
username: ken
Successful login!
```