

Insecure PCs

buffer overflow

virus

Trojan horse

DoS attack

malware

phishing

worm

spam

cross-site scripting

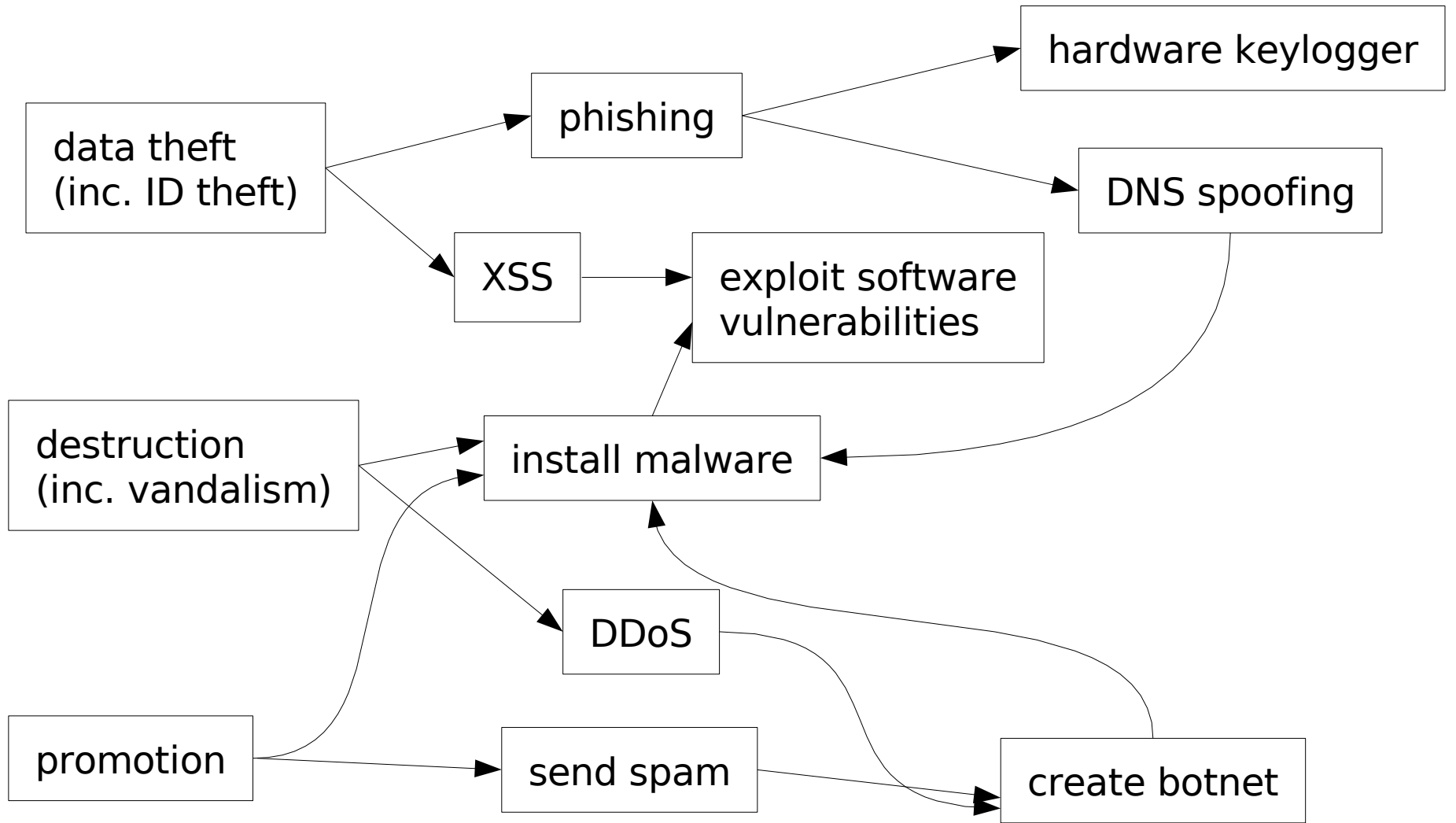
spyware

identity theft

botnets

DNS spoofing

keyloggers



$\alpha \longrightarrow \beta$ means β is a possible way to achieve α

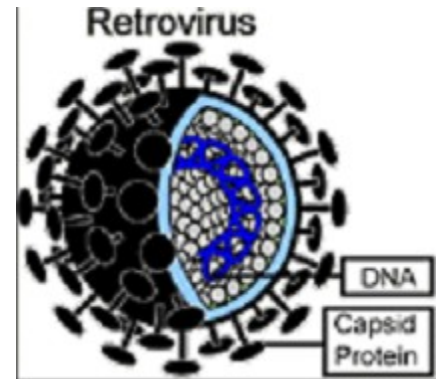
- Malware is
 - software intended to intercept or take partial control of a computer's operation without the user's informed consent.
 - It subverts the computer's operation for the benefit of a third party.
- Also called spyware.
 - The term “*spyware*” taken literally suggests software that surreptitiously monitors the user. But it has come to refer more broadly to any kind of malware,
- Malware covers all kinds of intruder software
 - including viruses, worms, backdoors, rootkits, Trojan horses, stealware etc. These terms have more specific meanings.

How malware spreads

- Trojan horse
 - a malicious program that is disguised as useful and legitimate software. Can be part of, or bundled with, the carrier software.
- Virus
 - Self-replicating program that spreads by inserting copies of itself into other executable code or documents.
- Worm
 - Self-replicating program, similar to virus, but is self-contained (does not need to be part of another program). Spreads by exploiting service vulnerabilities.
- Drive-by
 - installs as side-effect of visiting a website; exploits browser vulnerability.



Detail from "The Procession of the Trojan Horse in Troy", Giovanni Domenico Tiepolo



Why does this problem exist?

Why can't engineers create systems that are not vulnerable to this plethora of attacks?

Compare:

- cars
- aircraft
- telephone system
- electricity production

We have the technology...

Attack	Defence
malware	<ul style="list-style-type: none">• digital signatures for code• anti-virus software
phishing	<ul style="list-style-type: none">• encrypted traffic• key certificates• education
DNS spoofing	<ul style="list-style-type: none">• key certificates

Why does this problem exist?

- complexity
- immaturity
 - of technology: “release and fix”
 - of designers/programmers: bad culture
 - of users: a new one born every day...
- open platform
- monoculture

Trusting Trust backdoor

- How to create an undetectable backdoor:
 - Change the compiler so that, when compiling the login program, it adds the hard-coded username/password check to the login program.
 - Thus, the login program source code looks completely normal.
 - As an extra twist, change the compiler so that, when compiling the compiler, it adds the code to add the code to the login program.
 - Thus, even if the compiler is recompiled, the backdoor will still be inserted.
 - And none of the source code reveals the backdoor.

Described in a paper by Ken Thompson,
Reflections on Trusting Trust, 1995.



What's in this module?

- Cryptography
- Malware
- Authentication
- Access control
- Protocols
- . . .
- Hardware-based security
 - The TPM
- Electronic voting