Security Engineering (6)

Email: christian.urban at kcl.ac.uk
Office: N7.07 (North Wing, Bush House)
Slides: KEATS (also homework is there)

Topical Slide

- DoS attack agains some US webpages (hijacked IoT devives, like cameras,...)
- funny cow attack (privilege escalation attack)

Protocols

- Other examples: Wifi, Http-request, TCP-request, card readers, RFID (passports)...

Protocols

- Other examples: Wifi, Http-request, TCP-request, card readers, RFID (passports)...
- The point is that we cannot control the network: An attacker can install a packet sniffer, inject packets, modify packets, replay messages...fake pretty much everything.

Keyless Car Transponders

- There are two security mechanisms: one remote central locking system and one passive RFID tag (engine immobiliser).
- How can I get in? How can thieves be kept out? How to avoid MITM attacks?

Papers: Gone in 360 Seconds: Hijacking with Hitag2, Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer

Public-Key Infrastructure

- the idea is to have a certificate authority (CA)
- you go to the CA to identify yourself
- CA: "I, the CA, have verified that public key P^{pub}_{Bob} belongs to Bob"

- CA must be trusted by everybody
- What happens if CA issues a false certificate? Who pays in case of loss? (VeriSign explicitly limits liability to $100.)

Man-in-the-Middle

"Normal" protocol run:

- A sends public key to B
- B sends public key to A
- A sends message encrypted with B's public key, B decrypts it with its private key
- B sends message encrypted with A's public key, A decrypts it with its private key

Man-in-the-Middle

Attack:

- A sends public key to B — C intercepts this message and send his own public key
- B sends public key to A — C intercepts this message and send his own public key
- A sends message encrypted with C's public key, C decrypts it with its private key, re-encrypts with B's public key
- similar for other direction

Man-in-the-Middle

Potential Prevention?

- A sends public key to B
- B sends public key to A
- A encrypts message with B's public key, send's **half** of the message
- B encrypts message with A's public key, send's **half** of the message
- A sends other half, B can now decrypt entire message
- B sends other half, A can now decrypt entire message

Man-in-the-Middle

Potential Prevention?

- A sends public key to B
- B sends public key to A
- A encrypts message with B's public key, send's **half** of the message
- B encrypts message with A's public key, send's **half** of the message
- A sends other half, B can now decrypt entire message
- B sends other half, A can now decrypt entire message

Under which circumstances does this protocol prevent MiM-attacks, or does it?

Car Transponder (HiTag2)

1. C generates a random number N
2. C calculates $(F, G) = \{N\}_K$
3. $C \rightarrow T$: N, F
4. T calculates $(F', G') = \{N\}_K$
5. T checks that $F = F'$
6. $T \rightarrow C$: N, G'
7. C checks that $G = G'$

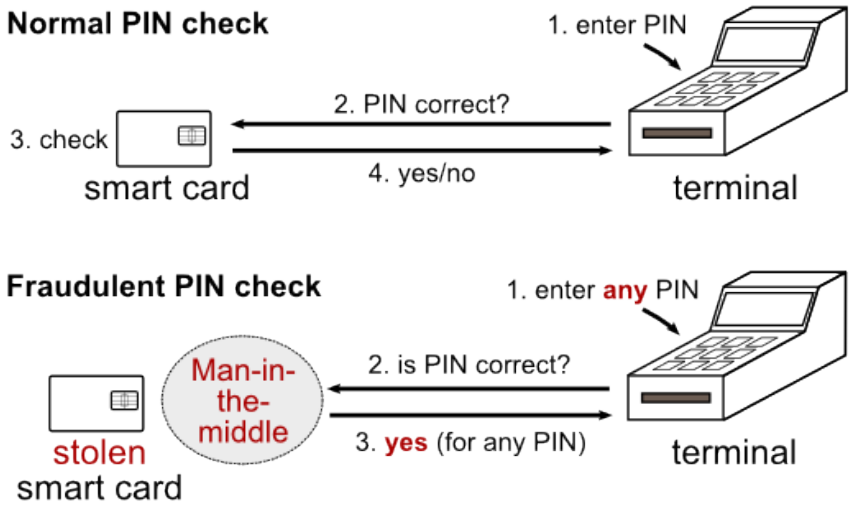
Car Transponder (HiTag2)

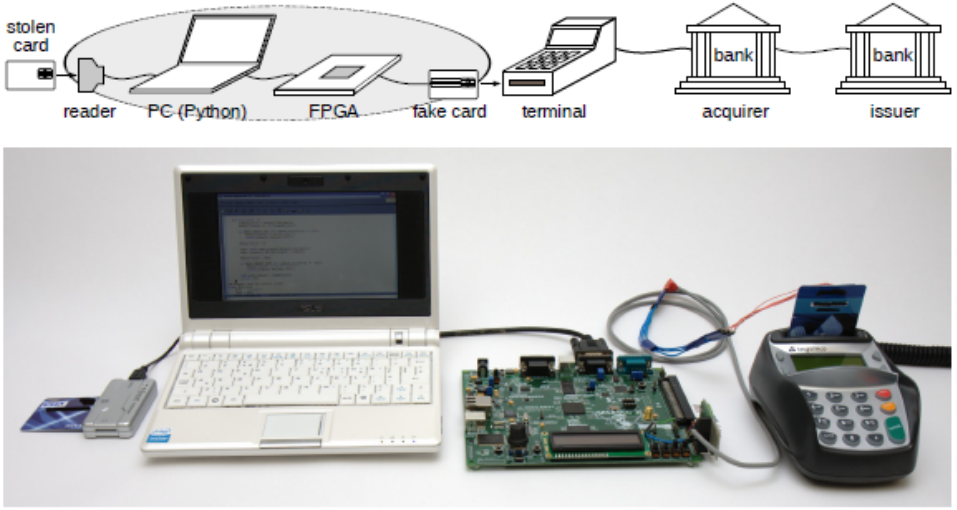
1. C generates a random number N
2. C calculates $(F, G) = \{N\}_K$
3. $C \rightarrow T$: N, F
4. T calculates $(F', G') = \{N\}_K$
5. T checks that $F = F'$
6. $T \rightarrow C$: N, G'
7. C checks that $G = G'$

This process means that the transponder believes the car knows the key K, and the car believes the transponder knows the key K. They have authenticated themselves to each other, or have they?

A Man-in-the-middle attack in real life:

- the card only says yes to the terminal if the PIN is correct
- trick the card in thinking transaction is verified by signature
- trick the terminal in thinking the transaction was verified by PIN





Problems with EMV

- it is a wrapper for many protocols
- specification by consensus (resulted unmanageable complexity)
- its specification is 700 pages in English plus 2000+ pages for testing, additionally some further parts are secret
- other attacks have been found

Protocols are Difficult

- even the systems designed by experts regularly fail
- the one who can fix a system should also be liable for the losses
- cryptography is often not the problem

A Simple PK Protocol

1. $A \rightarrow B : K_A^{pub}$
2. $B \rightarrow A : K_B^{pub}$
3. $A \rightarrow B : \{A, m\}_{K_B^{pub}}$
4. $B \rightarrow A : \{B, m'\}_{K_A^{pub}}$

A Simple PK Protocol

1. $A \rightarrow B : K_A^{pub}$
2. $B \rightarrow A : K_B^{pub}$
3. $A \rightarrow B : \{A, m\}_{K_B^{pub}}$
4. $B \rightarrow A : \{B, m'\}_{K_A^{pub}}$

unfortunately there is a simple man-in-the-middle-attack

A MITM Attack

1. $A \rightarrow E : K_A^{pub}$
2. $E \rightarrow B : K_E^{pub}$
3. $B \rightarrow E : K_B^{pub}$
4. $E \rightarrow A : K_E^{pub}$
5. $A \rightarrow E : \{A, m\}_{K_E^{pub}}$
6. $E \rightarrow B : \{E, m\}_{K_B^{pub}}$
7. $B \rightarrow E : \{B, m'\}_{K_E^{pub}}$
8. $E \rightarrow A : \{E, m'\}_{K_A^{pub}}$

A MITM Attack

1. $A \rightarrow E : K_A^{pub}$
2. $E \rightarrow B : K_E^{pub}$
3. $B \rightarrow E : K_B^{pub}$
4. $E \rightarrow A : K_E^{pub}$
5. $A \rightarrow E : \{A, m\}_{K_E^{pub}}$
6. $E \rightarrow B : \{E, m\}_{K_B^{pub}}$
7. $B \rightarrow E : \{B, m'\}_{K_E^{pub}}$
8. $E \rightarrow A : \{E, m'\}_{K_A^{pub}}$

and A and B have no chance to detect it

Interlock Protocol

The interlock protocol ("best bet" against MITM):

1. $A \rightarrow B : K_A^{pub}$
2. $B \rightarrow A : K_B^{pub}$
3. $\{A, m\}_{K_B^{pub}} \mapsto H_1, H_2$
   $\{B, m'\}_{K_A^{pub}} \mapsto M_1, M_2$
4. $A \rightarrow B : H_1$
5. $B \rightarrow A : \{H_1, M_1\}_{K_A^{pub}}$
6. $A \rightarrow B : \{H_2, M_1\}_{K_B^{pub}}$
7. $B \rightarrow A : M_2$

Splitting Messages

$$\underbrace{\texttt{0X1peUVTGJK+H70mMjAM8p}}_{\{A,m\}_{K_B^{pub}}}$$

$$\underbrace{\texttt{0X1peUVTGJK}}_{H_1} \quad \underbrace{\texttt{+H70mMjAM8p}}_{H_2}$$

- you can also use the even and odd bytes
- the point is you cannot decrypt the halves, even if you have the key

$$A \rightarrow C : K_A^{pub}$$
$$C \rightarrow B : K_C^{pub}$$
$$B \rightarrow C : K_B^{pub}$$
$$C \rightarrow A : K_C^{pub}$$

$$\{A, m\}_{K_C^{pub}} \mapsto H_1, H_2$$
$$\{B, m'\}_{K_C^{pub}} \mapsto M_1, M_2$$

$$\{C, a\}_{K_B^{pub}} \mapsto C_1, C_2$$
$$\{C, b\}_{K_A^{pub}} \mapsto D_1, D_2$$

$$A \rightarrow C : H_1$$
$$C \rightarrow B : C_1$$
$$B \rightarrow C : \{C_1, M_1\}_{K_C^{pub}}$$
$$C \rightarrow A : \{H_1, D_1\}_{K_A^{pub}}$$
$$A \rightarrow C : \{H_2, D_1\}_{K_C^{pub}}$$
$$C \rightarrow B : \{C_2, M_1\}_{K_B^{pub}}$$
$$B \rightarrow C : M_2$$
$$C \rightarrow A : D_2$$

$$A \rightarrow C : K_A^{pub}$$
$$C \rightarrow B : K_C^{pub}$$
$$B \rightarrow C : K_B^{pub}$$
$$C \rightarrow A : K_C^{pub}$$

$$\{A, m\}_{K_C^{pub}} \mapsto H_1, H_2$$
$$\{B, m'\}_{K_C^{pub}} \mapsto M_1, M_2$$

$$\{C, a\}_{K_B^{pub}} \mapsto C_1, C_2$$
$$\{C, b\}_{K_A^{pub}} \mapsto D_1, D_2$$

$$A \rightarrow C : H_1$$
$$C \rightarrow B : C_1$$
$$B \rightarrow C : \{C_1, M_1\}_{K_C^{pub}}$$
$$C \rightarrow A : \{H_1, D_1\}_{K_A^{pub}}$$
$$A \rightarrow C : \{H_2, D_1\}_{K_C^{pub}}$$
$$C \rightarrow B : \{C_2, M_1\}_{K_B^{pub}}$$
$$B \rightarrow C : M_2$$
$$C \rightarrow A : D_2$$

m = How is your grandmother? m' = How is the weather today in London?

- you have to ask something that cannot be imitated (requires A and B know each other)
- what happens if m and m' are voice messages?

- you have to ask something that cannot be imitated (requires A and B know each other)
- what happens if m and m' are voice messages?

- So C can either leave the communication unchanged, or invent a complete new conversation

- the moral: establishing a secure connection from "zero" is almost impossible—you need to rely on some established trust
- that is why PKI relies on certificates, which however are badly, badly realised

Trusted Third Parties

Simple protocol for establishing a secure connection via a mutually trusted 3rd party (server):

$$
\begin{aligned}
&A \rightarrow S : A, B \\
&S \rightarrow A : \{K_{AB}, \{K_{AB}\}_{K_{BS}}\}_{K_{AS}} \\
&A \rightarrow B : \{K_{AB}\}_{K_{BS}} \\
&A \rightarrow B : \{m\}_{K_{AB}}
\end{aligned}
$$

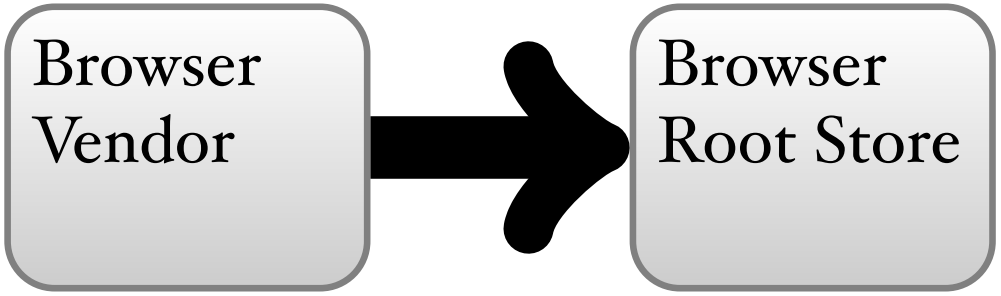
PKI: The Main Idea

- the idea is to have a certificate authority (CA)
- you go to the CA to identify yourself
- CA: "I, the CA, have verified that public key P^{pub}_{Bob} belongs to Bob"

- CA must be trusted by everybody

- certificates are time limited, and can be revoked
- What happens if CA issues a false certificate? Who pays in case of loss? (VeriSign explicitly limits liability to $100.)

PKI: Chains of Trust



- CAs make almost no money anymore, because of stiff competition
- browser companies are not really interested in security; only in market share

PKI: Weaknesses

CAs just cannot win (make any profit):

- there are hundreds of CAs, which issue millions of certificates and the error rate is small
- users (servers) do not want to pay or pay as little as possible

- a CA can issue a certificate for any domain not needing any permission (CAs are meant to undergo audits, but...DigiNotar)
- if a CA has issued many certificates, it "becomes too big to fail"
- Can we be sure CAs are not just frontends of some government organisation?

PKI: Weaknesses

- many certificates are issued via Whois, whether you own the domain...if you hijacked a domain, it is easy to obtain certificates
- the revocation mechanism does not work (Chrome has given up on general revocation lists)
- lax approach to validation of certificates (Have you ever bypassed certification warnings?)
- sometimes you want to actually install invalid certificates (self-signed)

PKI: Attacks

- Go directly after root certificates
  - governments can demand private keys
  - 10 years ago it was estimated that breaking a 1024 bit key takes one year and costs 10 - 30 Mio $; this is now reduced to 1 Mio $
- Go after buggy implementations of certificate validation
- Social Engineering
  - in 2001 somebody pretended to be from Microsoft and asked for two code-signing certificates

The eco-system is completely broken (it relies on thousands of entities to do the right thing). Maybe DNSSEC where keys can be attached to domain names is a way out.

Real Attacks

- In 2011, DigiNotar (Dutch company) was the first CA that got compromised comprehensively, and where many fraudulent certificates were issued to the wild. It included approximately 300,000 IP addresses, mostly located in Iran. The attackers (in Iran?) were likely interested "only" in collecting gmail passwords.
- The Flame malware piggy-bagged on this attack by advertising malicious Windows updates to some targeted systems (mostly in Iran, Israel, Sudan).

PKI is Broken

- PKI and certificates are meant to protect you against MITM attacks, but if the attack occurs your are presented with a warning and you need to decide whether you are under attack.
- Webcontent gets often loaded from 3rd-party servers, which might not be secured
- Misaligned incentives: browser vendors are not interested in breaking webpages with invalid certificates

Why are there so many invalid certificates?

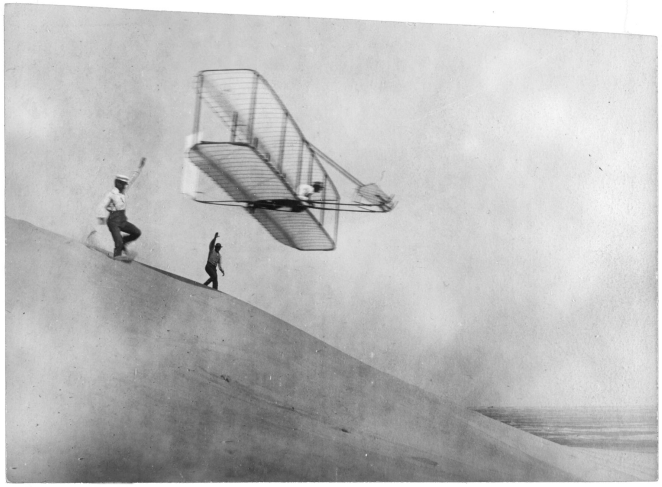
- insufficient name coverage (www.example.com should include example.com)
- IoT: many appliances have web-based admin interfaces; the manufacturer cannot know under which IP and domain name the appliances are run (so cannot install a valid certificate)
- expired certificates, or incomplete chains of trust (servers are supposed to supply them)

Mid-Term

- homework, handouts, programs...

Any Questions?

Security Engineering



Wright brothers, 1901

Airbus, 2005

1st Lecture

- chip-and-pin, banks vs. customers

  the one who can improve security should also be liable for the losses

1st Lecture

- chip-and-pin, banks vs. customers

  the one who can improve security should also be liable for the losses

- hashes and salts to guarantee data integrity
- storing passwords (you should know the difference between brute force attacks and dictionary attacks; how do salts help?)

1st Lecture: Cookies

- good uses of cookies?
- bad uses of cookies: snooping, tracking, profiling...the "disadvantage" is that the user is in control, because you can delete them

"Please track me using cookies."

1st Lecture: Cookies

- good uses of cookies?
- bad uses of cookies: snooping, tracking, profiling...the "disadvantage" is that the user is in control, because you can delete them

  "Please track me using cookies."

- fingerprinting beyond browser cookies

  Pixel Perfect: Fingerprinting Canvas in HTML5
  (a research paper from 2012)
  http://cseweb.ucsd.edu/~hovav/papers/ms12.html

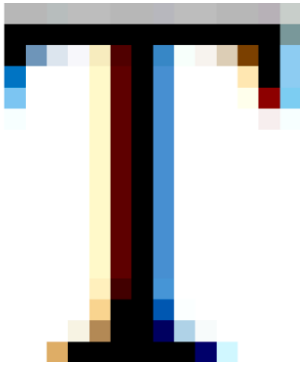
1st Lecture: Cookies

- a bit of JavaScript and HTML5 + canvas

Firefox

55b2257ad0f20ecbf927fb66a15c61981f7ed8fc

Safari



17bc79f8111e345f572a4f87d6cd780b445625d3

- no actual drawing needed

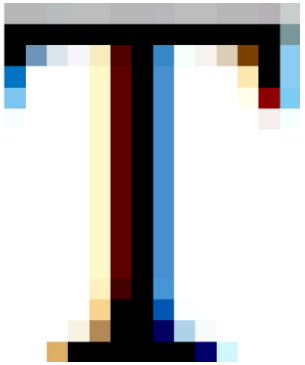
1st Lecture: Cookies

- a bit of JavaScript and HTML5 + canvas

Firefox

55b2257ad0f20ecbf927fb66a15c61981f7ed8fc

Safari



17bc79f8111e345f572a4f87d6cd780b445625d3

- no actual drawing needed
- in May 2014 a crawl of 100,000 popular webpages revealed 5.5% already use canvas fingerprinting

https://securehomes.esat.kuleuven.be/~gacar/persistent/the_web_never_forgets.pdf

1st Lecture: Cookies

Remember the small web-app I showed you where a cookie protected a counter?

- NYT, the cookie looks the “resource” - harm
- imaginary discount unlocked by cookie - no harm

2nd Lecture: E-Voting

Where are paper ballots better than voice voting?

- Integrity
- Ballot Secrecy
- Voter Authentication
- Enfranchisement
- Availability

2nd Lecture: E-Voting

- recently an Australian parliamentary committee found: e-voting is highly vulnerable to hacking and Australia will not use it any time soon

2nd Lecture: E-Voting

- recently an Australian parliamentary committee found: e-voting is highly vulnerable to hacking and Australia will not use it any time soon

- Alex Halderman, Washington D.C. hack

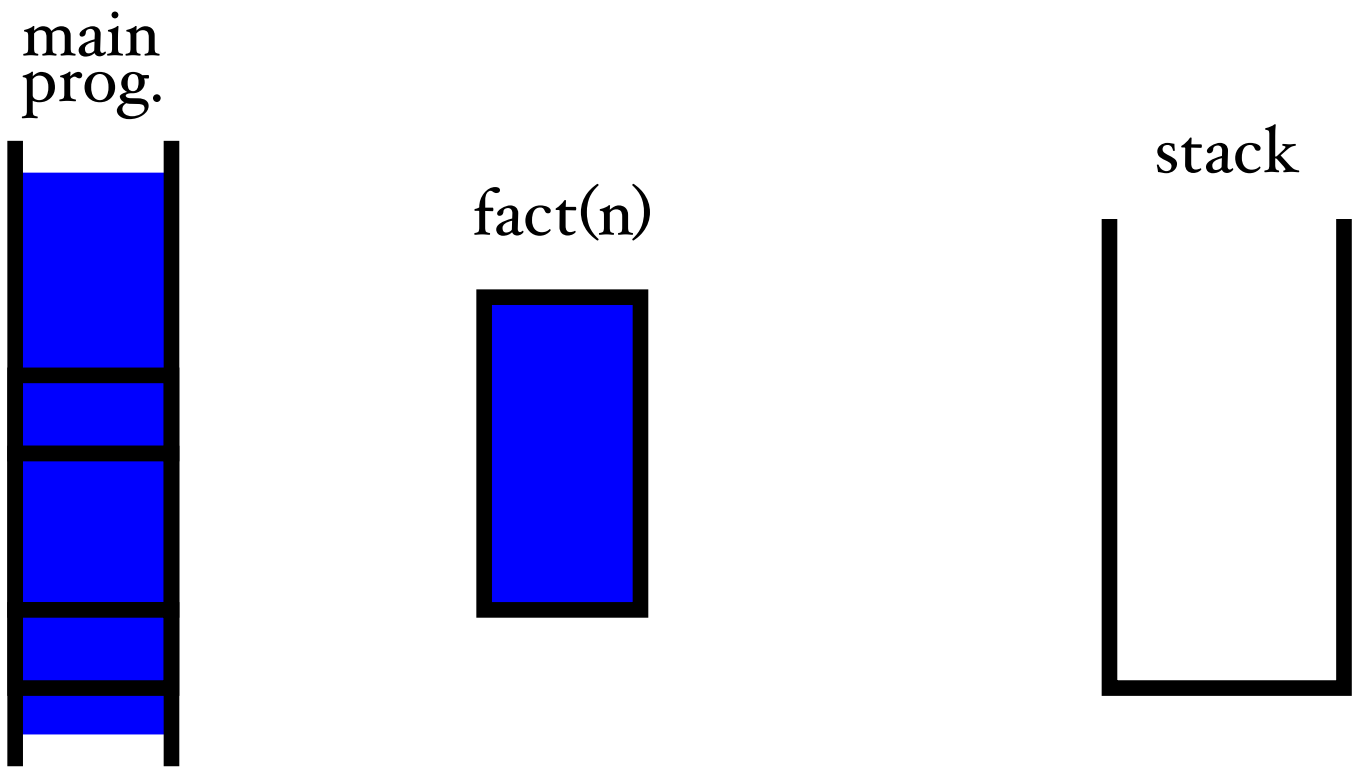
  https://jhalderm.com/pub/papers/dcvoting-fc12.pdf

- PDF-ballot tampering at the wireless router (the modification is nearly undetectable and leaves no traces; MITM attack with firmware updating)

  http://galois.com/wp-content/uploads/2014/11/technical-hack-a-pdf.pdf

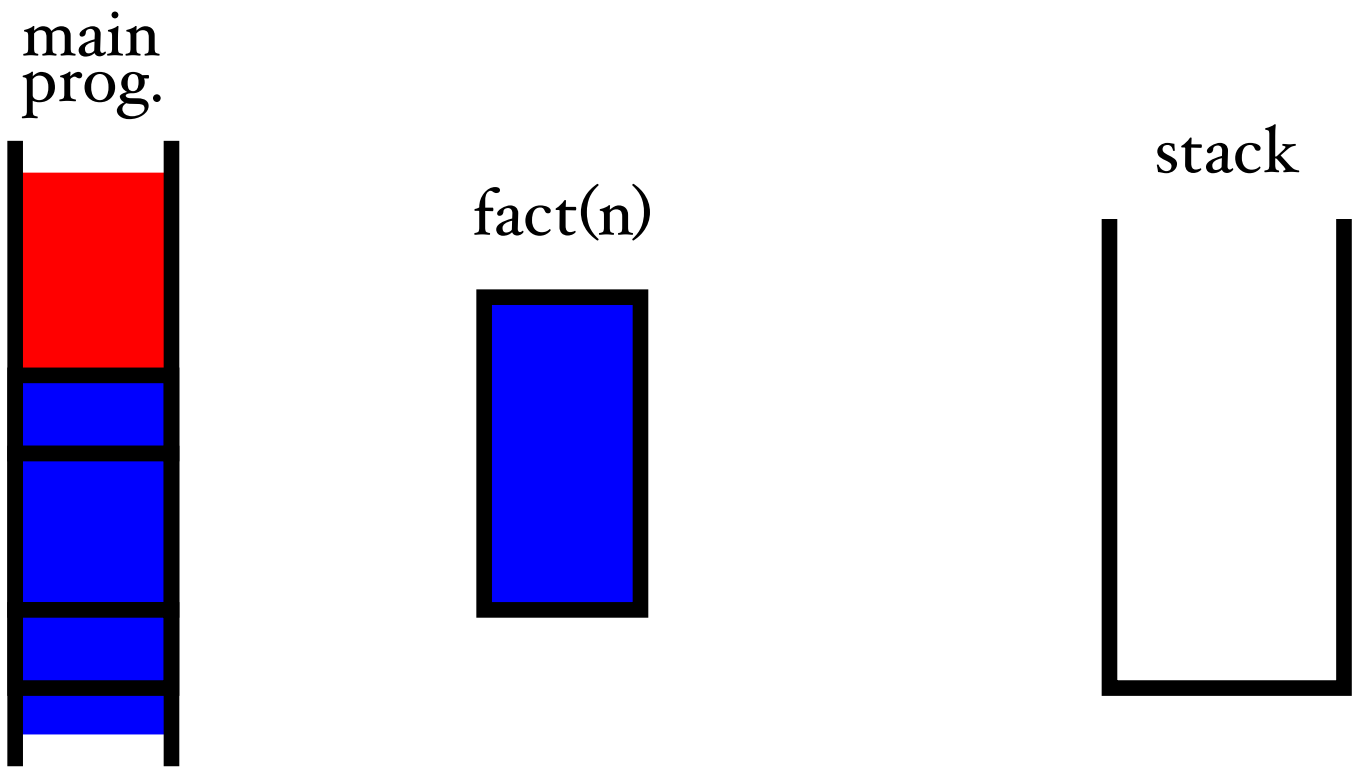
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



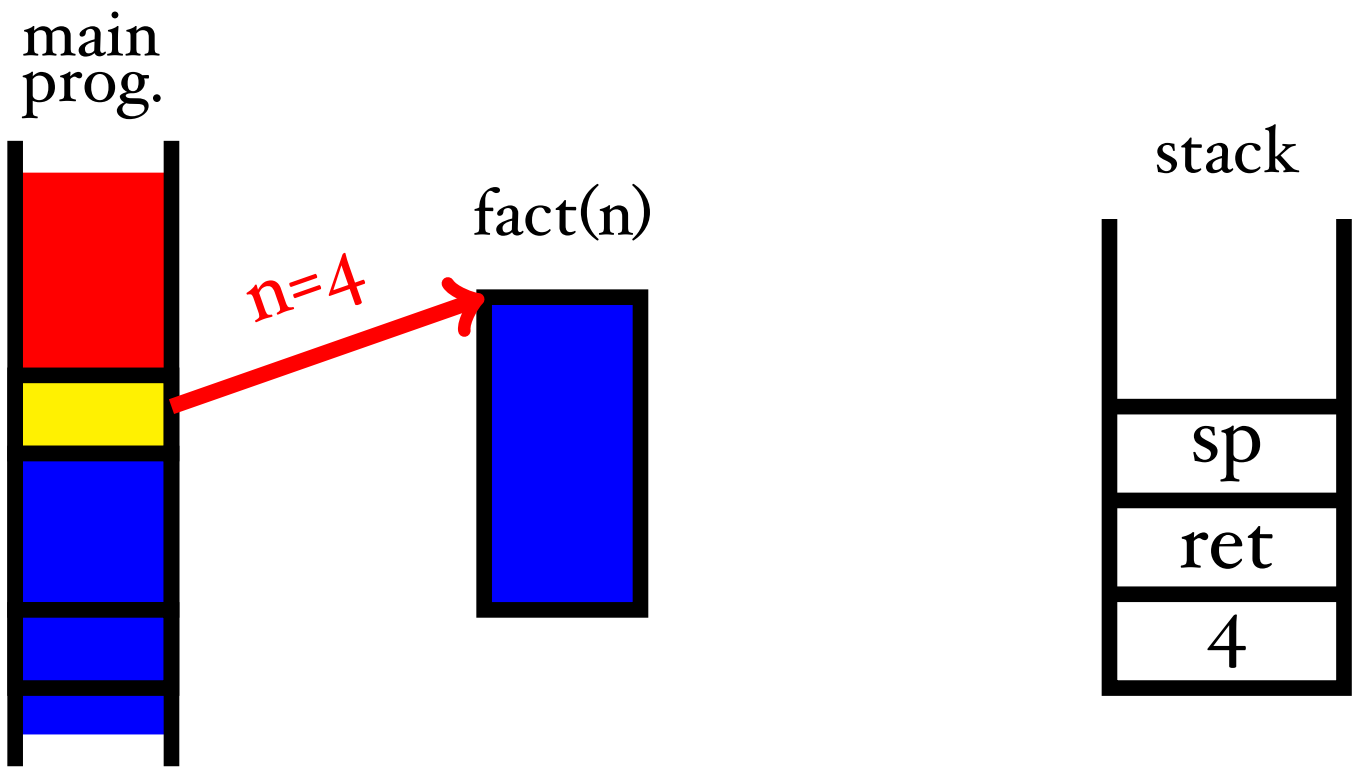
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



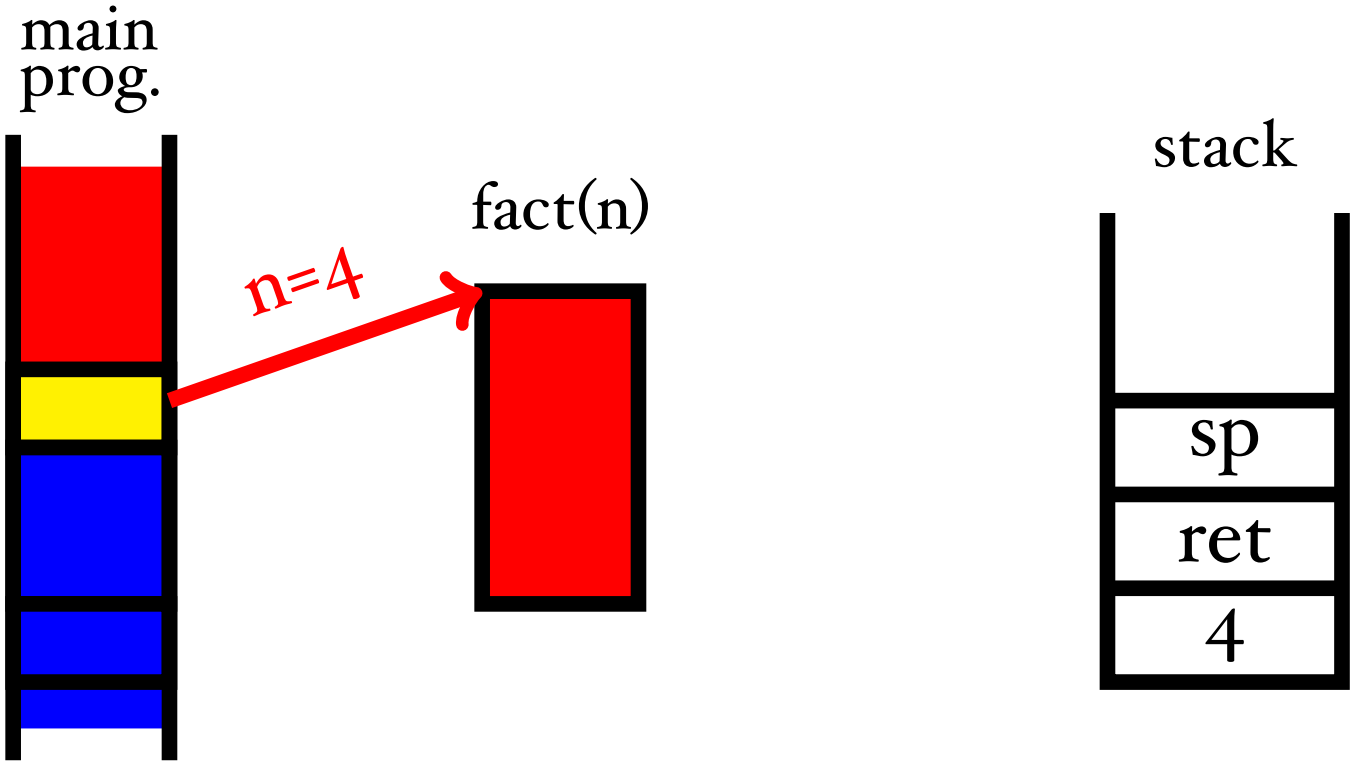
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



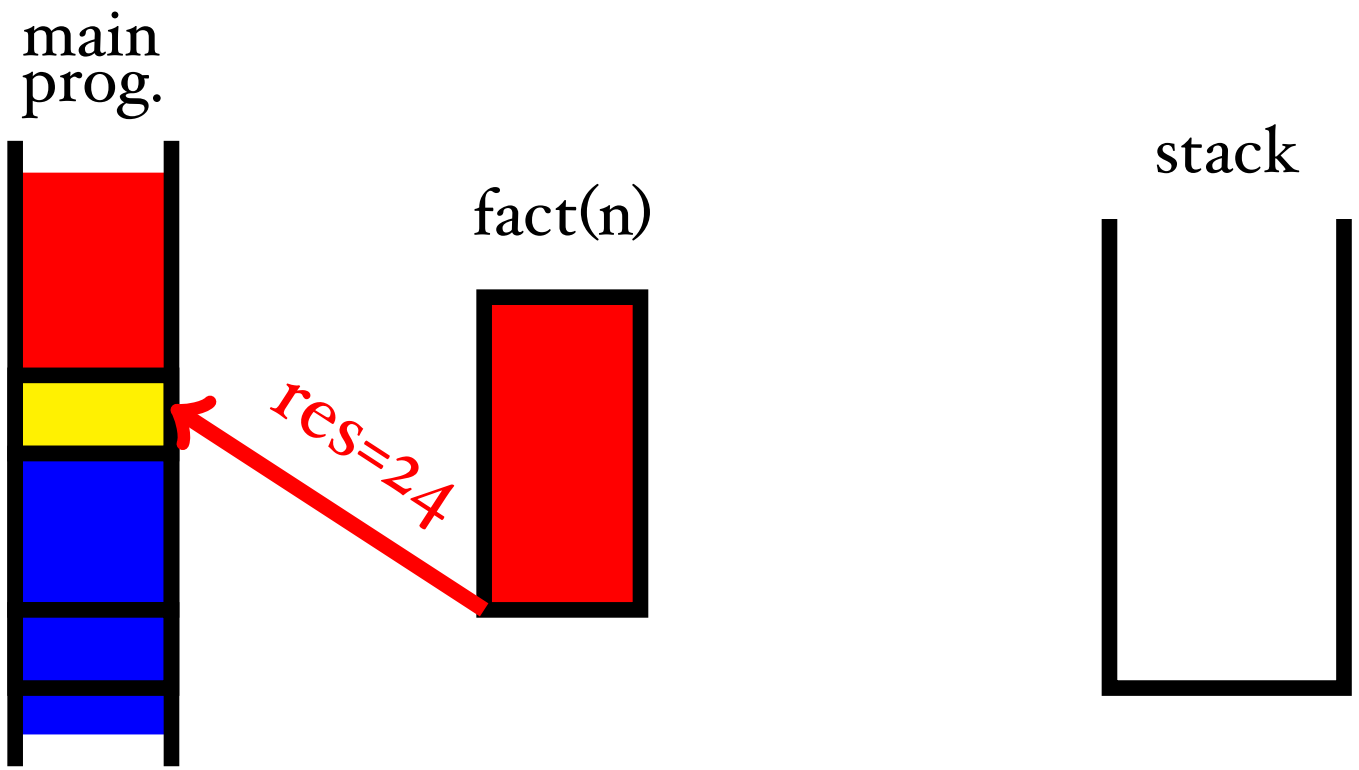
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



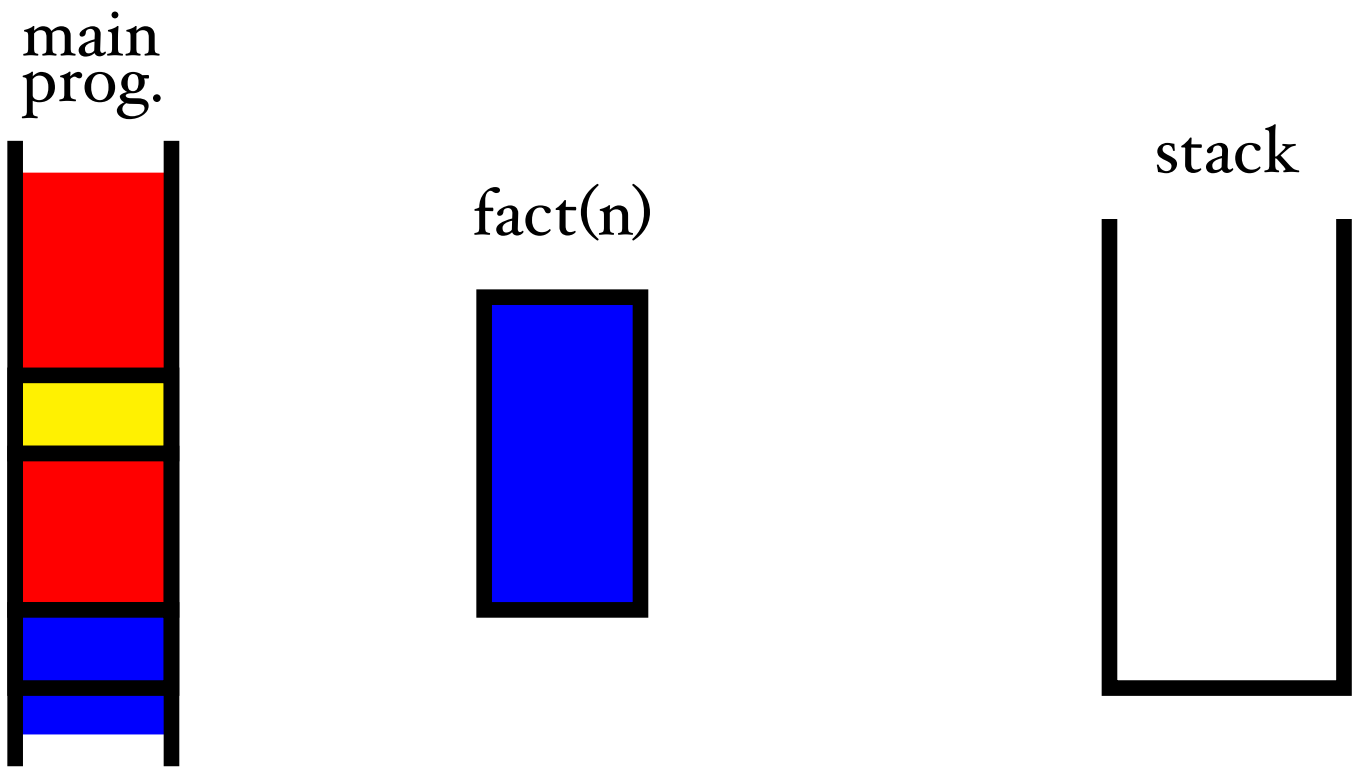
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



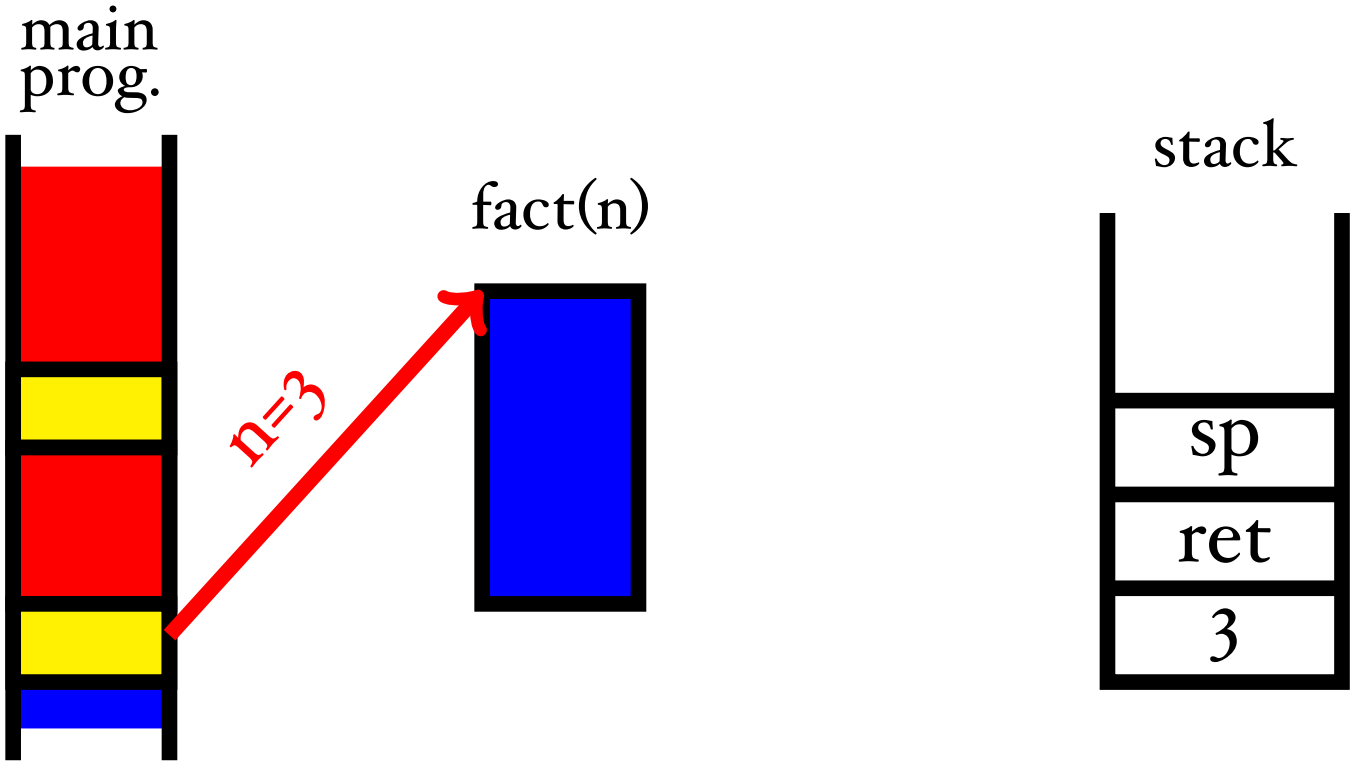
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



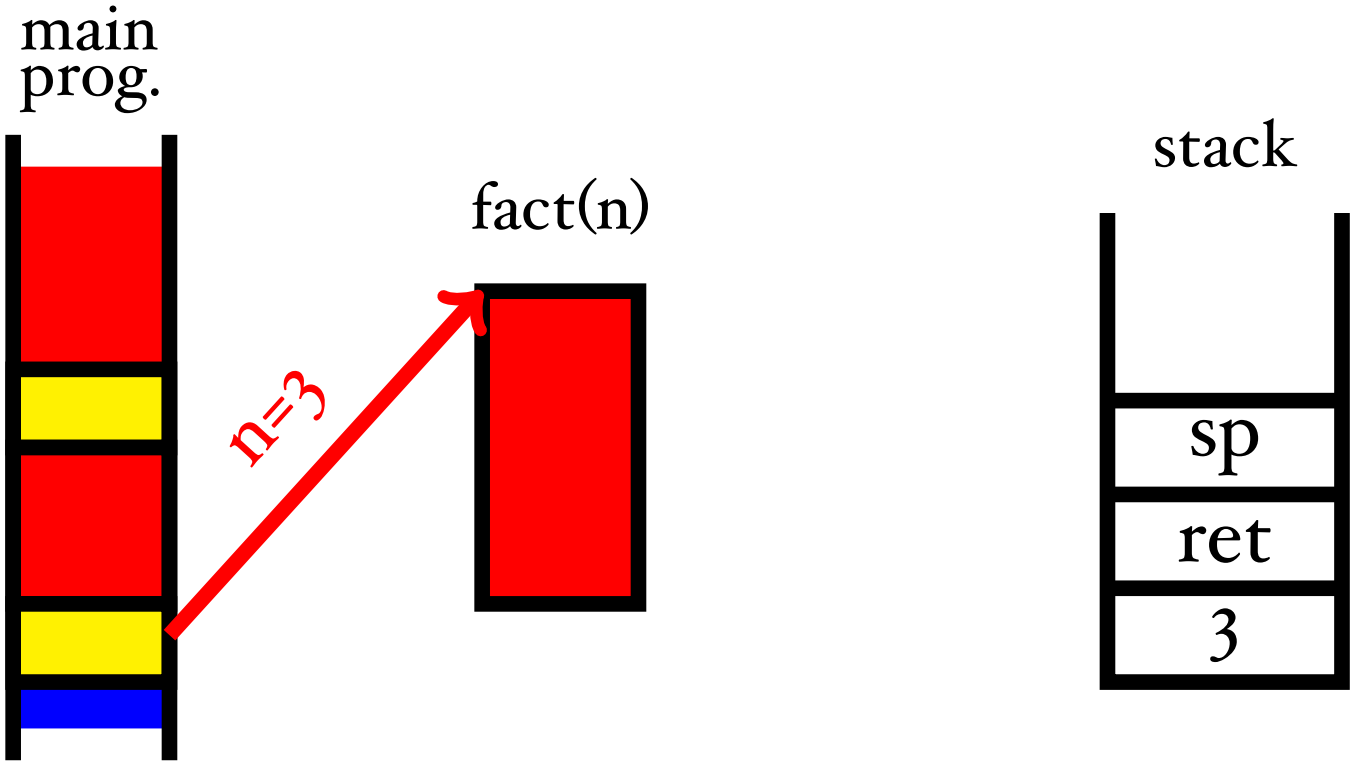
3rd Lecture: Buffer Overflow Attacks

- the problem arises from the way C/C++ organises its function calls



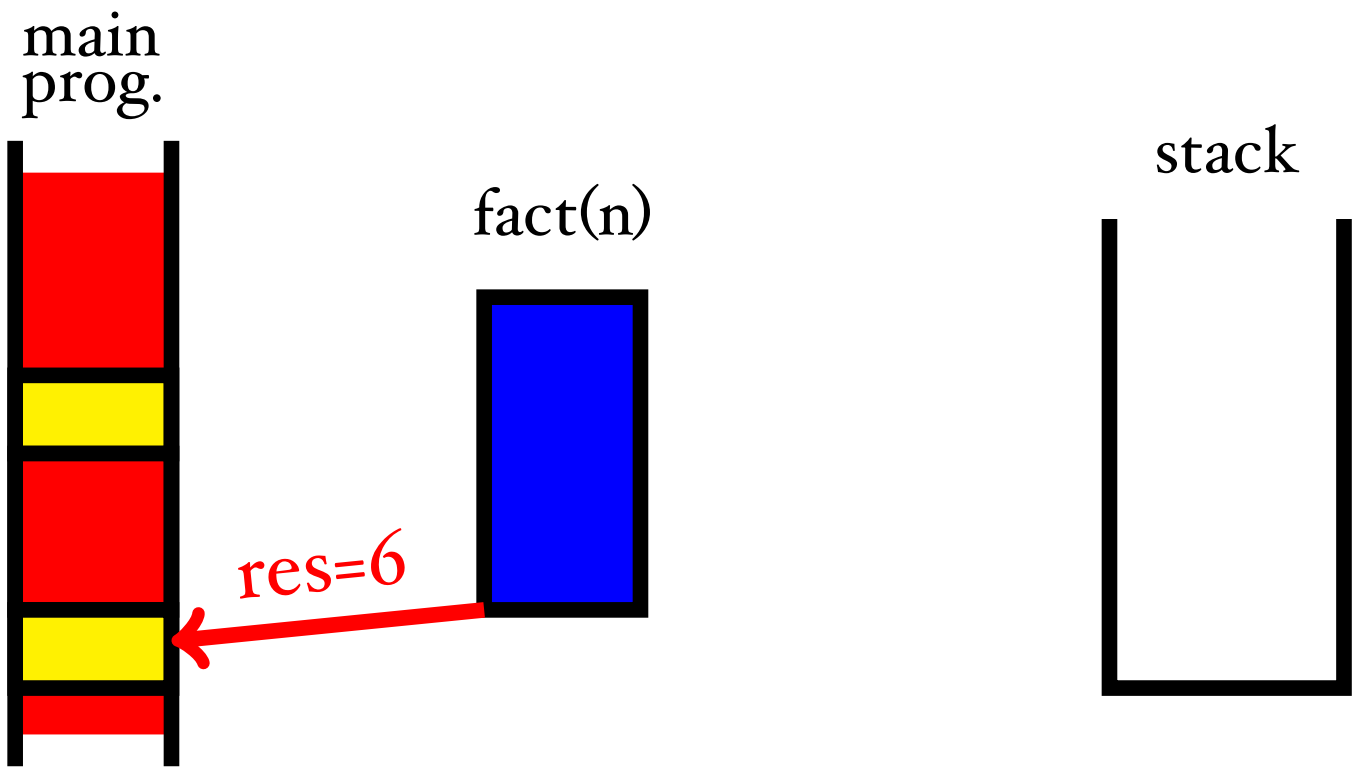
3rd Lecture: Buffer Overflow Attacks

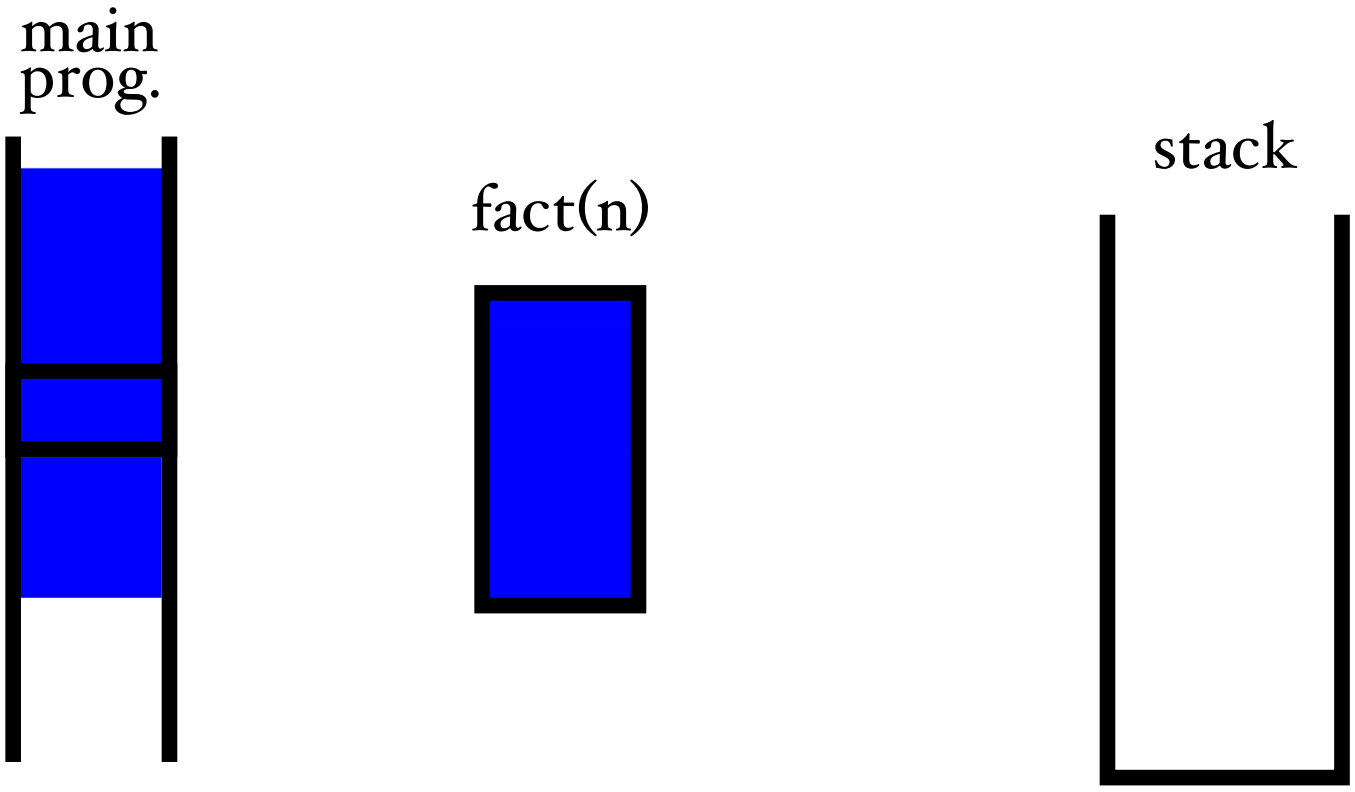
- the problem arises from the way C/C++ organises its function calls

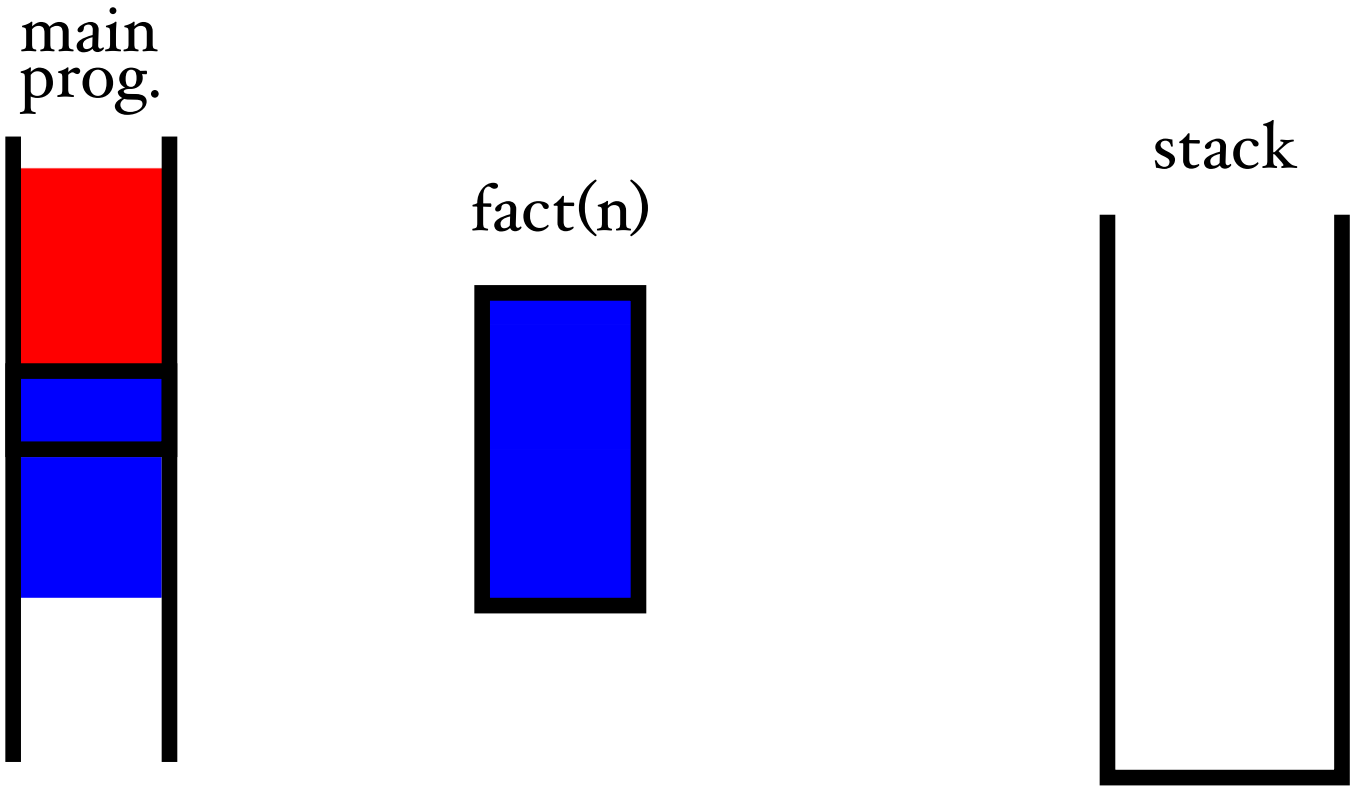


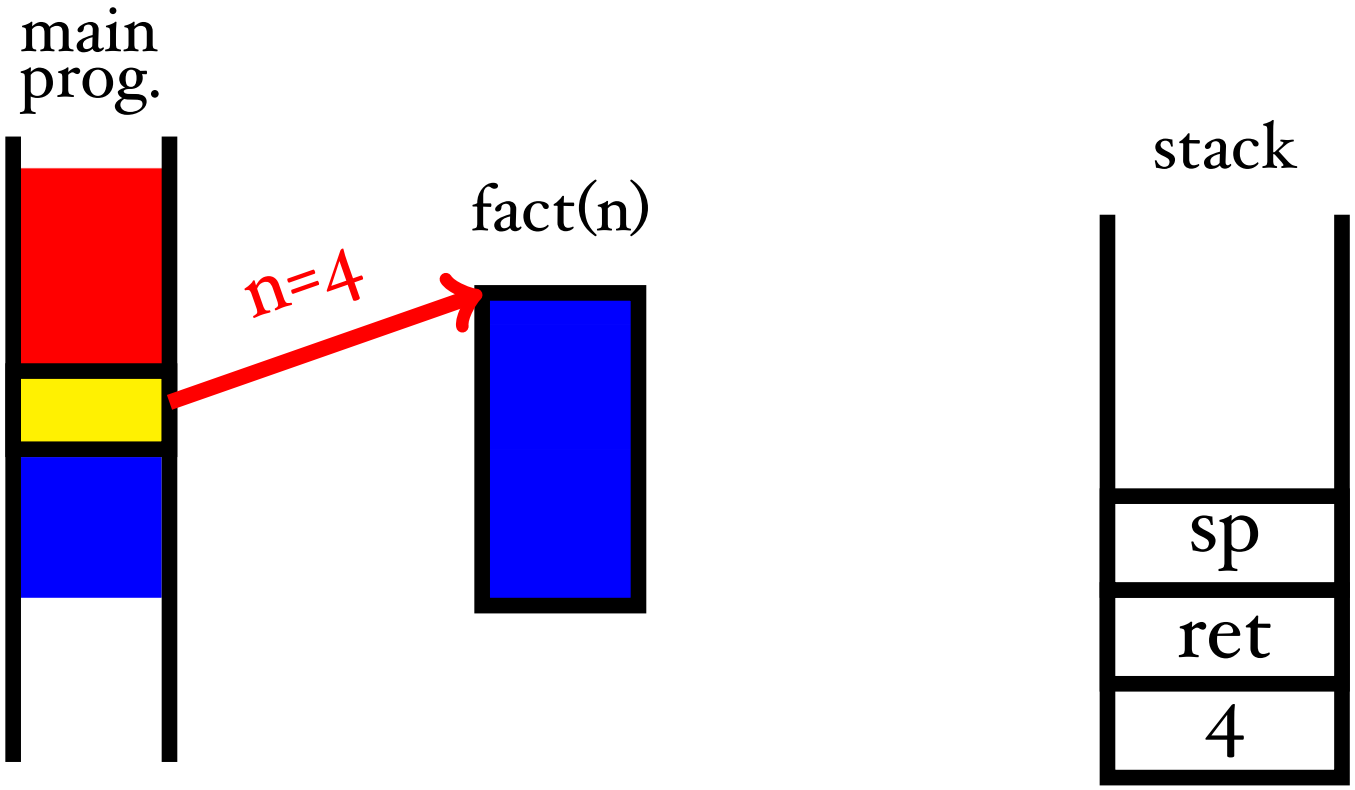
3rd Lecture: Buffer Overflow Attacks

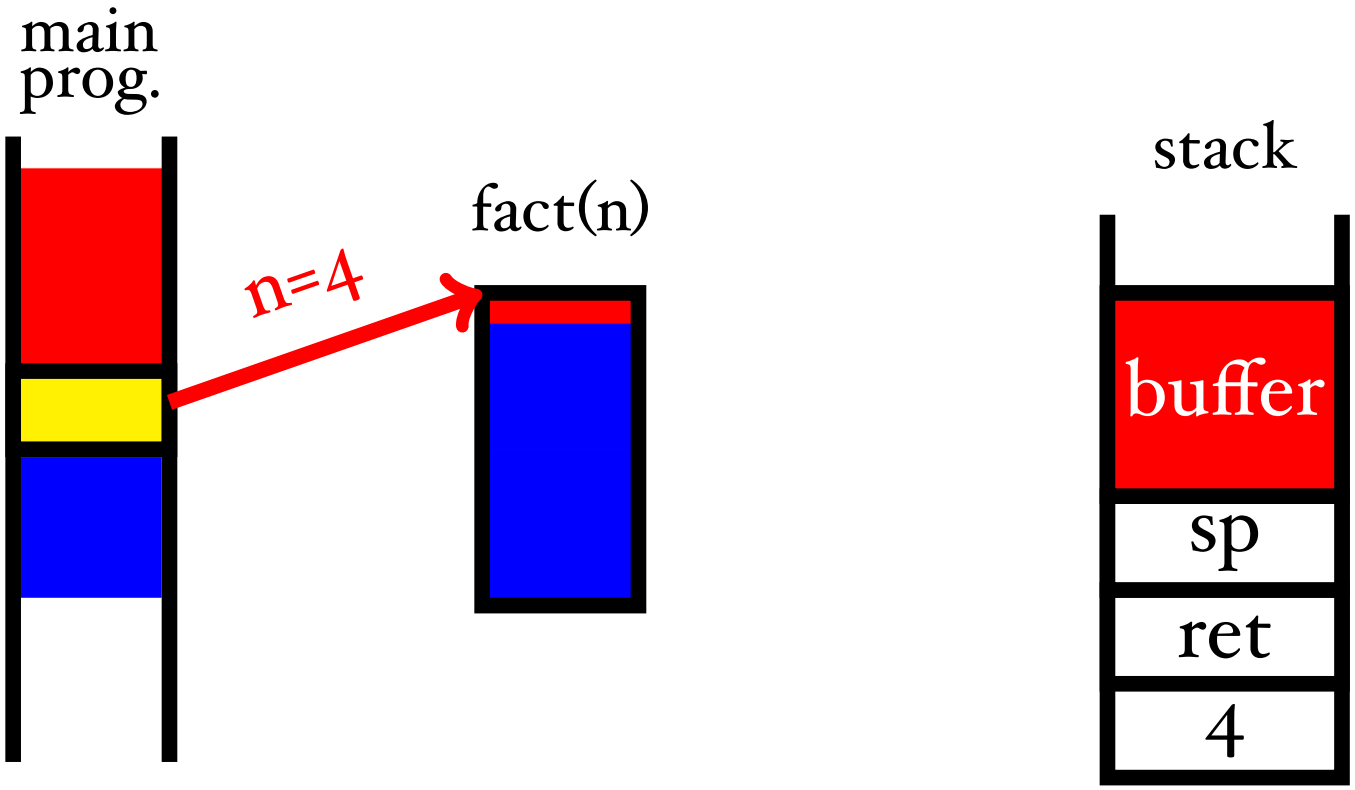
- the problem arises from the way C/C++ organises its function calls

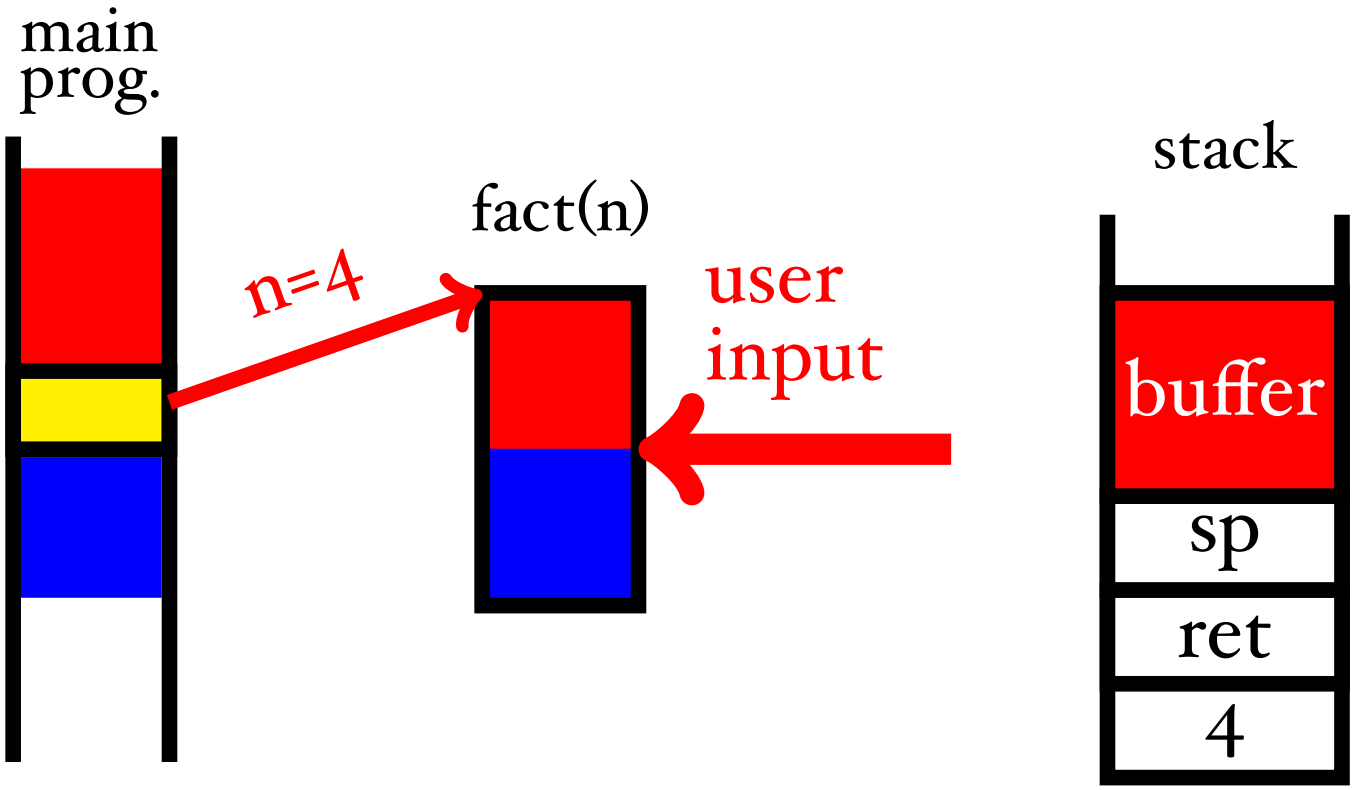


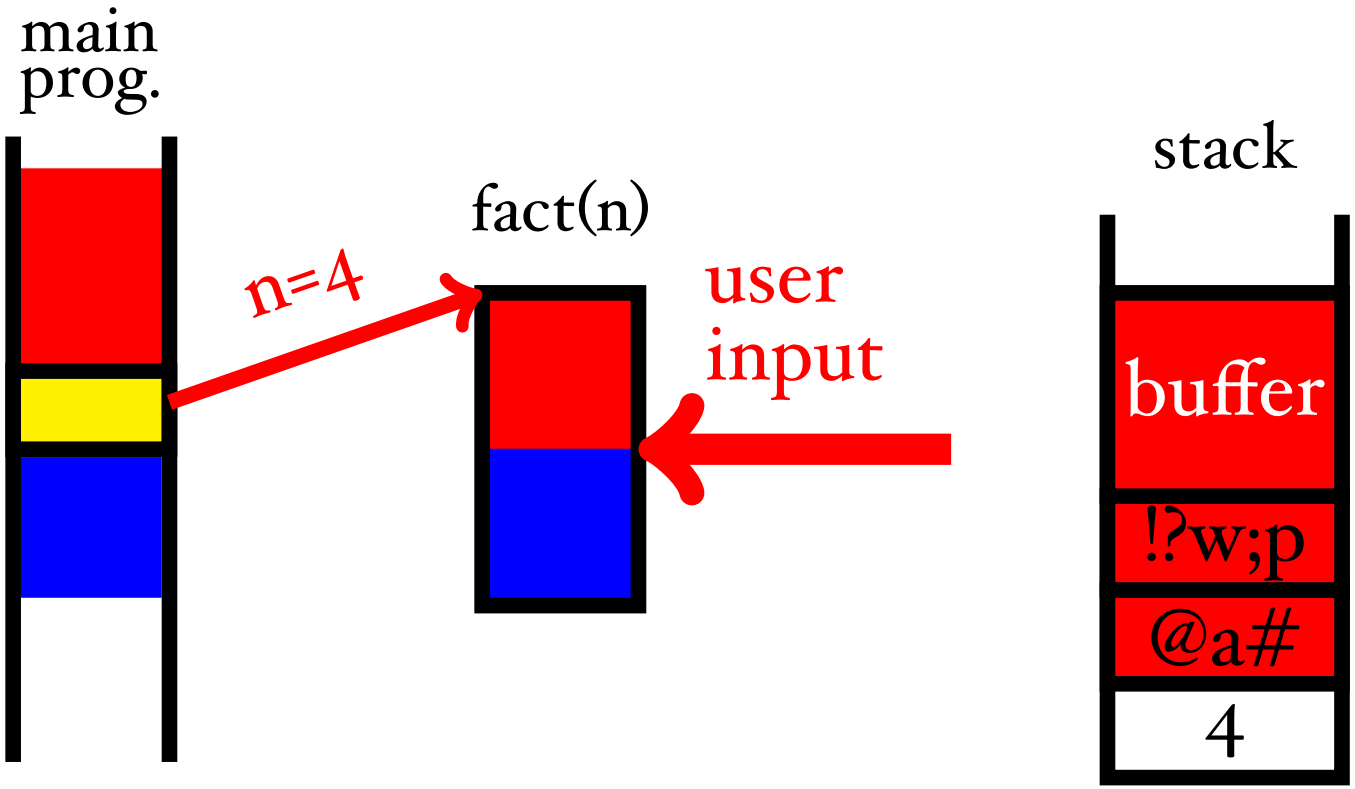

main
prog.
fact(n)
stack

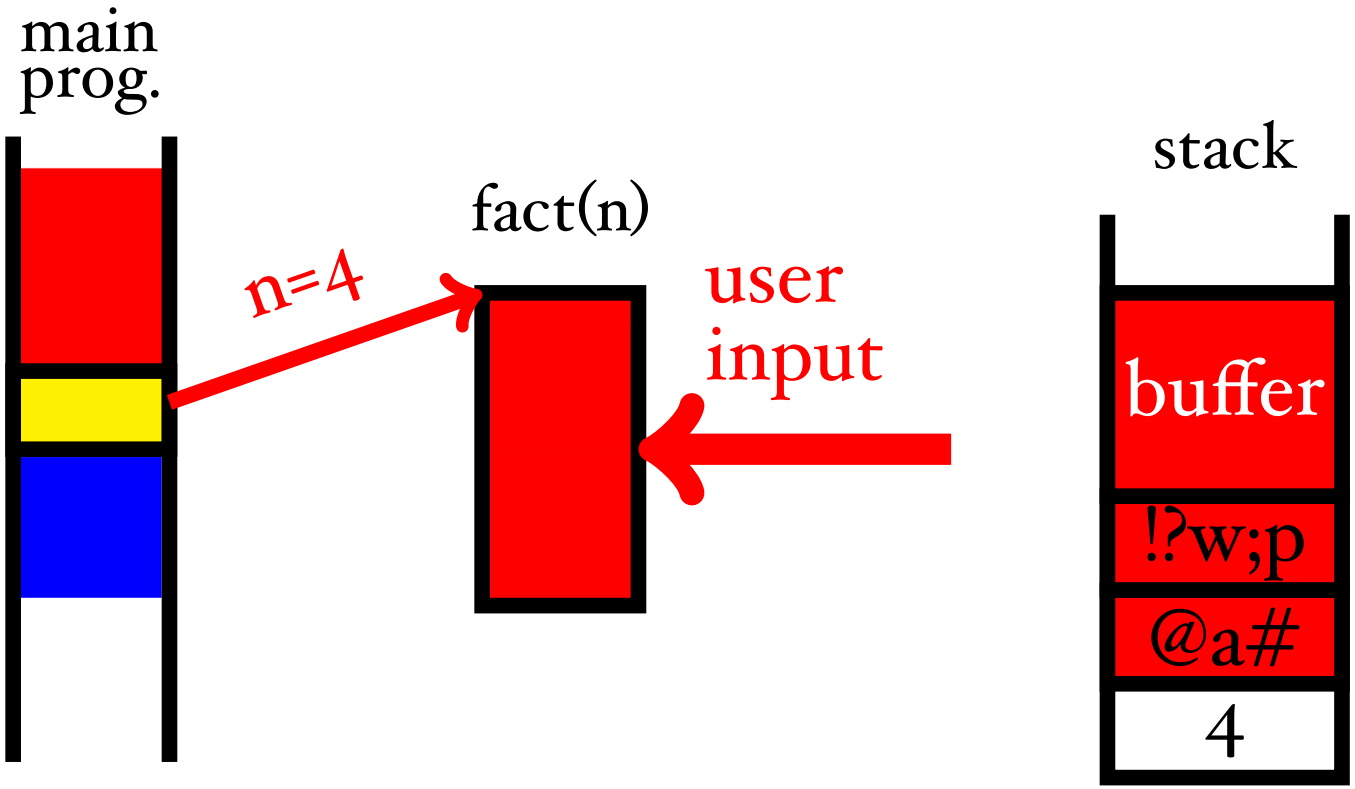

main
prog.
fact(n)
stack

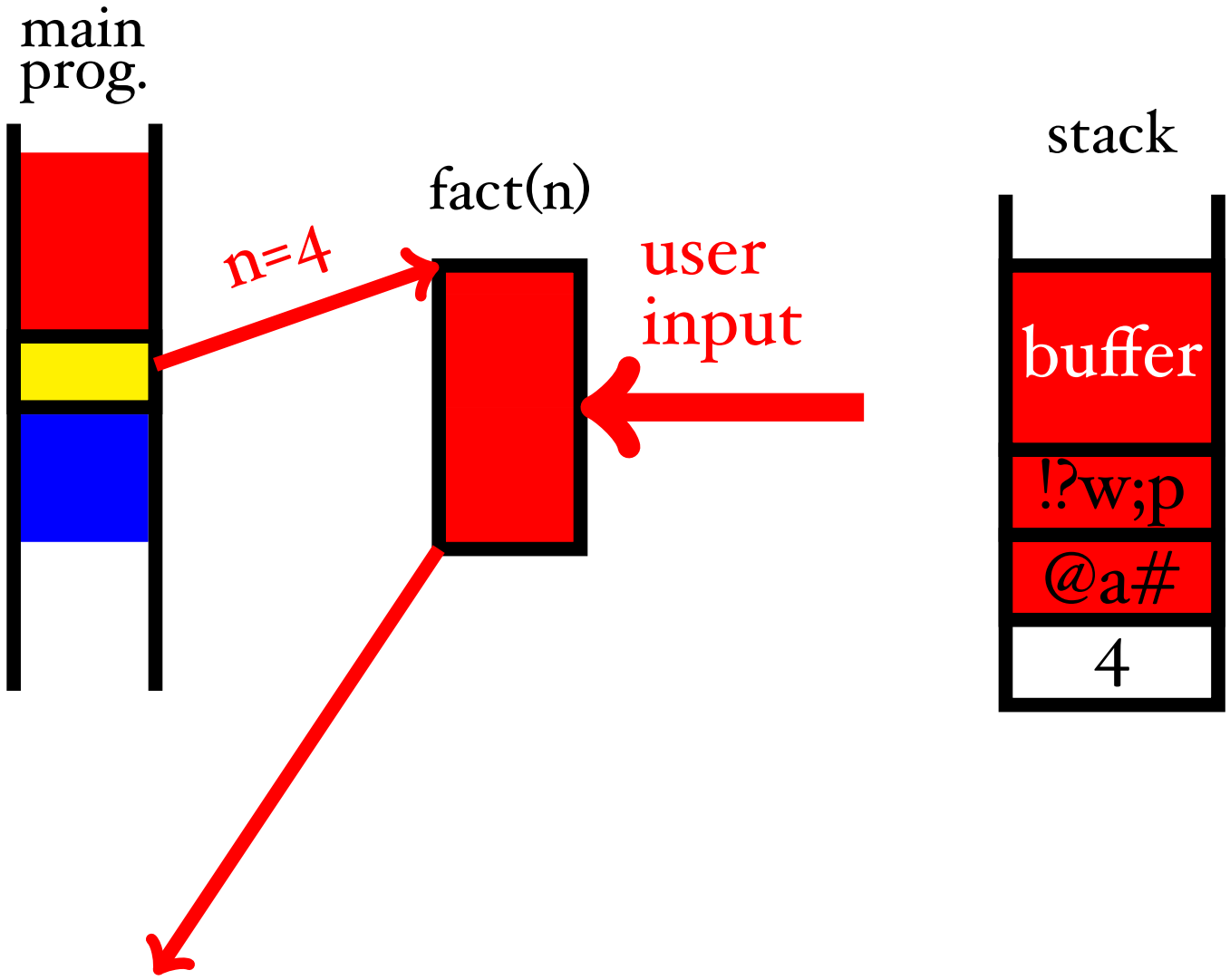

main
prog.
n=4
fact(n)
stack
sp
ret
4


main
prog.
n=4
fact(n)
stack
buffer
sp
ret
4


main
prog.
n=4
fact(n)
user
input
stack
buffer
sp
ret
4

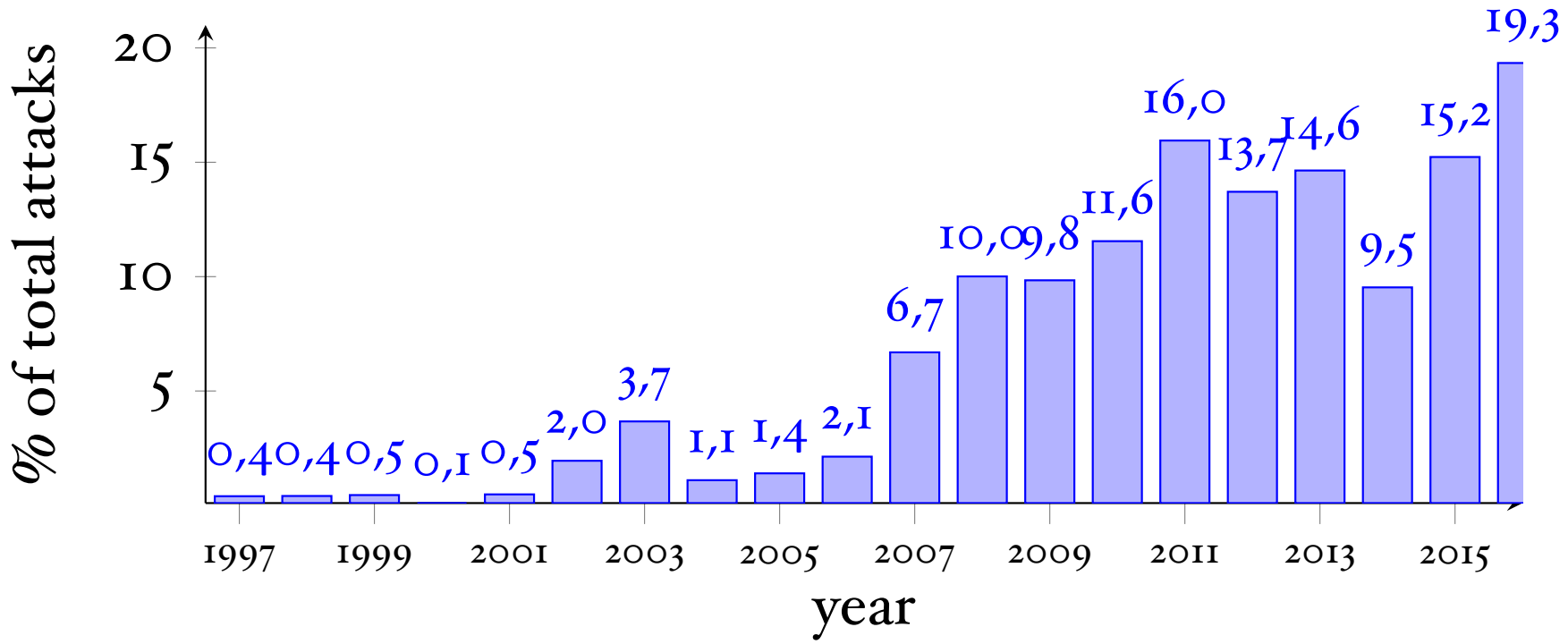

main
prog.
n=4
fact(n)
user
input
stack
buffer
!?w;p
@a#
4


main
prog.
n=4
fact(n)
user
input
stack
buffer
!?w;p
@a#
4


main
prog.
n=4
fact(n)
user
input
stack
buffer
!?w;p
@a#
4

3rd Lecture: Buffer Overflow Attacks

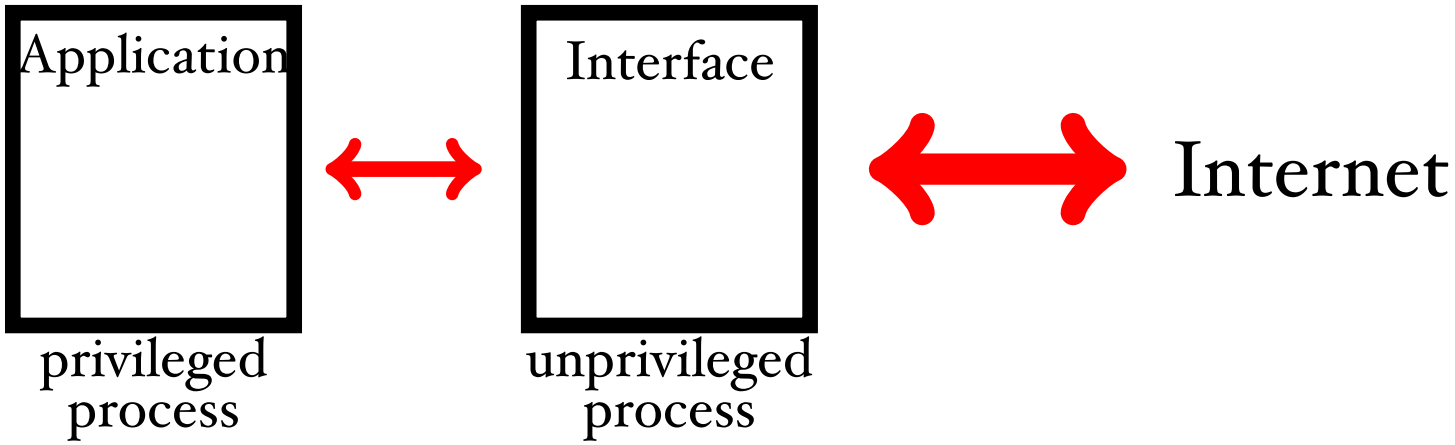
US National Vulnerability Database
(636 out of 6675 in 2014)



http://web.nvd.nist.gov/view/vuln/statistics

4th Lecture: Unix Access Control

- privileges are specified by file access permissions ("everything is a file")



- the idea is to make the attack surface smaller and mitigate the consequences of an attack

4th Lecture: Unix Access Control

- when a file with setuid is executed, the resulting process will assume the UID given to the owner of the file

```
$ ls -ld . * */*
drwxr-xr-x 1 ping staff  32768 Apr  2 2010 .
-rw----r-- 1 ping students  31359 Jul 24 2011 manual.txt
-r--rw--w- 1 bob students    4359 Jul 24 2011 report.txt
-rwsr--r-x 1 bob students  141359 Jun  1 2013 microedit
dr--r-xr-x 1 bob staff      32768 Jul 23 2011 src
-rw-r--r-- 1 bob staff      81359 Feb 28 2012 src/code.c
-r--rw---- 1 emma students    959 Jan 23 2012 src/code.h
```

4th Lecture: Unix Access Control

- Alice wants to have her files readable, except for her office mates.
- make sure you understand the setuid and setgid bits; why are they necessary for login and passwd