Redes de Computadores
2020/2021

Introdução ao Projecto:
“RC Two Factor Authentication”

IST LEIC-A

4 componentes:

- *PD*, *User*, *AS*, FS

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

4 componentes:

- *PD*, *User*, *AS*, FS

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

4 componentes:

- *PD*, *User*, *AS*, FS

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
   -n sigma03 -p 58000
>
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

4 componentes:

- *PD*, *User*, *AS*, FS

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
   -n sigma03 -p 58000
>
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
```

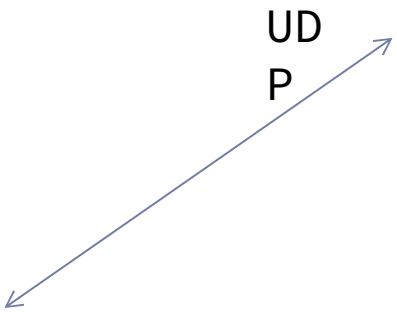
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
    -n sigma03 -p 58000
> reg 12345 password
```

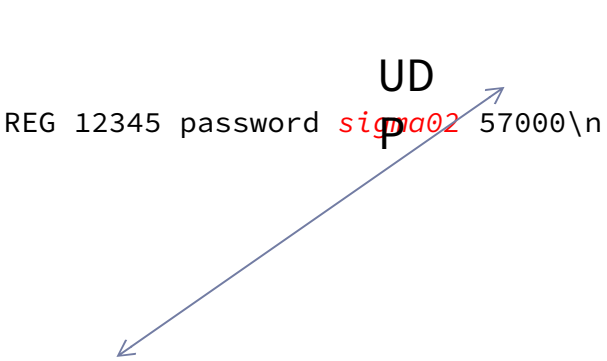
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
```

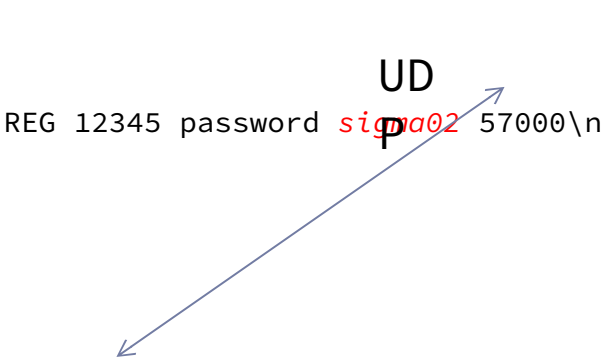
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
    -n sigma03 -p 58000
> reg 12345 password
```

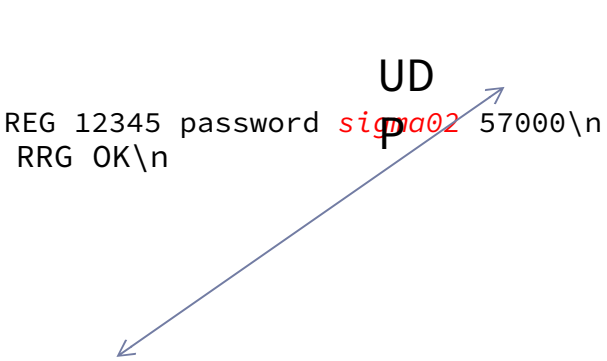
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
    -n sigma03 -p 58000
> reg 12345 password
```

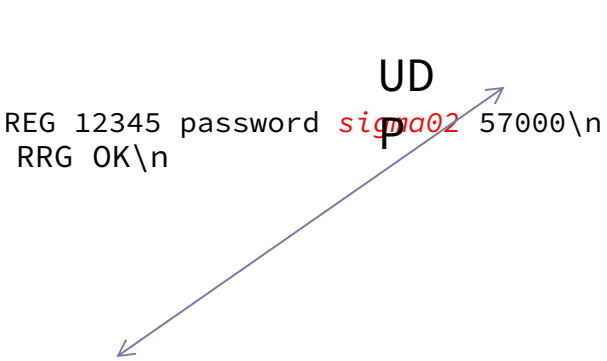
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
    -n sigma03 -p 58000
> reg 12345 password
  Registration
successful
>
```

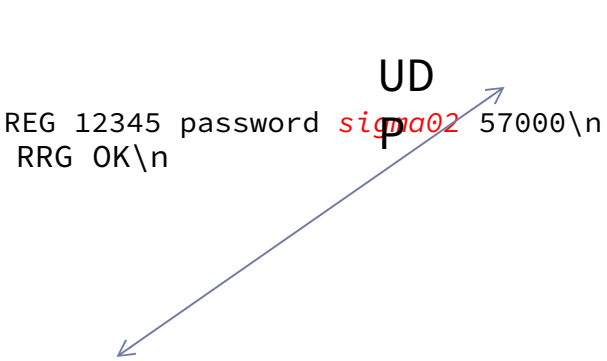
FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
```

PD: sigma02.ist.utl.pt

```
$/PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
  Registration
successful
>
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

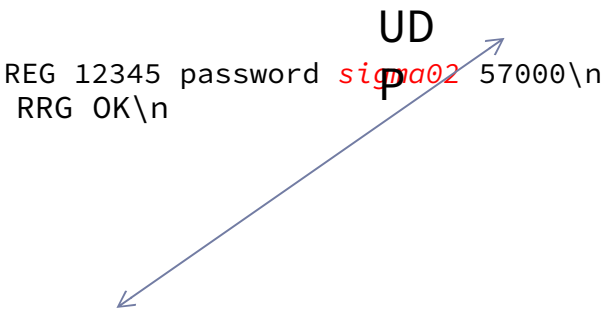
User: sigma01.ist.utl.pt

```
$
```

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
```

PD: sigma02.ist.utl.pt

```
$./PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
  Registration
successful
>
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

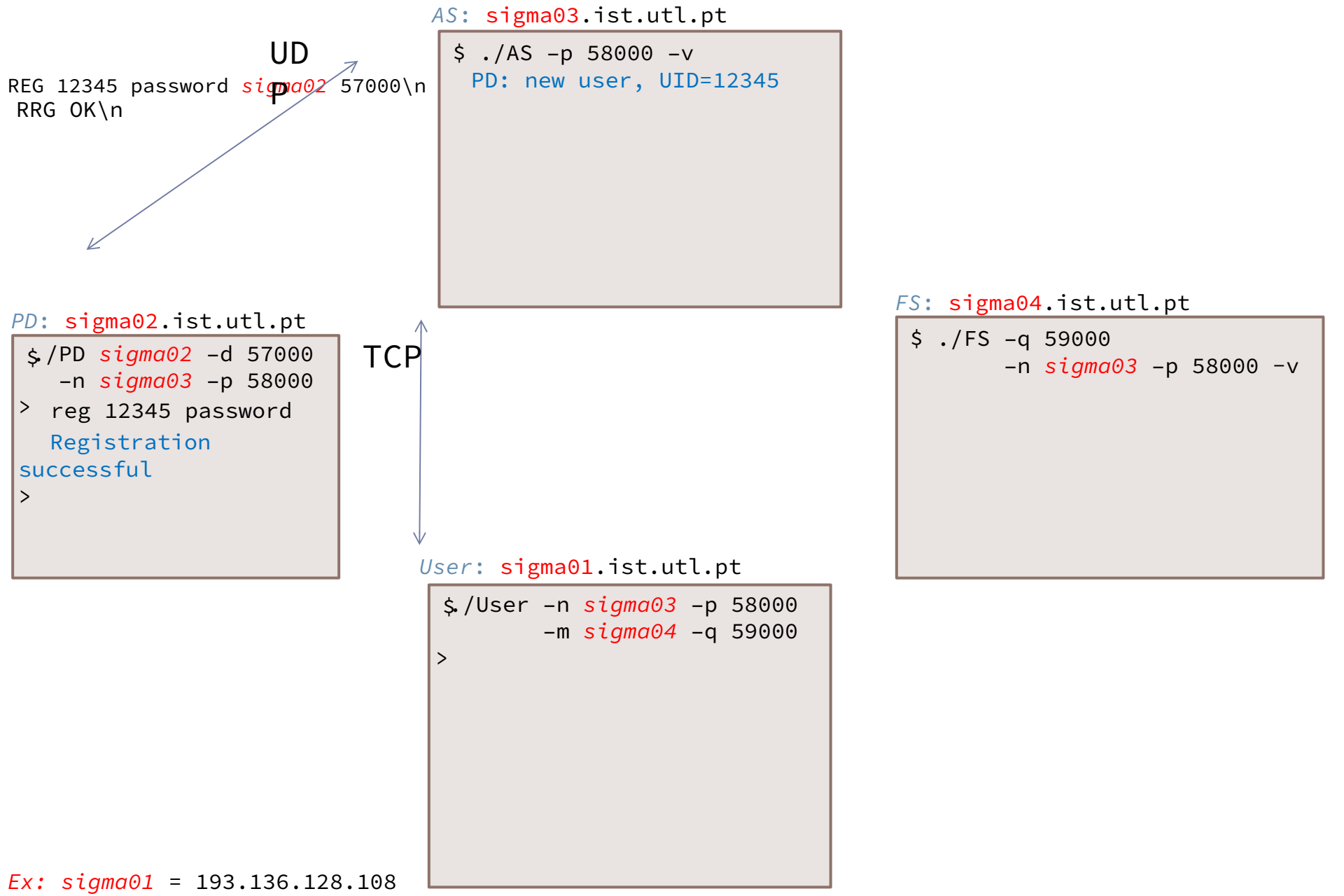
User: sigma01.ist.utl.pt

```
$./User -n sigma03 -p 58000
        -m sigma04 -q 59000
>
```

Ex: sigma01 = 193.136.128.108

4 componentes:

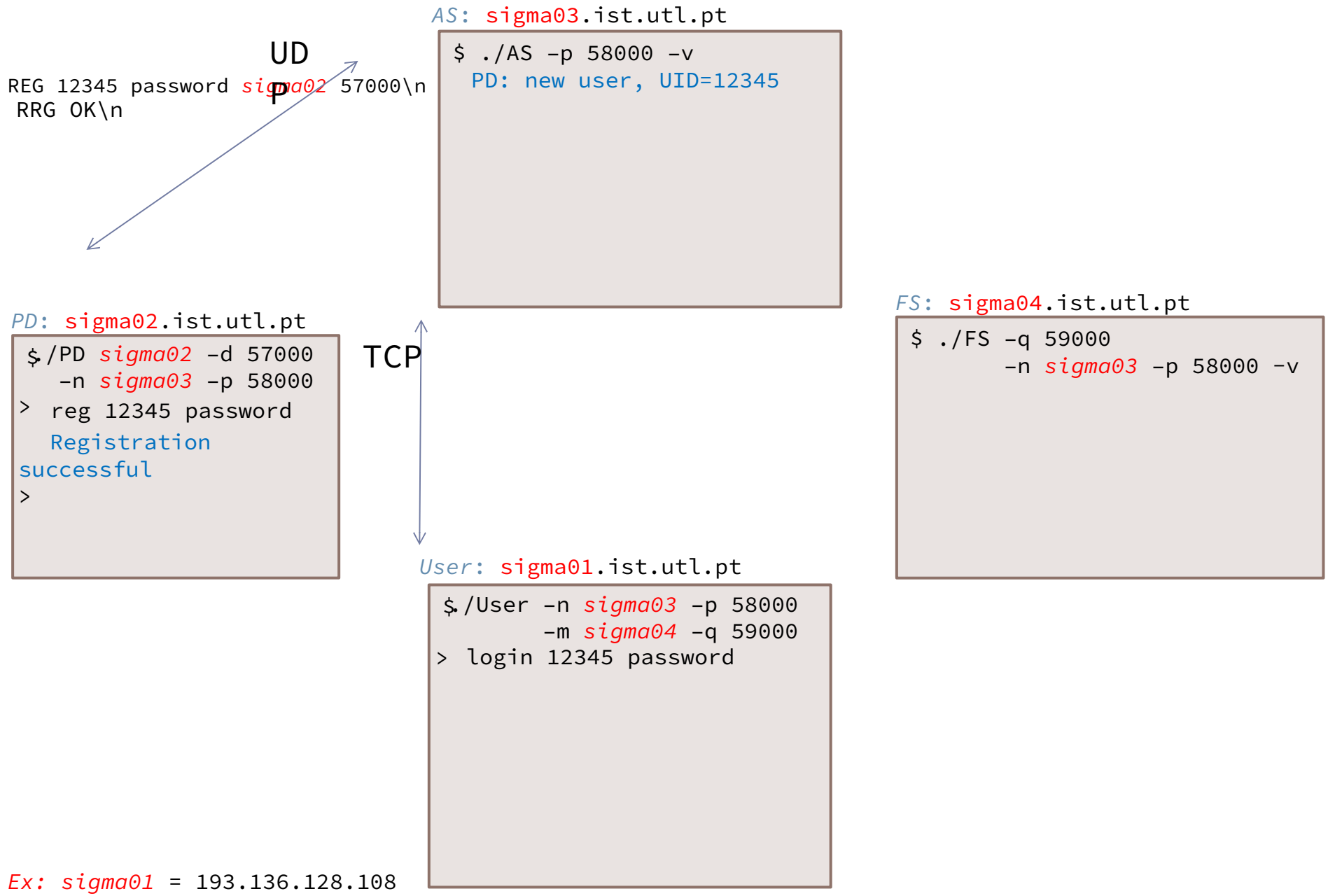
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

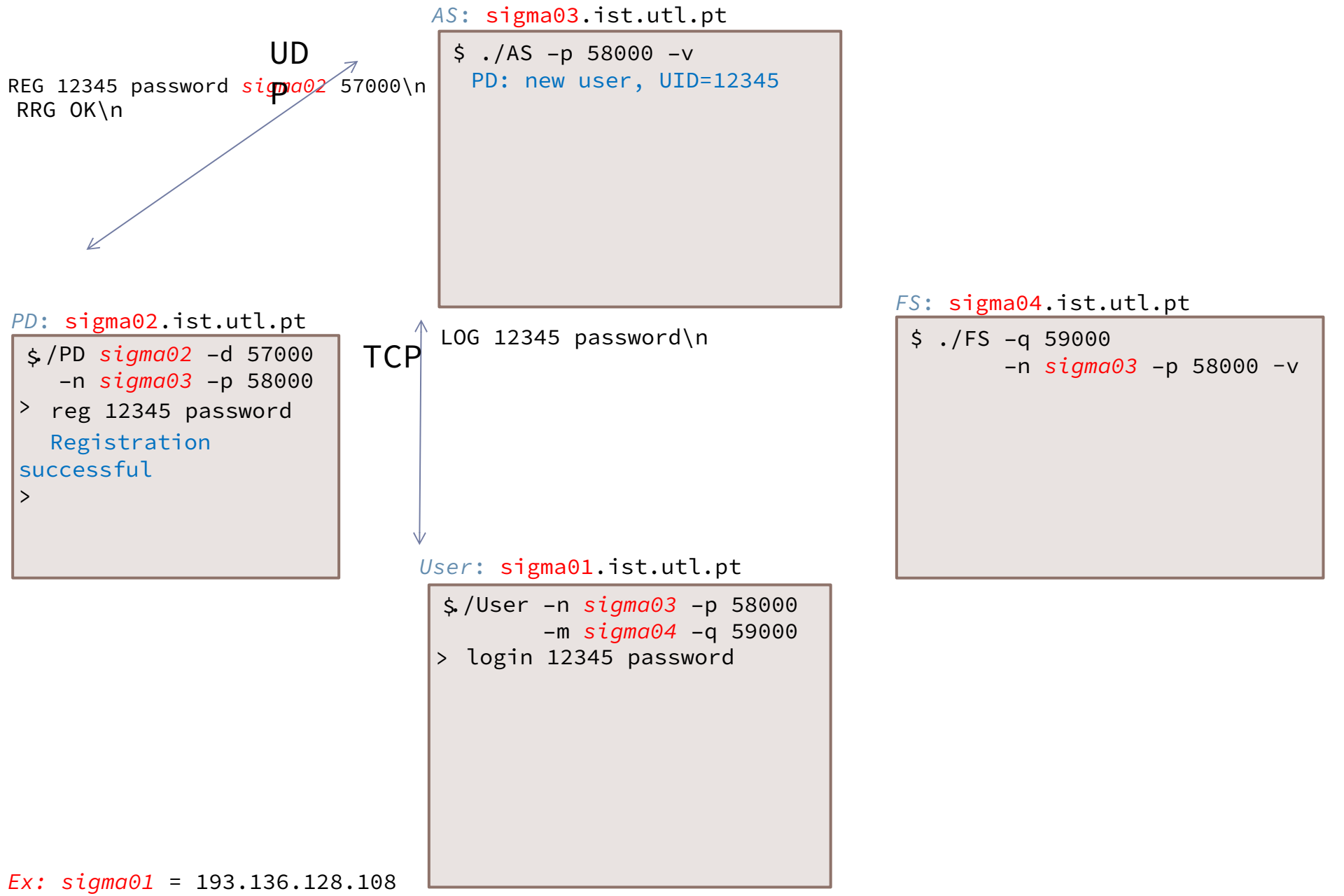
4 componentes:

- *PD*, *User*, *AS*, FS



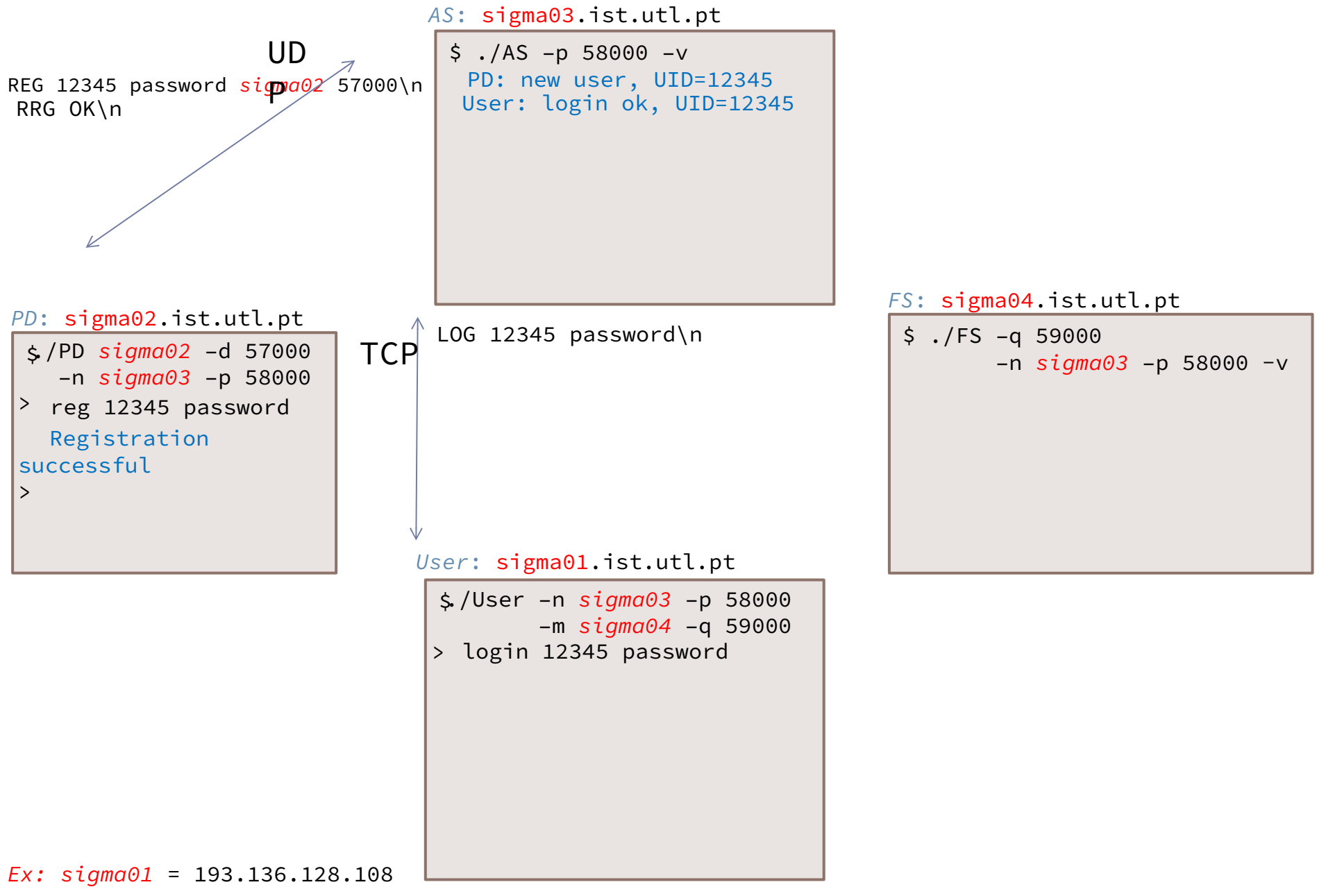
4 componentes:

- *PD*, *User*, *AS*, FS



4 componentes:

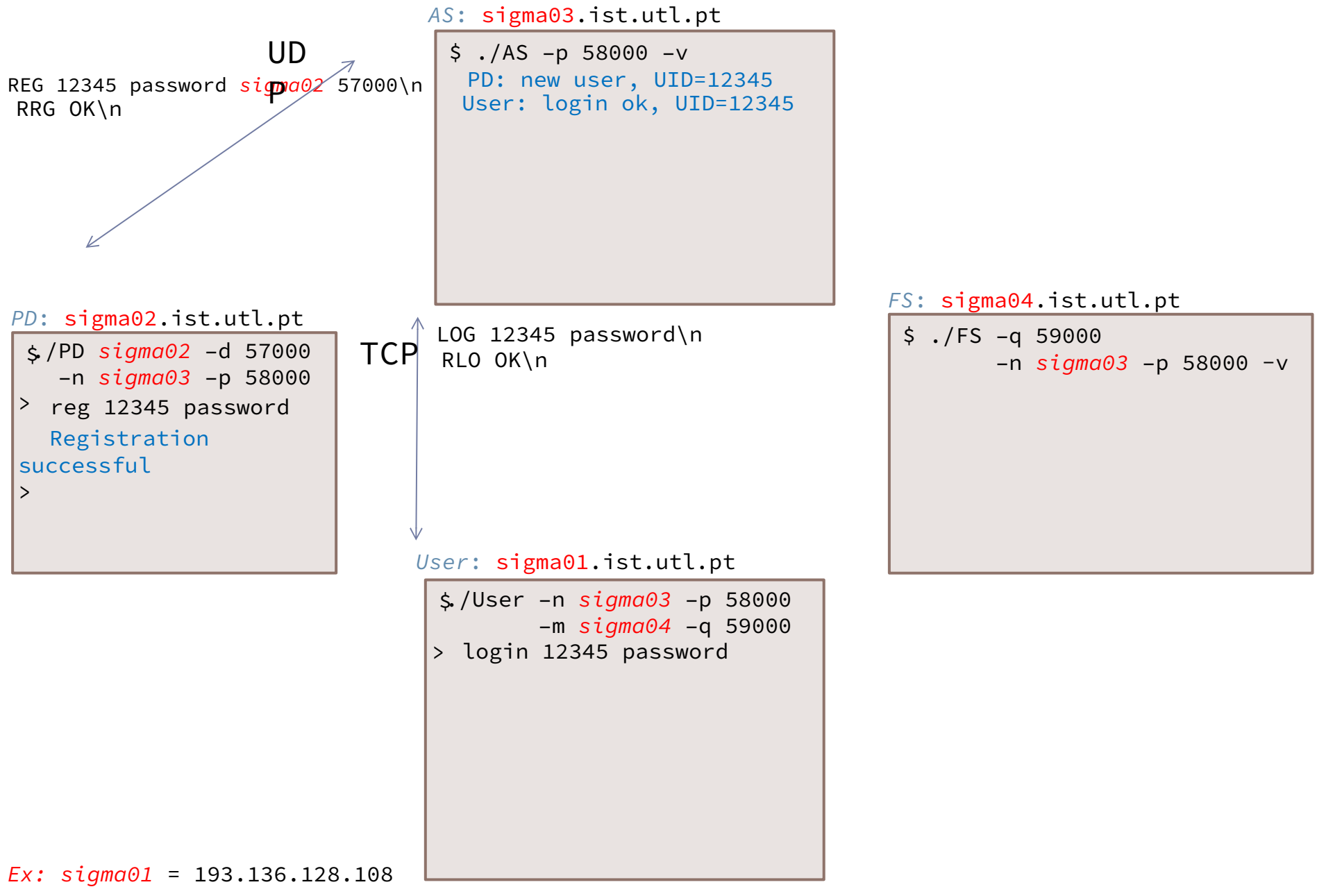
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

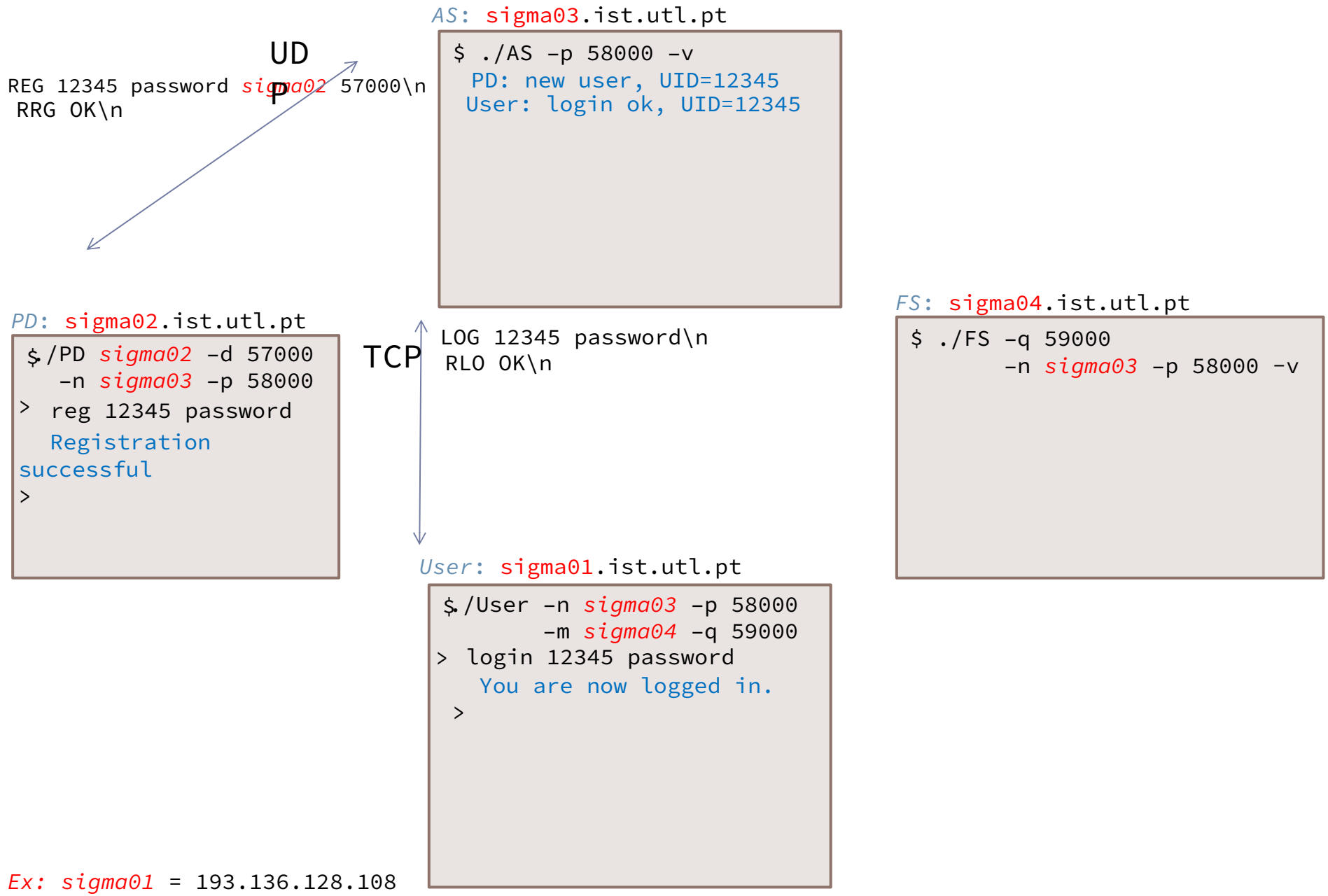
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

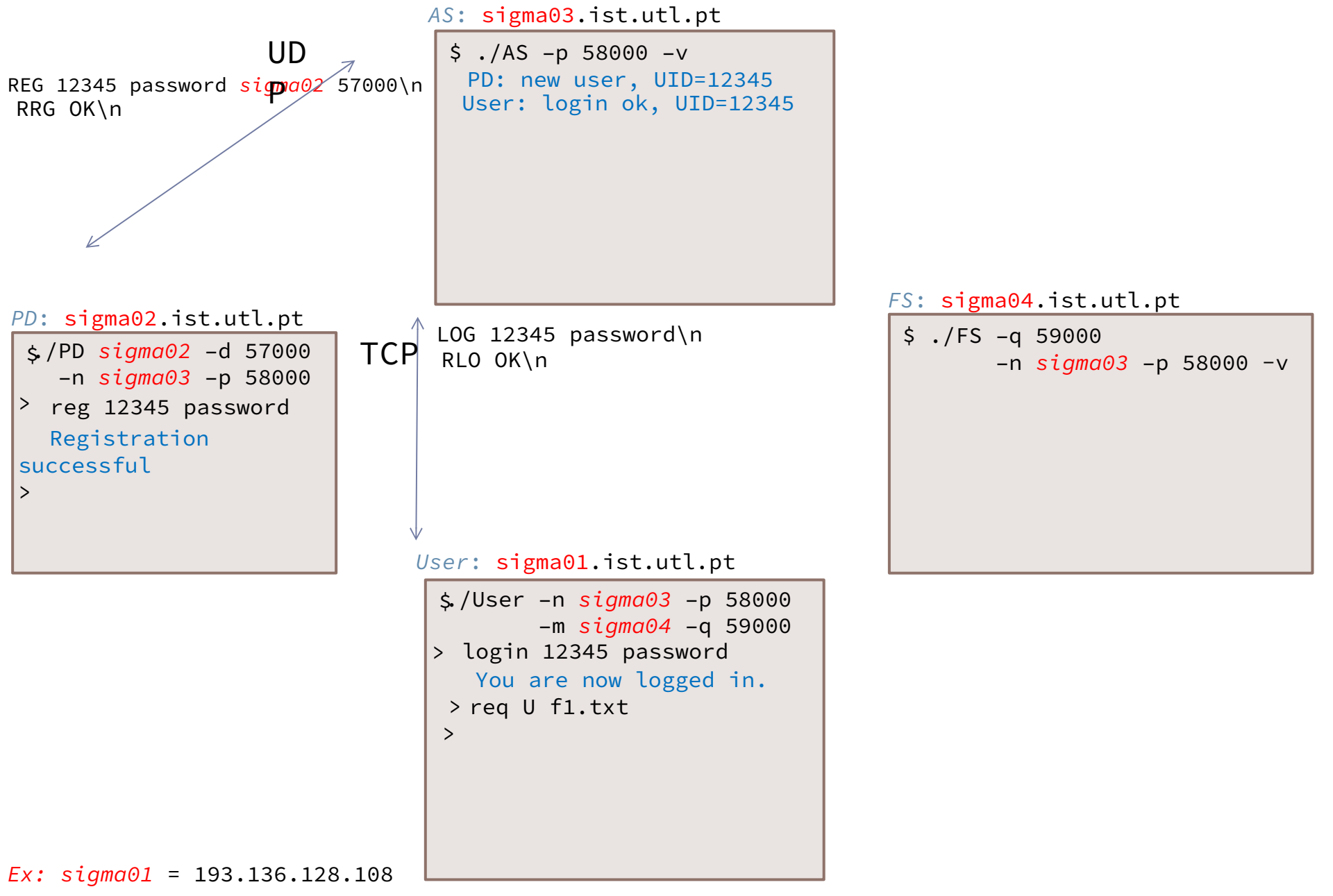
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

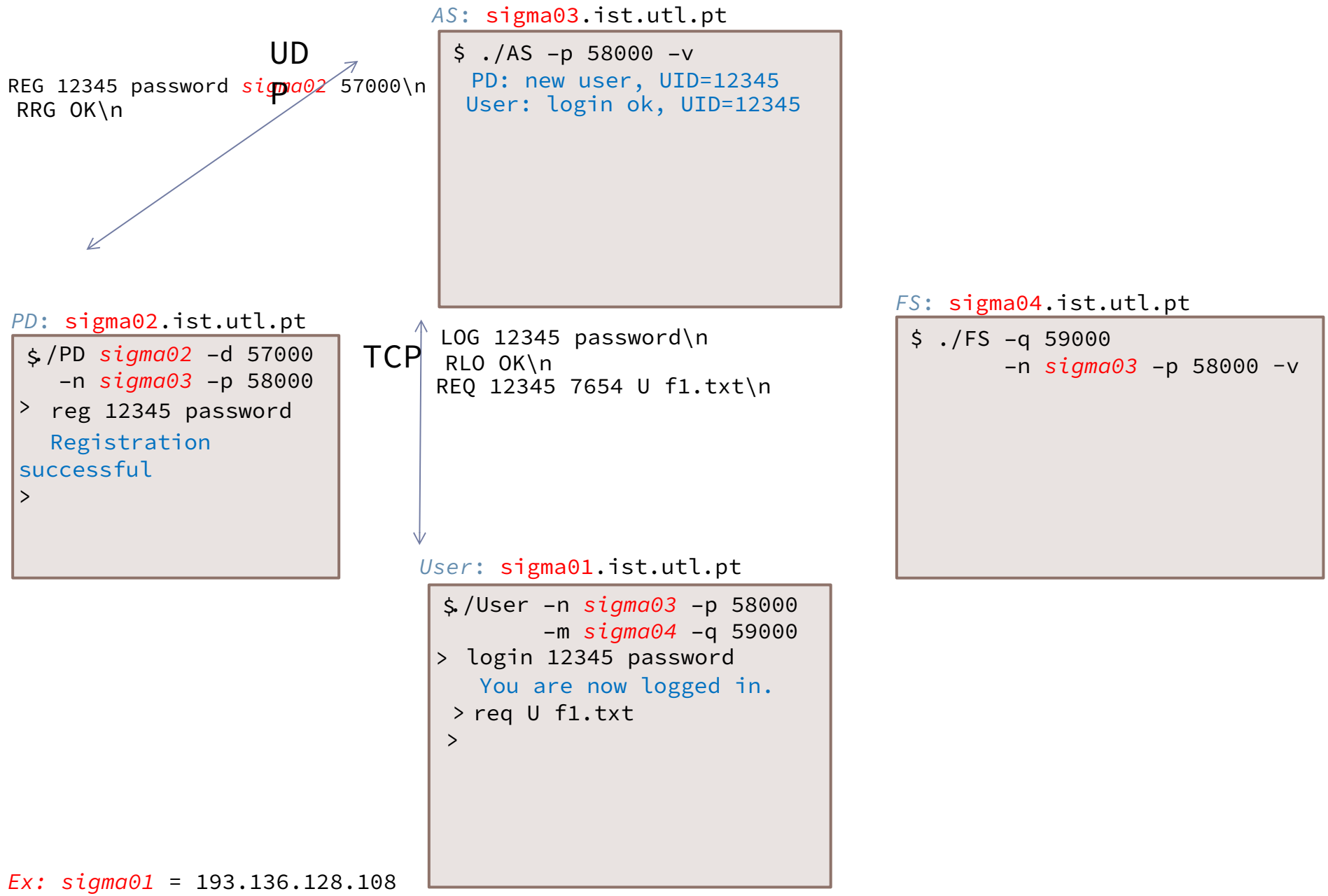
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

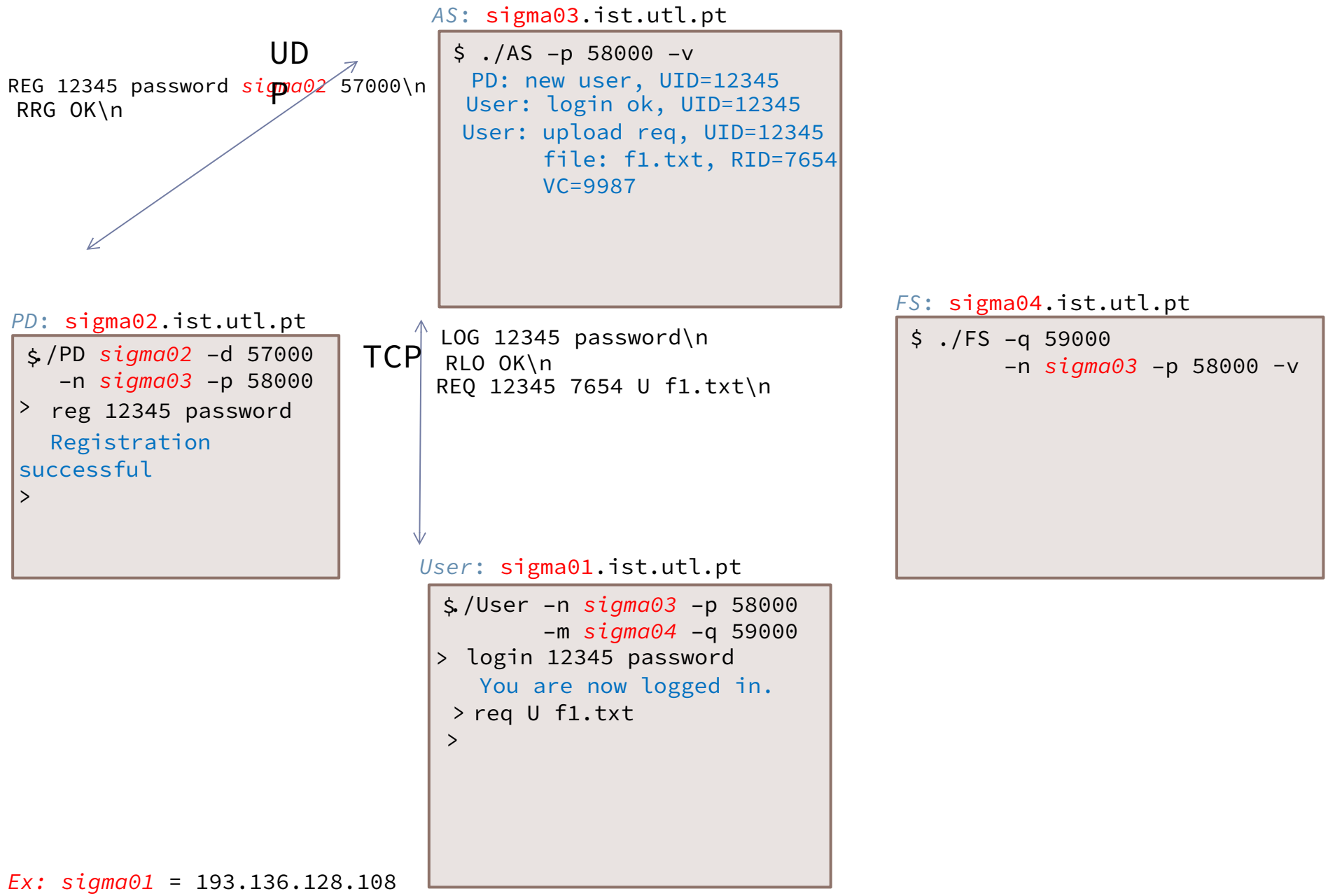
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

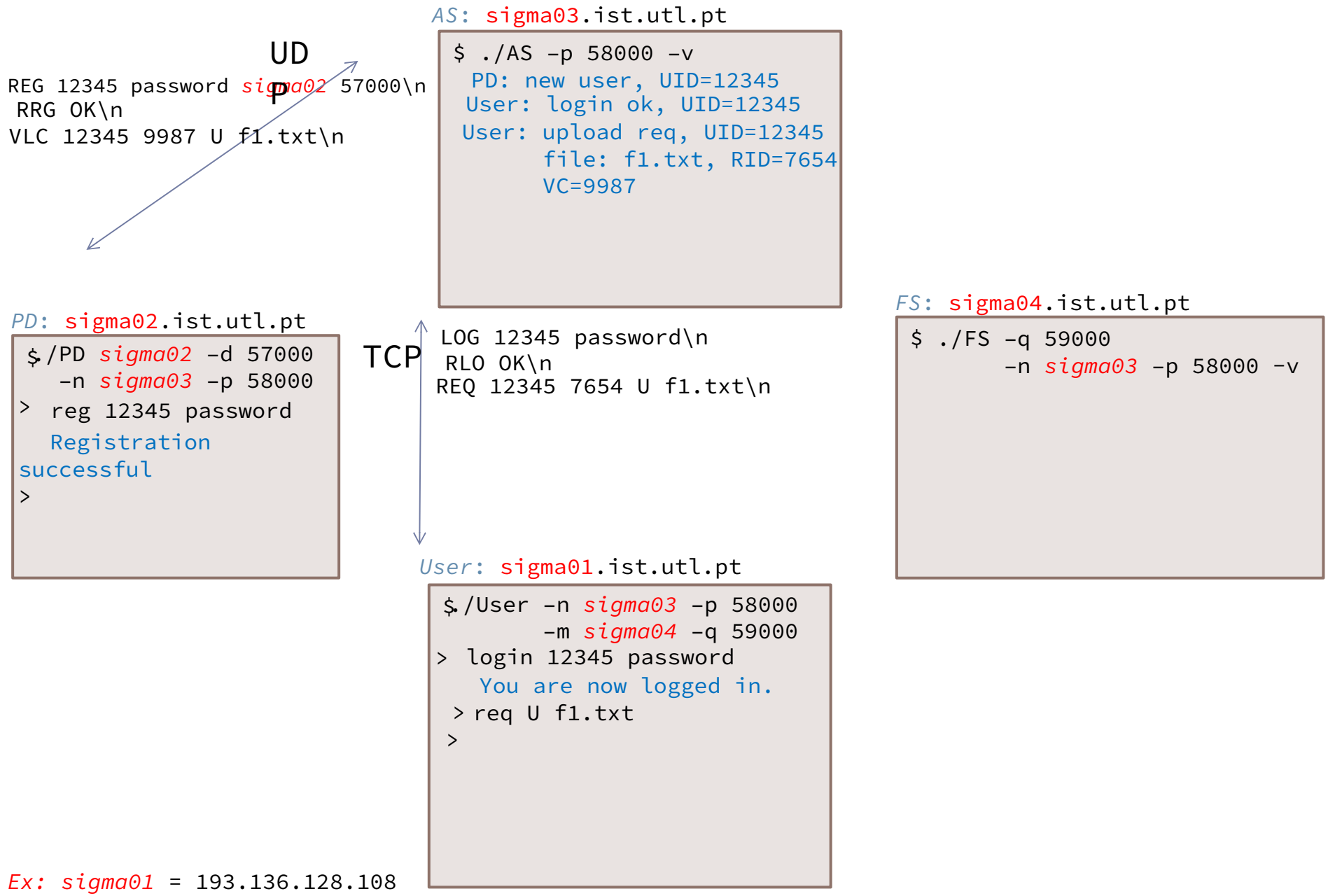
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

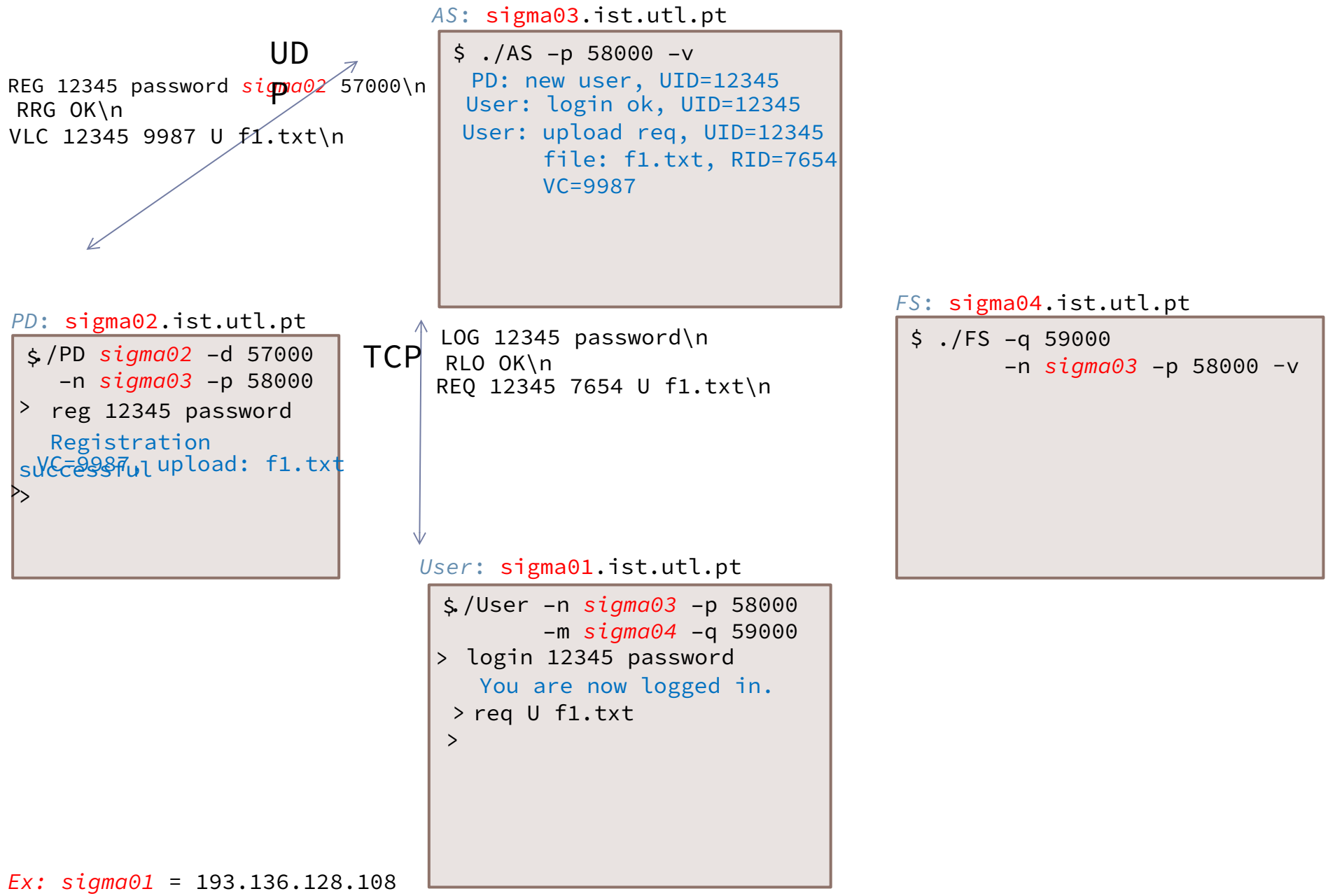
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

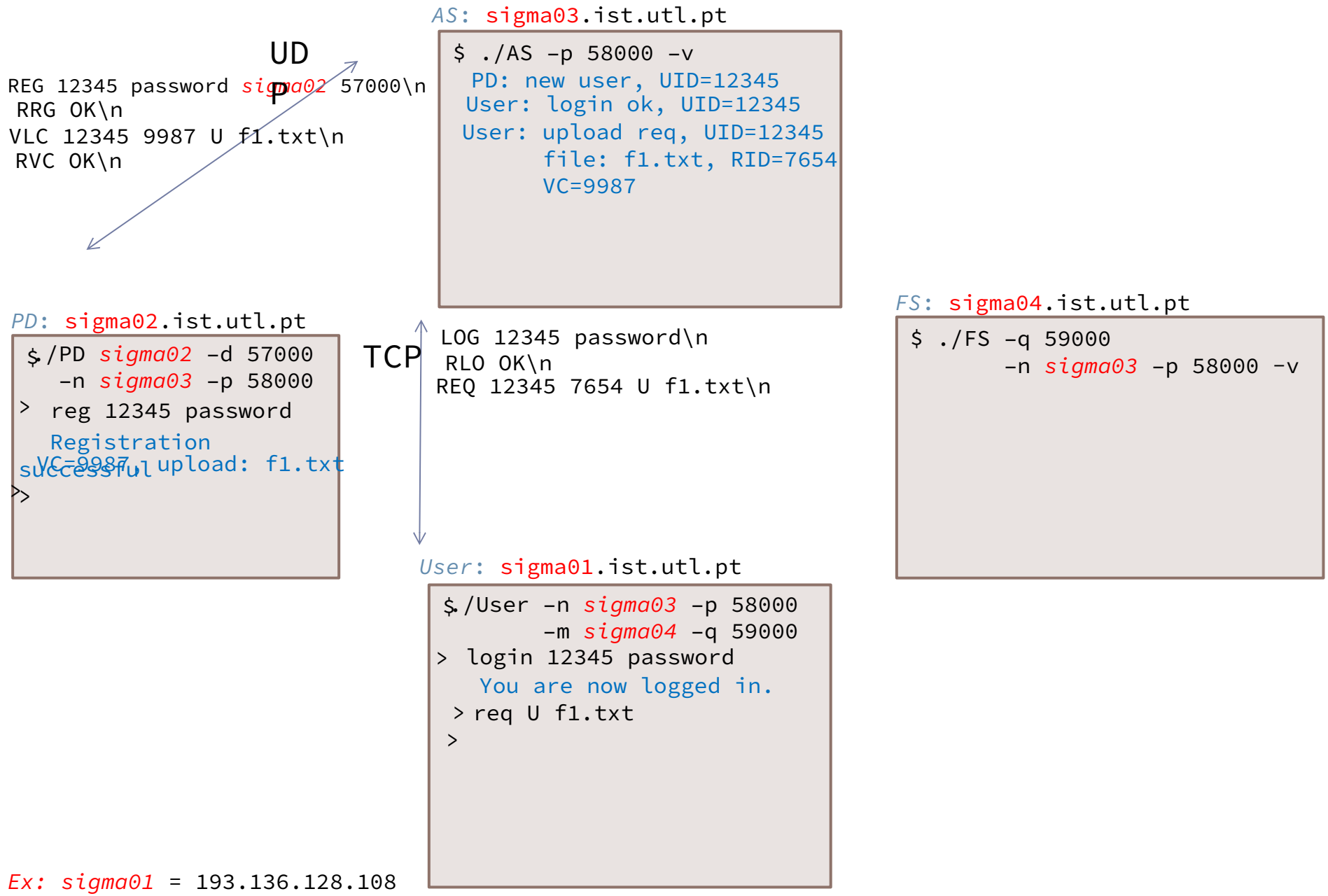
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

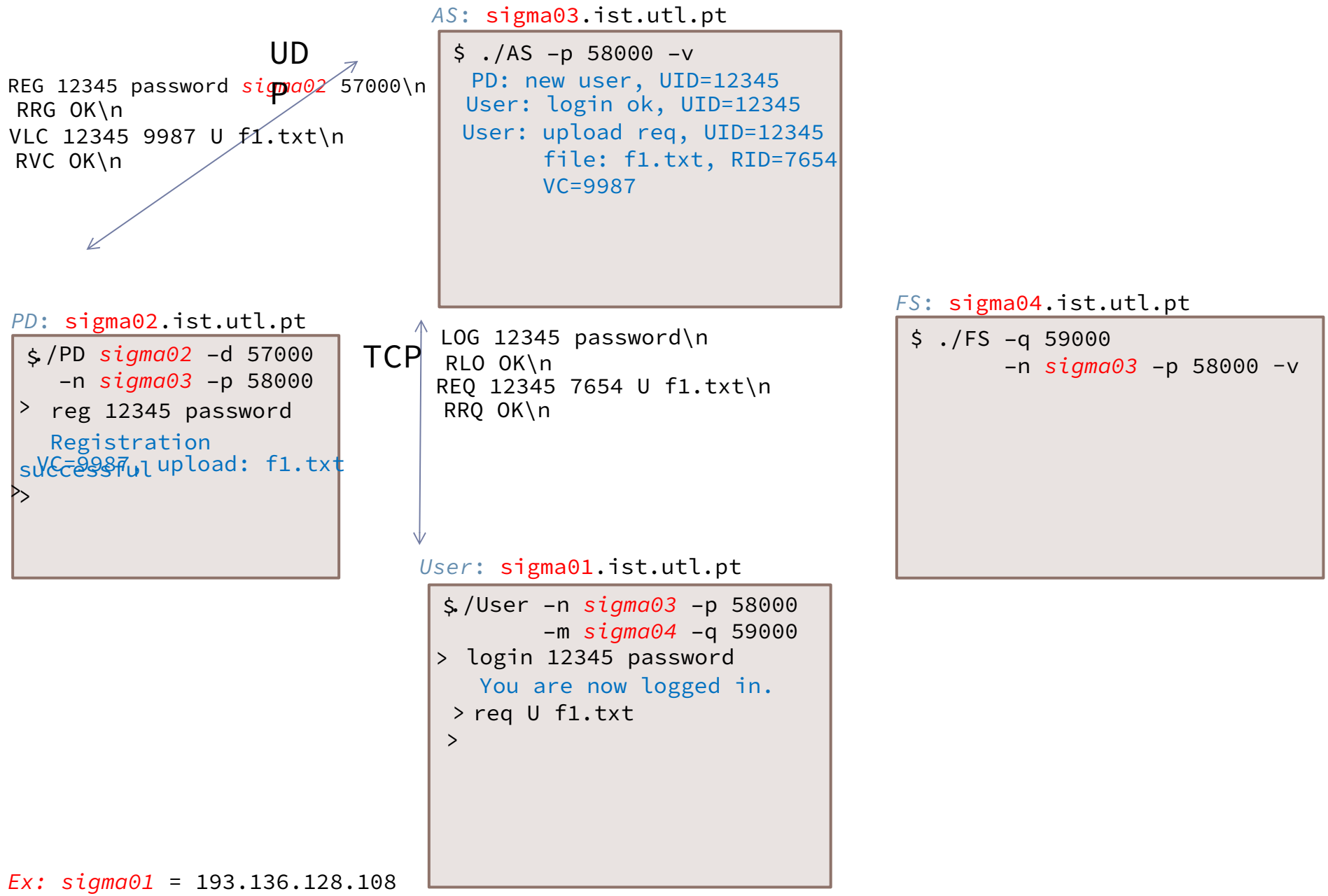
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

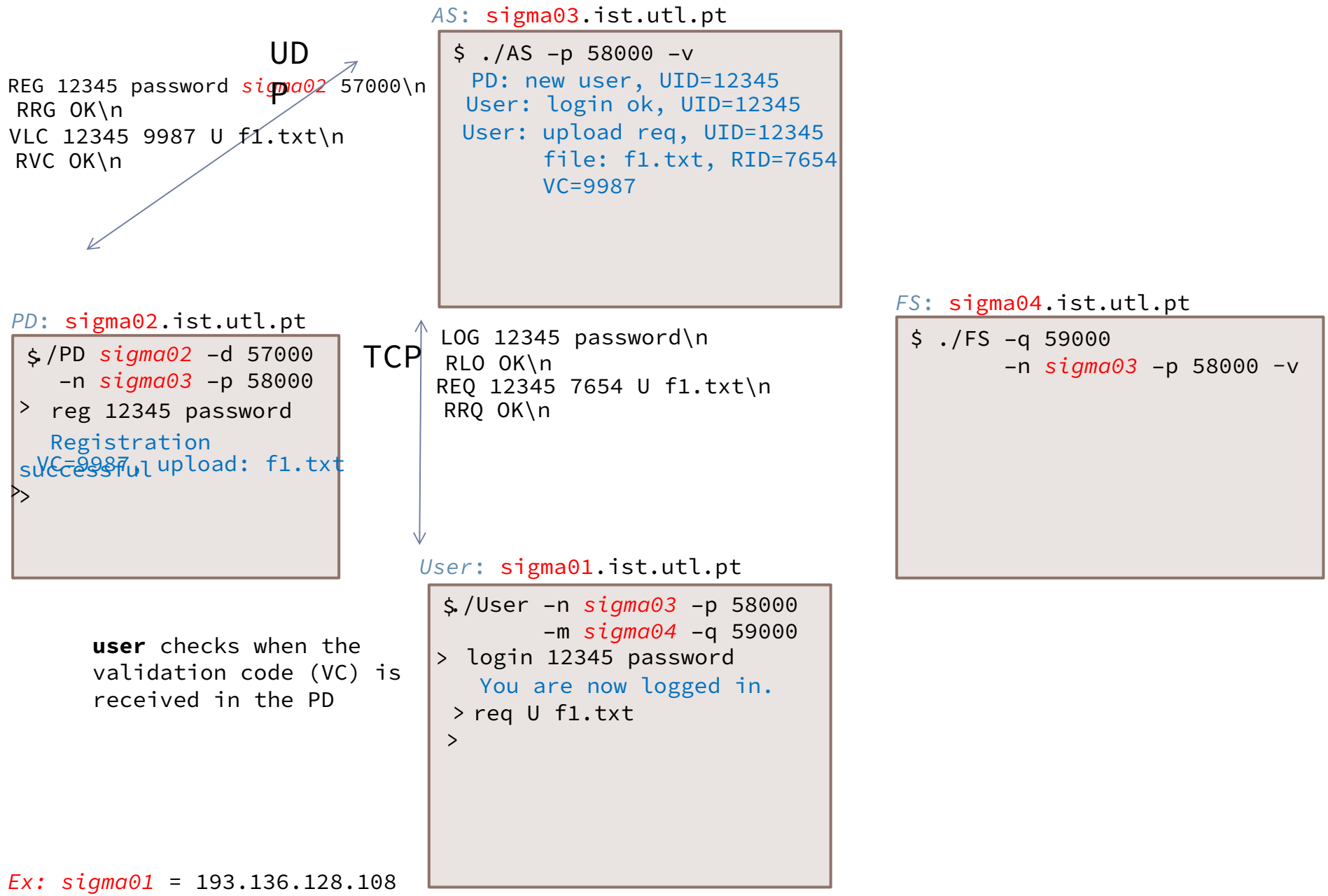
- *PD*, *User*, *AS*, FS



Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

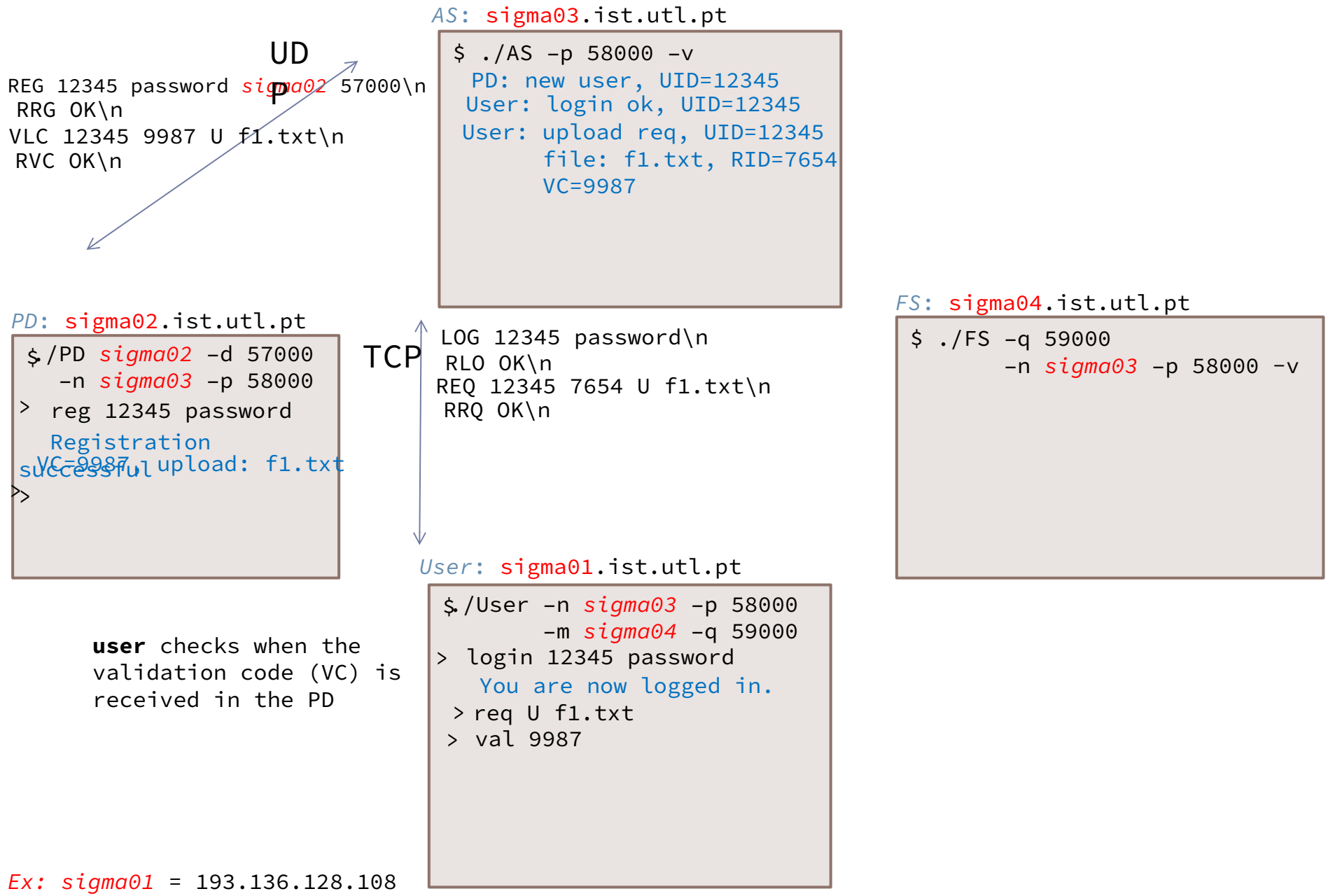


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

UDP

```
REG 12345 password sigma02 57000\n
 RRG OK\n
VLC 12345 9987 U f1.txt\n
 RVC OK\n
```

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
 User: login ok, UID=12345
 User: upload req, UID=12345
       file: f1.txt, RID=7654
       VC=9987
```

PD: sigma02.ist.utl.pt

```
$./PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
   Registration
 VC=9987 upload: f1.txt
successful
>>
```

TCP

```
LOG 12345 password\n
 RLO OK\n
REQ 12345 7654 U f1.txt\n
 RRQ OK\n
AUT 12345 7654 9987\n
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

User: sigma01.ist.utl.pt

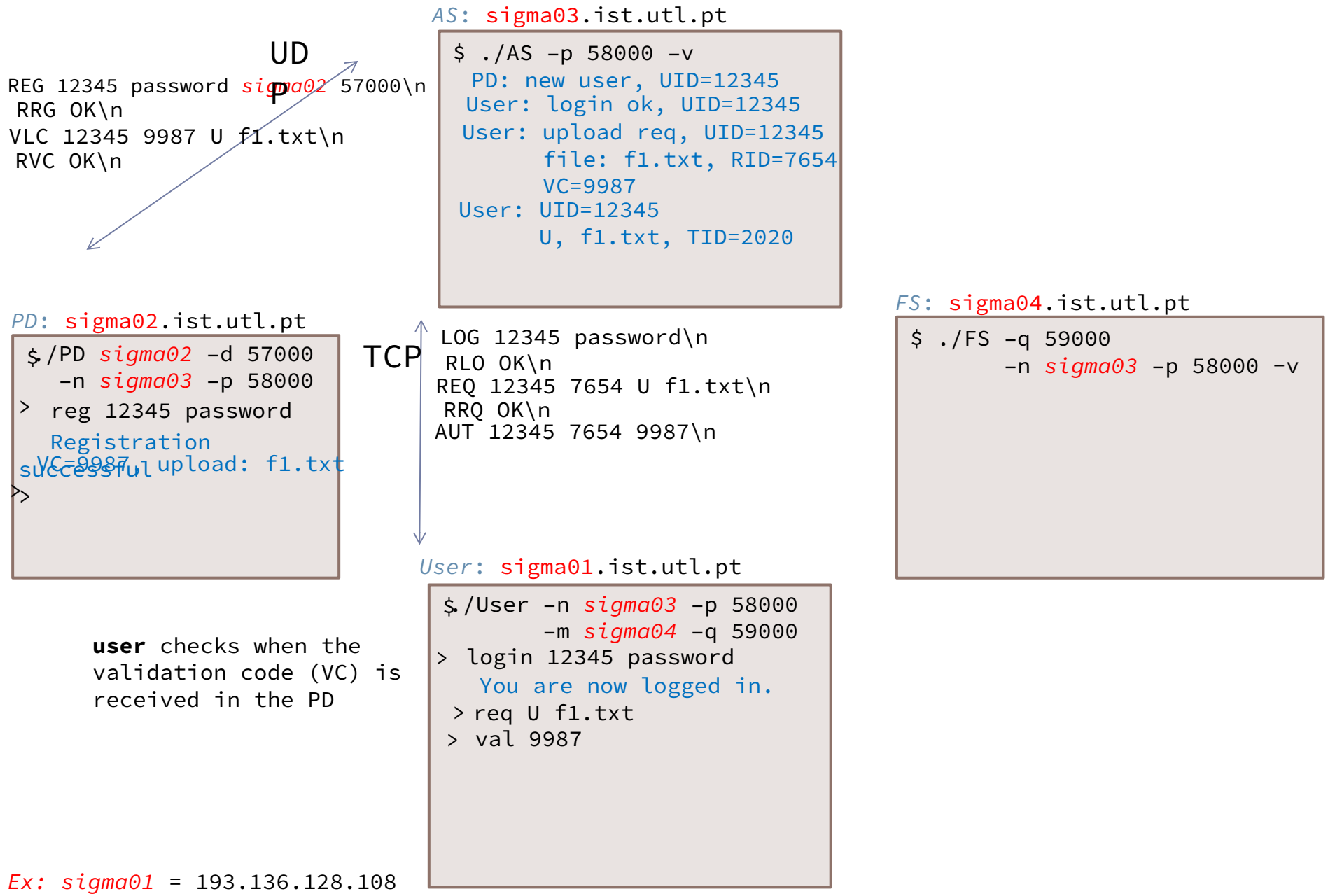
```
$./User -n sigma03 -p 58000
        -m sigma04 -q 59000
> login 12345 password
   You are now logged in.
 > req U f1.txt
 > val 9987
```

user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

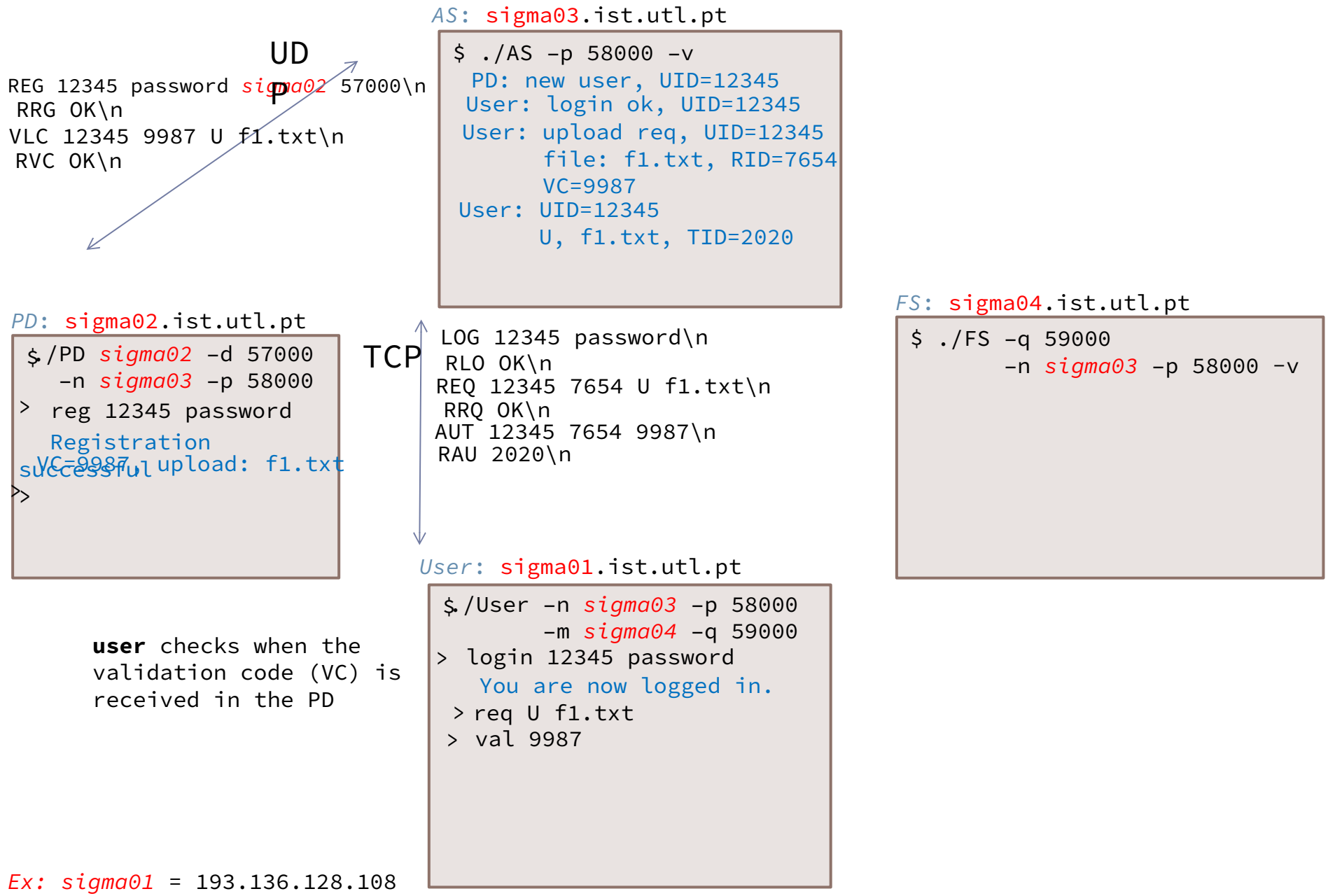


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD, User, AS,* FS

UDP

```
REG 12345 password sigma02 57000\n
 RRG OK\n
VLC 12345 9987 U f1.txt\n
 RVC OK\n
```

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
 User: login ok, UID=12345
 User: upload req, UID=12345
       file: f1.txt, RID=7654
       VC=9987
 User: UID=12345
       U, f1.txt, TID=2020
```

PD: sigma02.ist.utl.pt

```
$./PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
   Registration
   VC=9987 upload: f1.txt
successful
>>
```

TCP

```
LOG 12345 password\n
 RLO OK\n
REQ 12345 7654 U f1.txt\n
 RRQ OK\n
AUT 12345 7654 9987\n
RAU 2020\n
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

User: sigma01.ist.utl.pt

```
$./User -n sigma03 -p 58000
        -m sigma04 -q 59000
> login 12345 password
   You are now logged in.
 > req U f1.txt
 > val 9987
   Authenticated! (TID=2020)
 >
```

user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD, User, AS,* FS

UDP

```
REG 12345 password sigma02 57000\n
 RRG OK\n
VLC 12345 9987 U f1.txt\n
 RVC OK\n
```

AS: sigma03.ist.utl.pt

```
$ ./AS -p 58000 -v
  PD: new user, UID=12345
 User: login ok, UID=12345
 User: upload req, UID=12345
       file: f1.txt, RID=7654
       VC=9987
 User: UID=12345
       U, f1.txt, TID=2020
```

PD: sigma02.ist.utl.pt

```
$ /PD sigma02 -d 57000
   -n sigma03 -p 58000
> reg 12345 password
   Registration
   VC=9987 successful
   upload: f1.txt
>>
```

TCP

```
LOG 12345 password\n
 RLO OK\n
REQ 12345 7654 U f1.txt\n
 RRQ OK\n
AUT 12345 7654 9987\n
RAU 2020\n
```

FS: sigma04.ist.utl.pt

```
$ ./FS -q 59000
       -n sigma03 -p 58000 -v
```

User: sigma01.ist.utl.pt

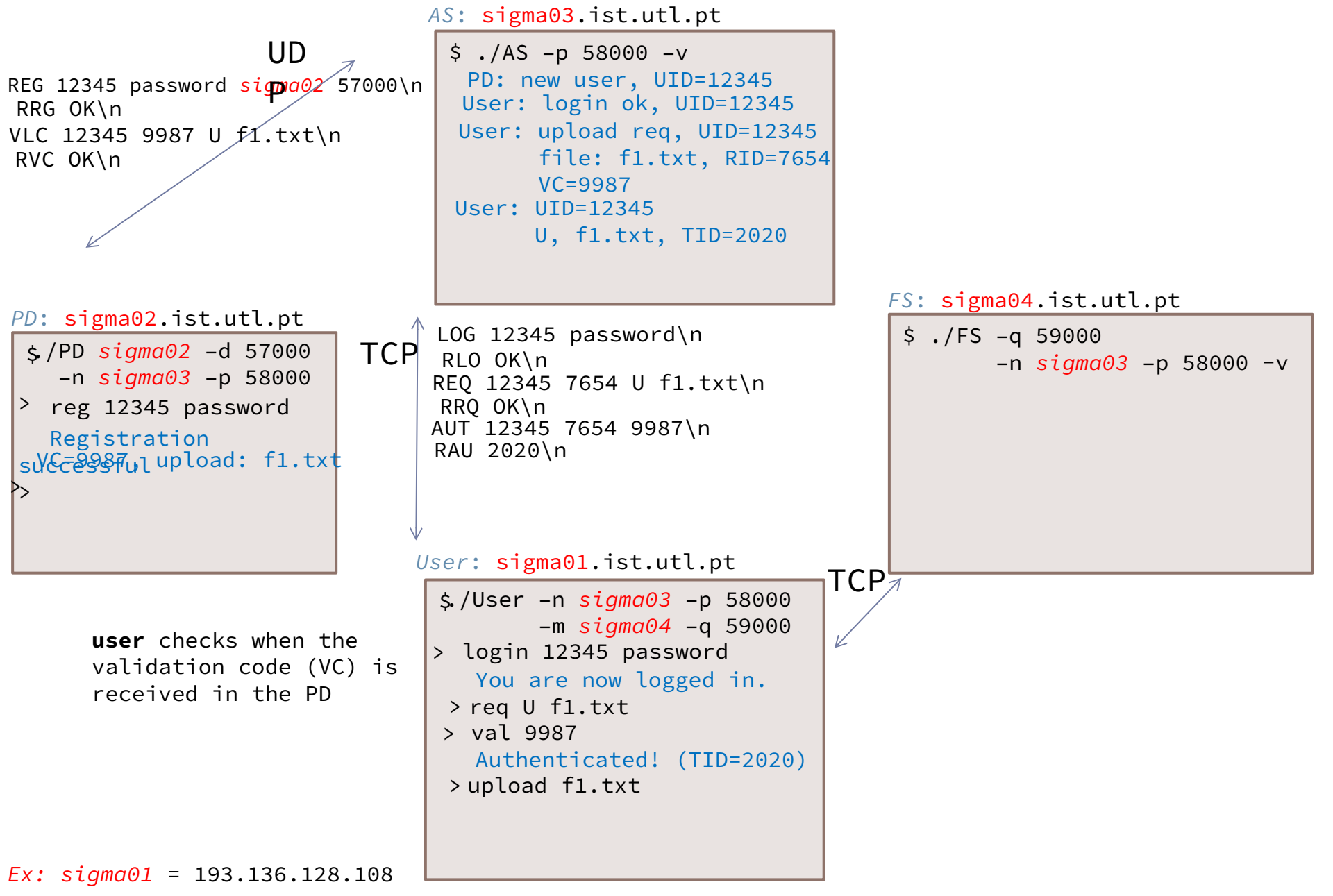
```
$ ./User -n sigma03 -p 58000
         -m sigma04 -q 59000
> login 12345 password
   You are now logged in.
 > req U f1.txt
 > val 9987
   Authenticated! (TID=2020)
 > upload f1.txt
```

user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD, User, AS,* FS

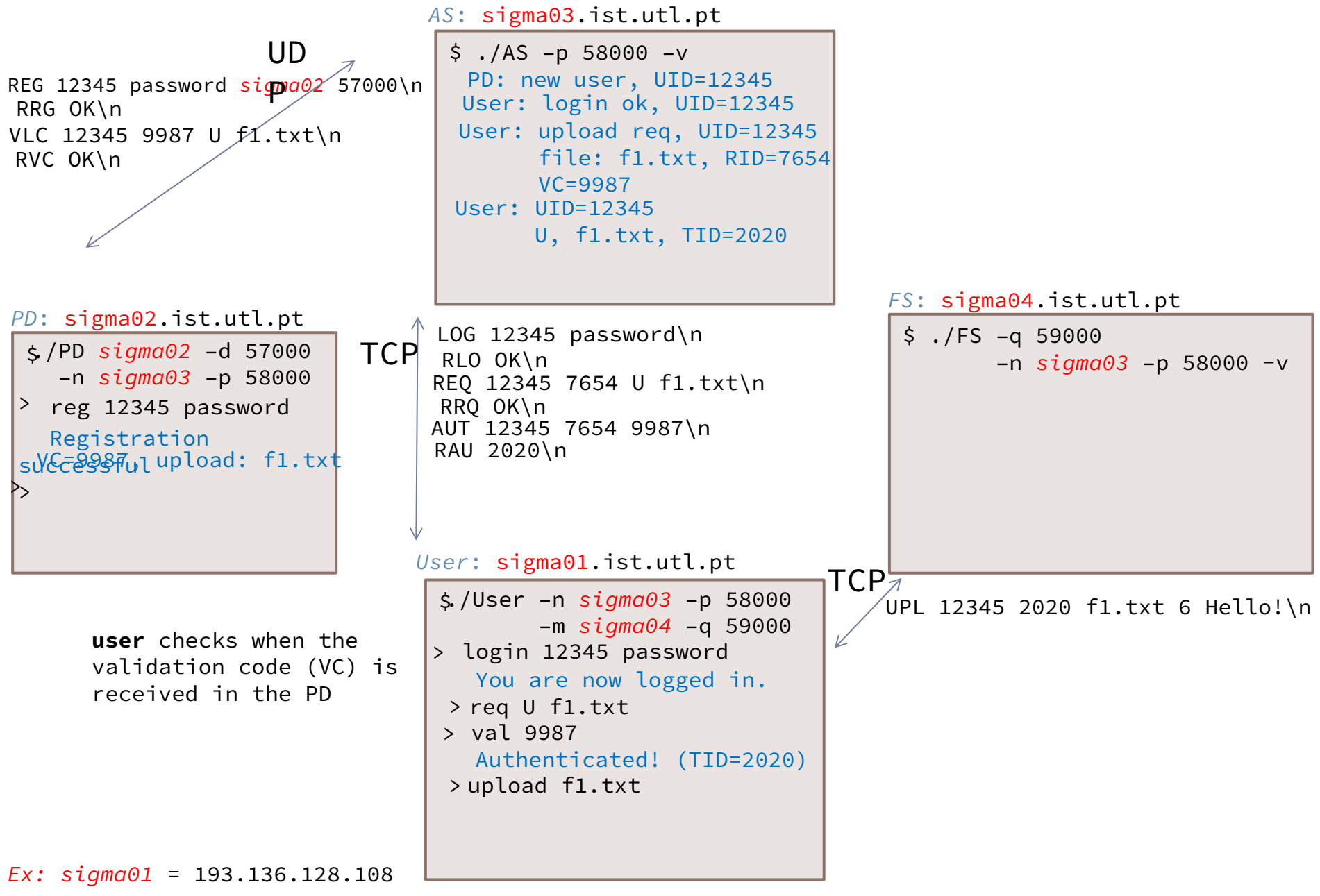


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

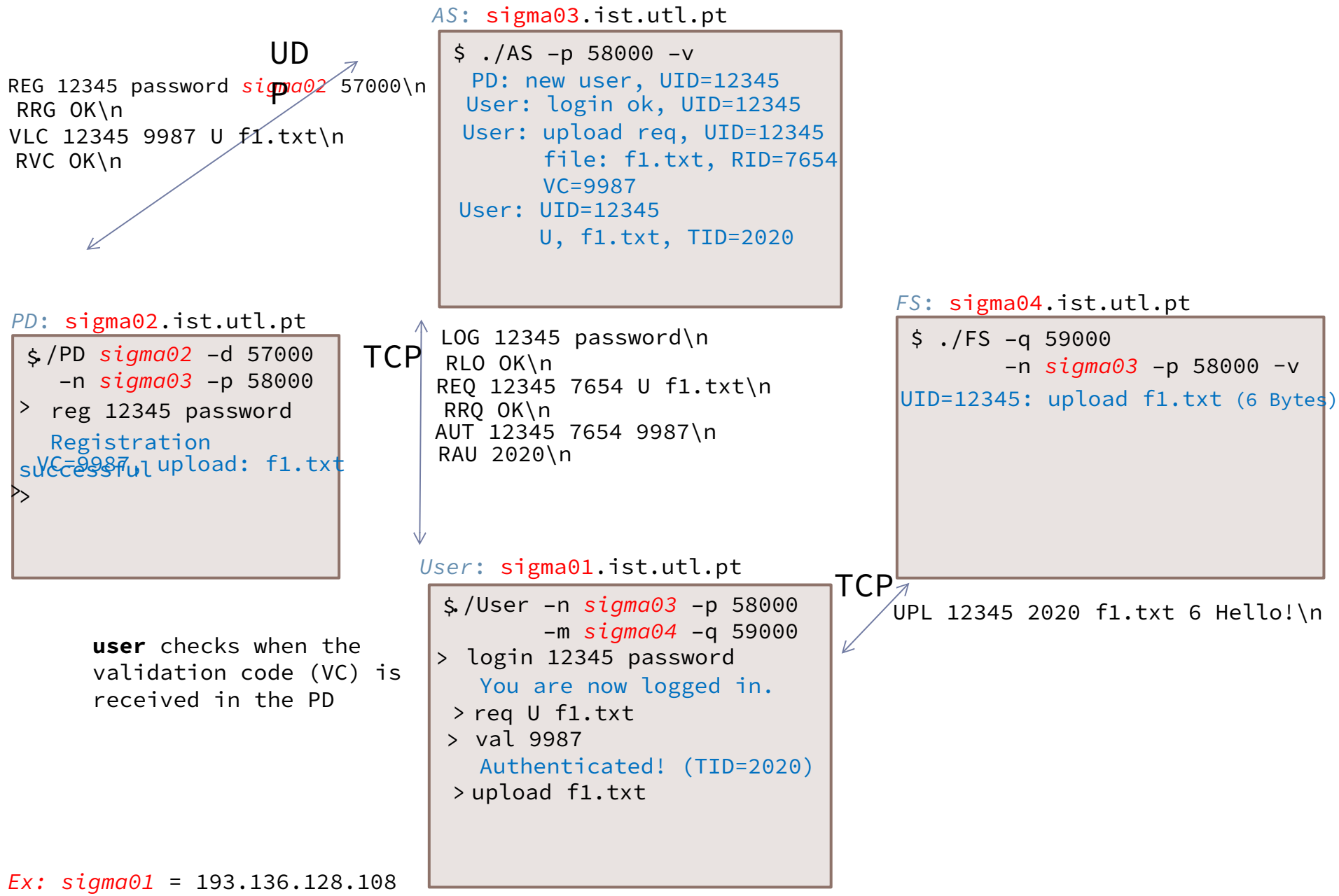


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

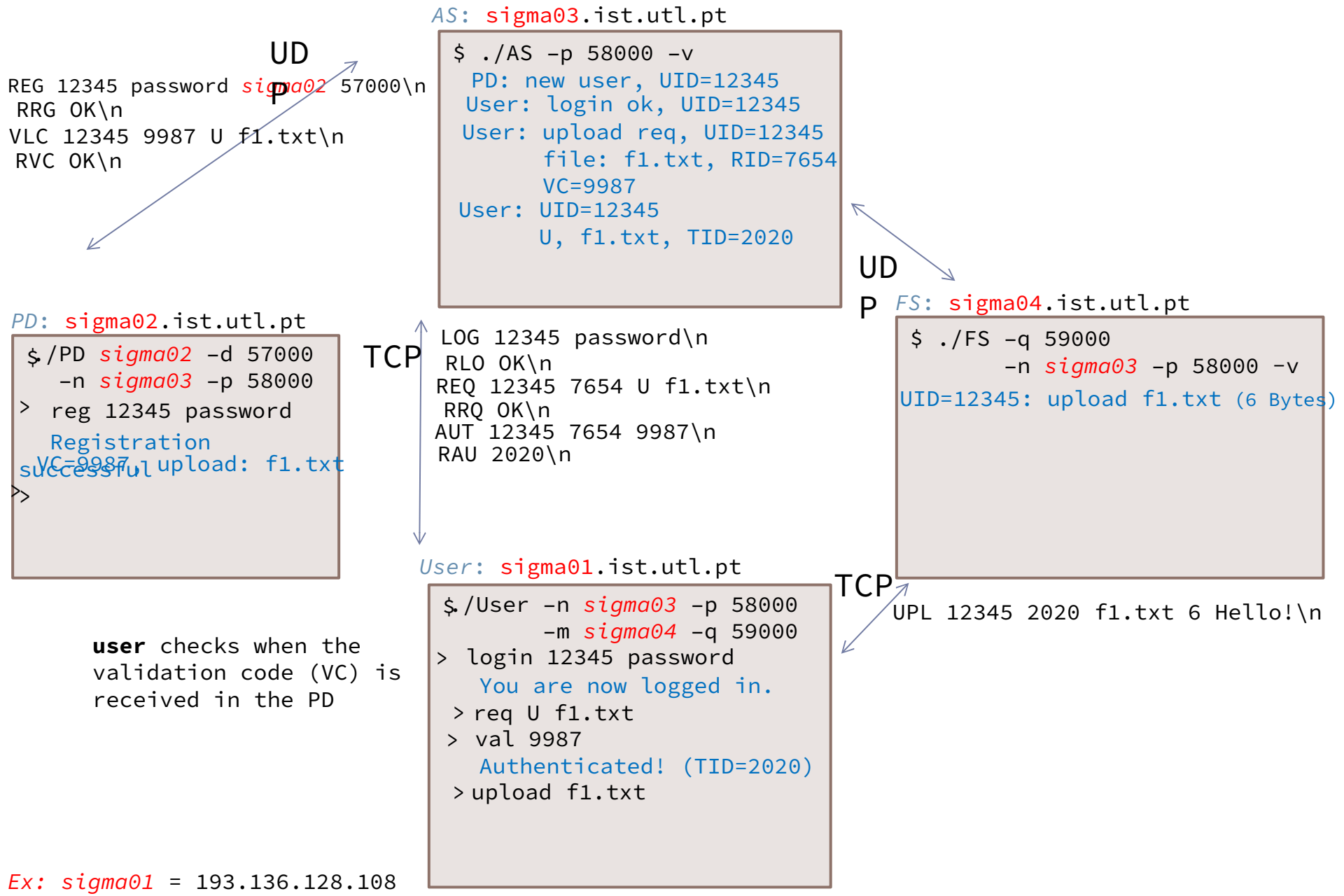


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

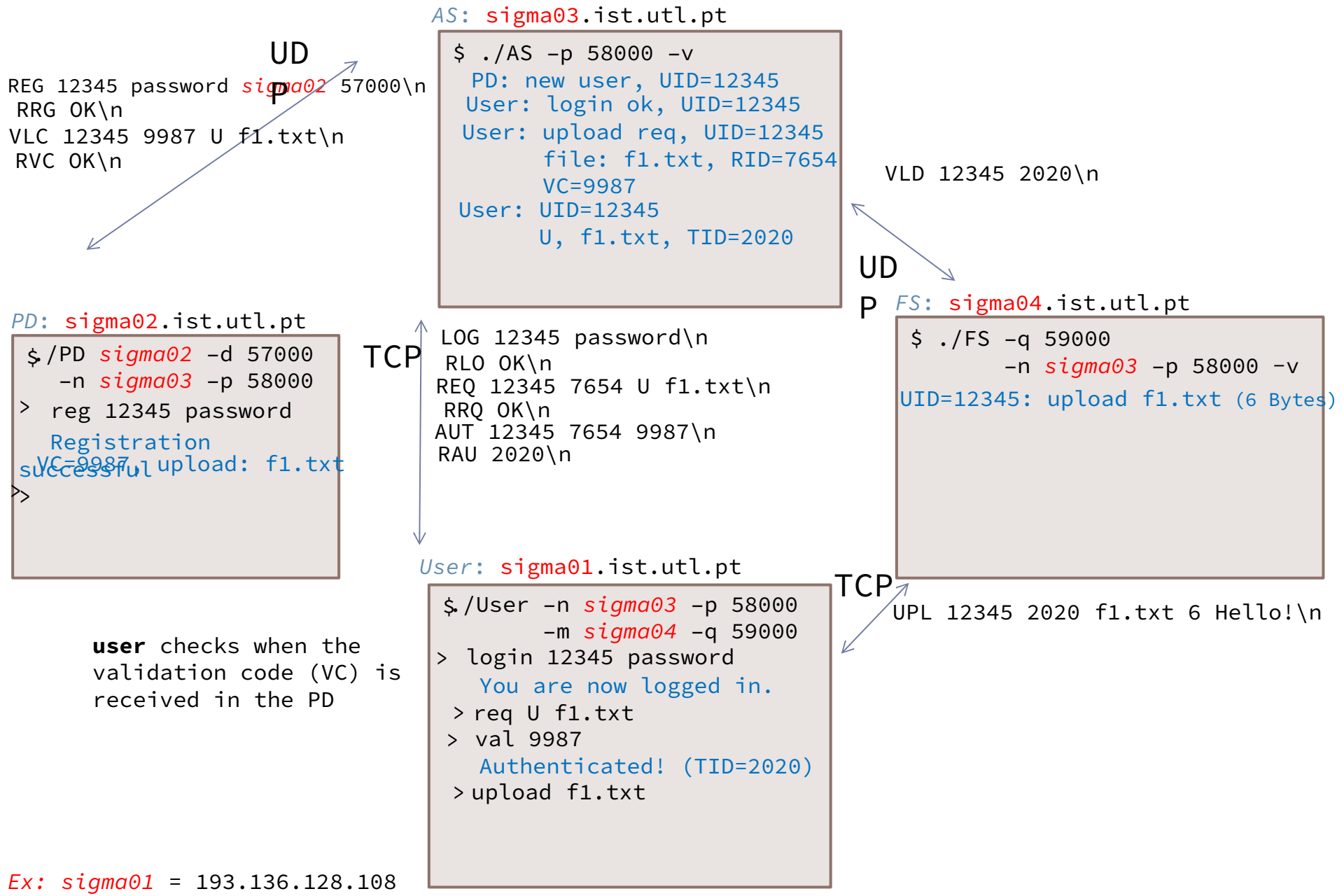


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

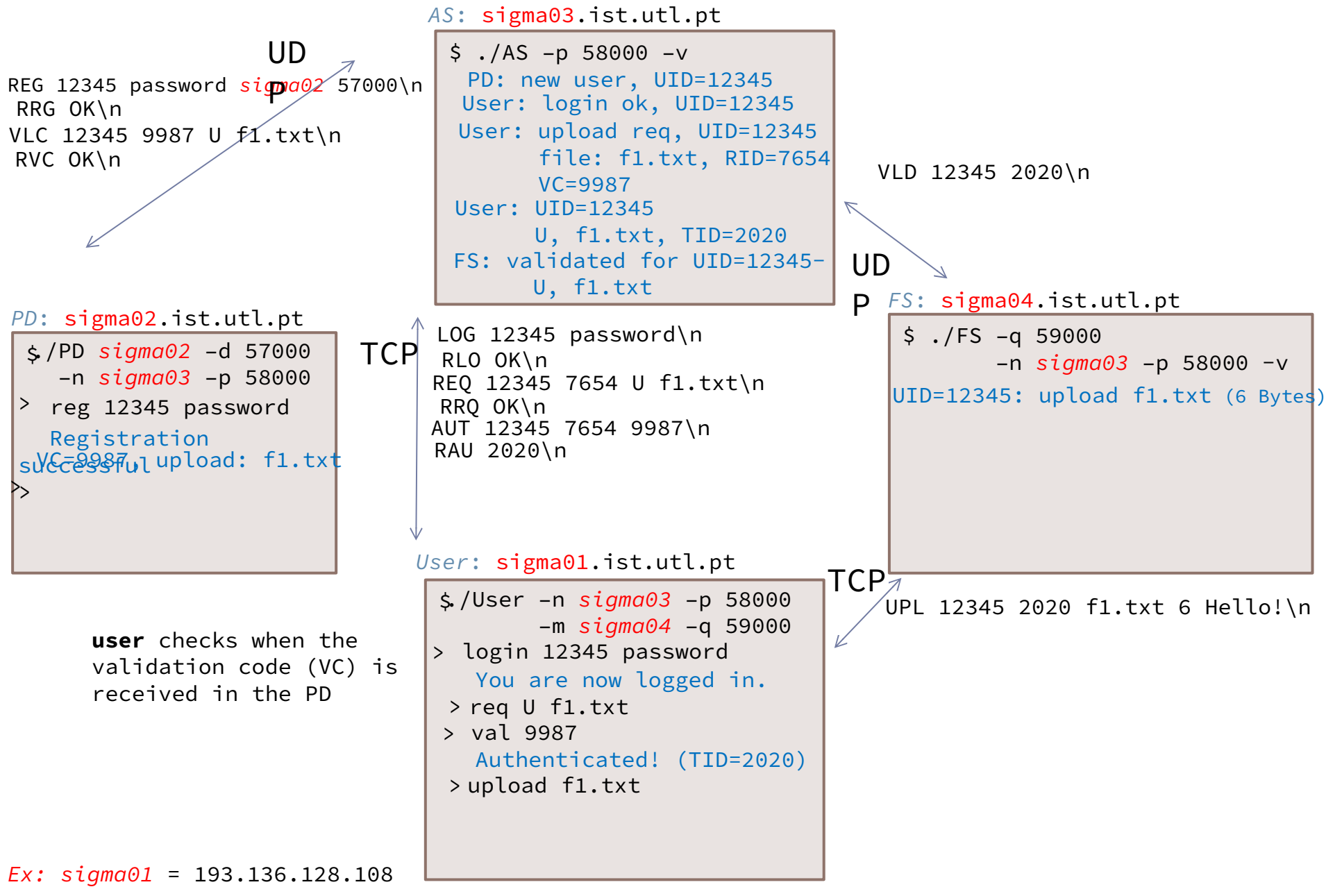


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

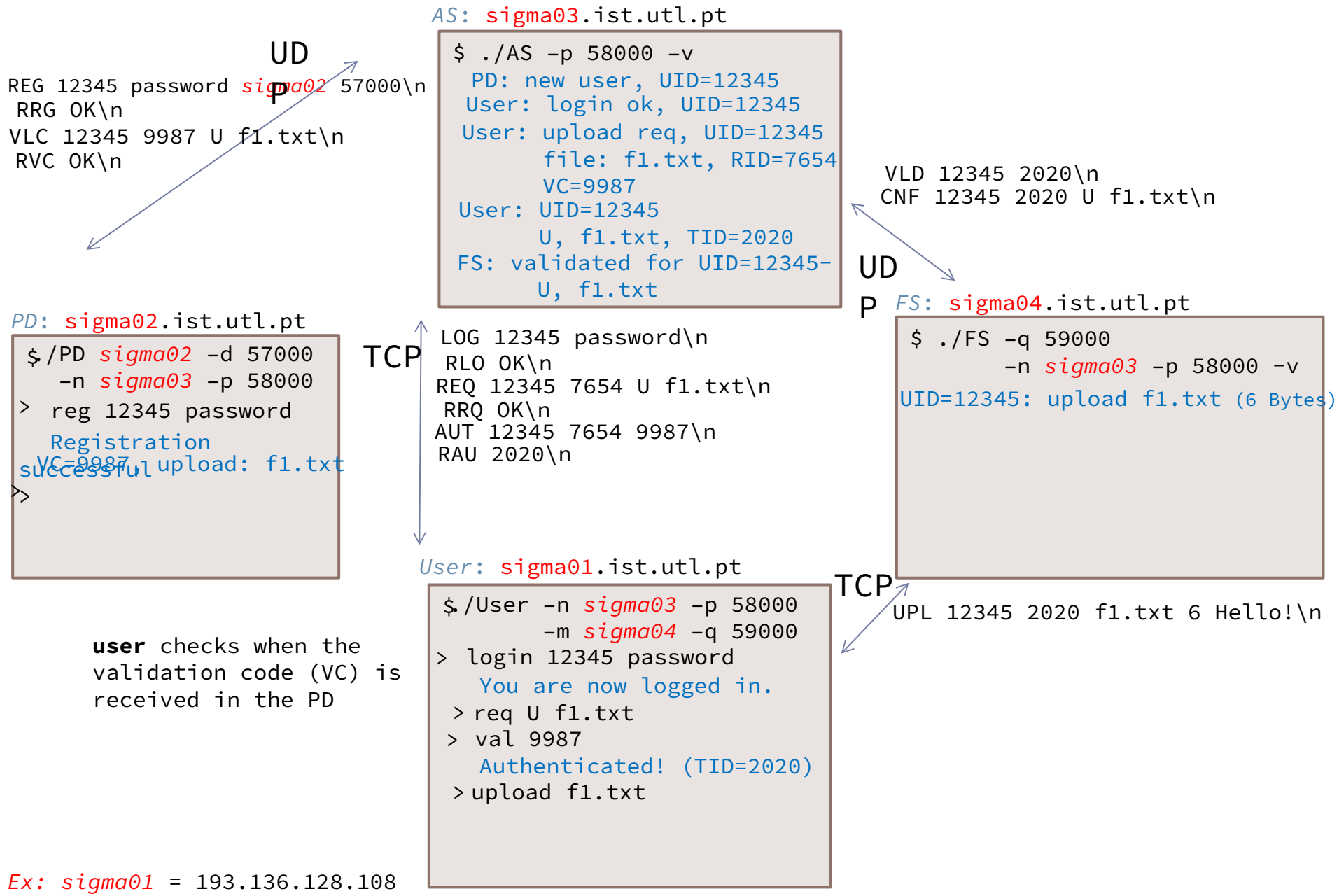


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

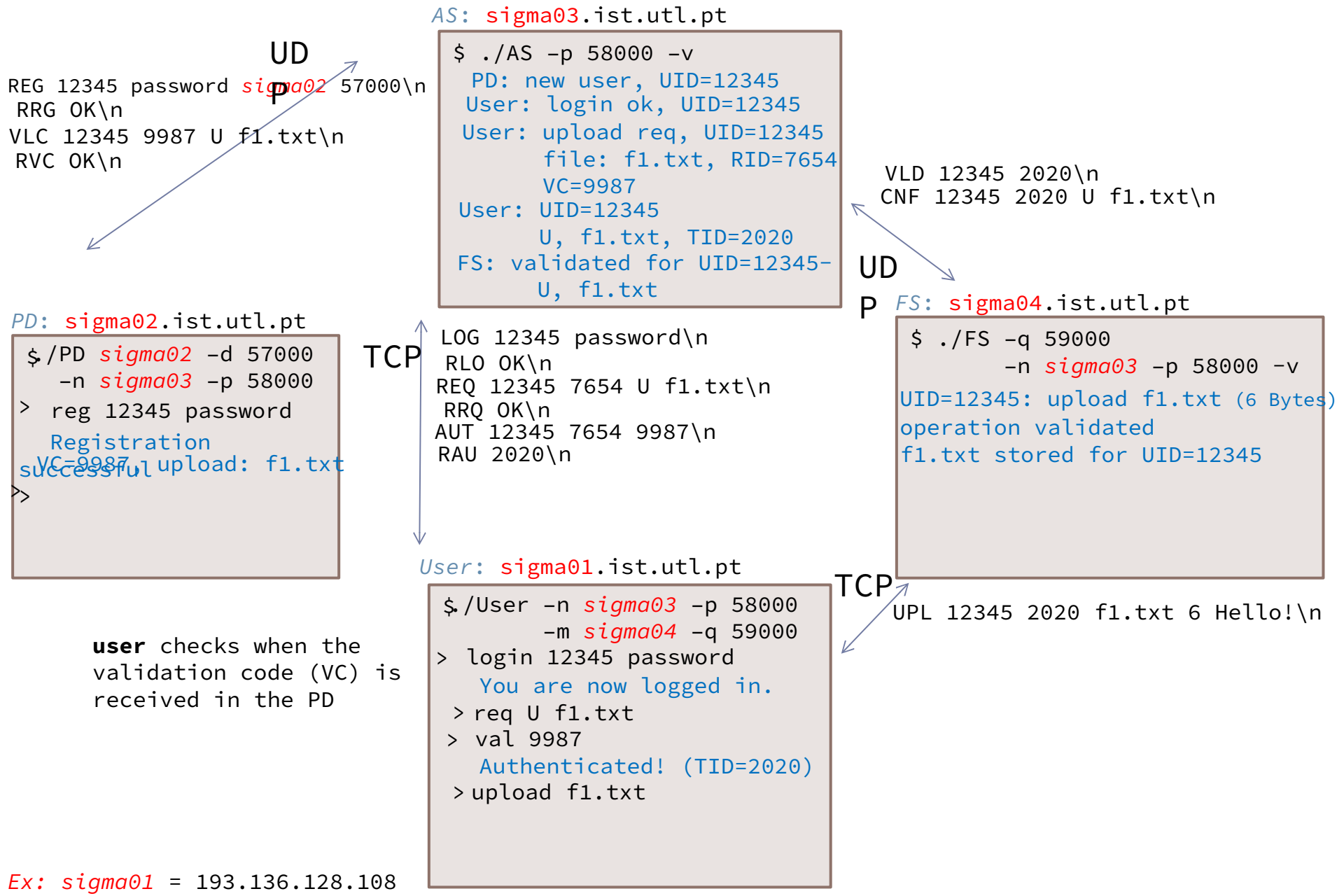


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS

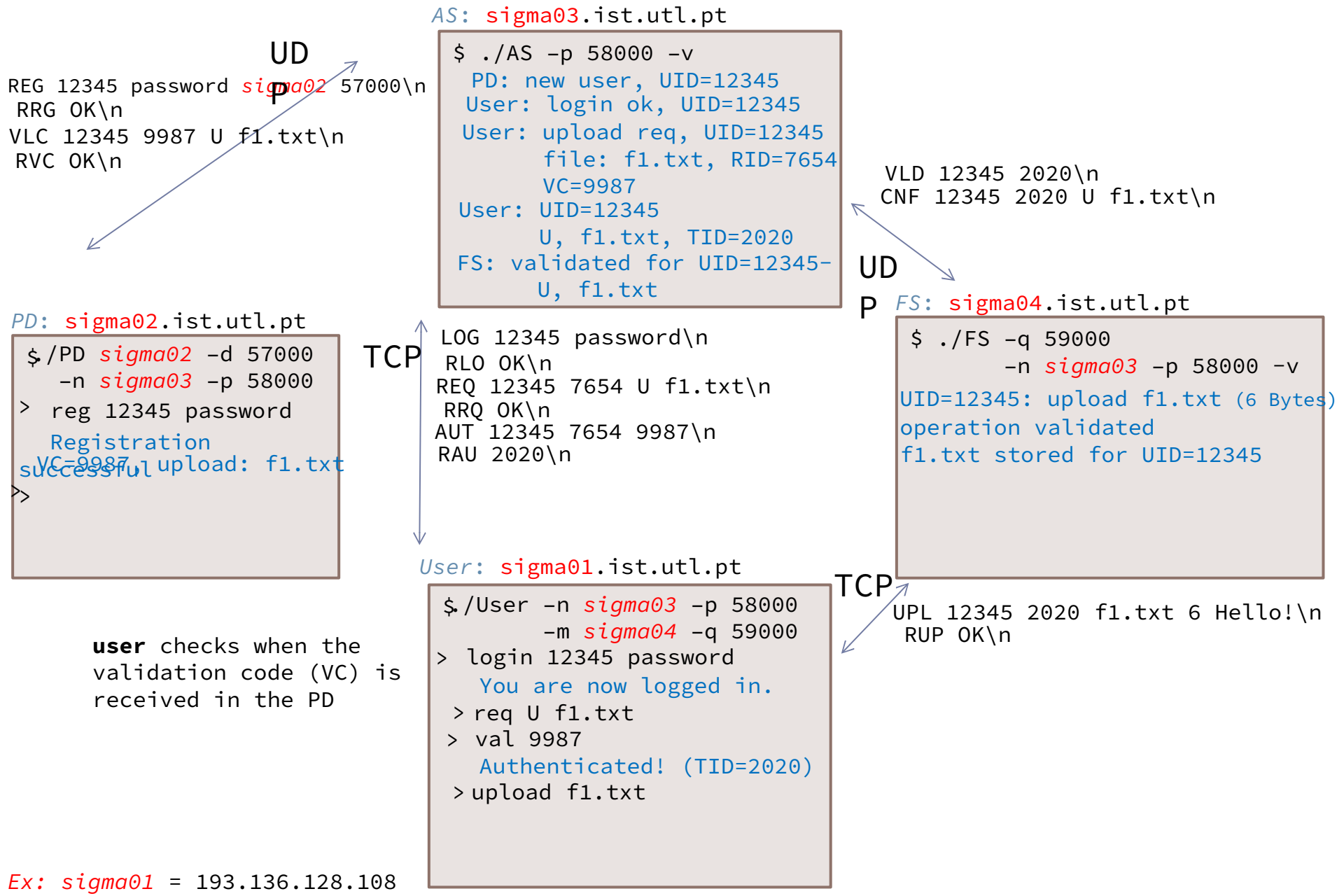


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

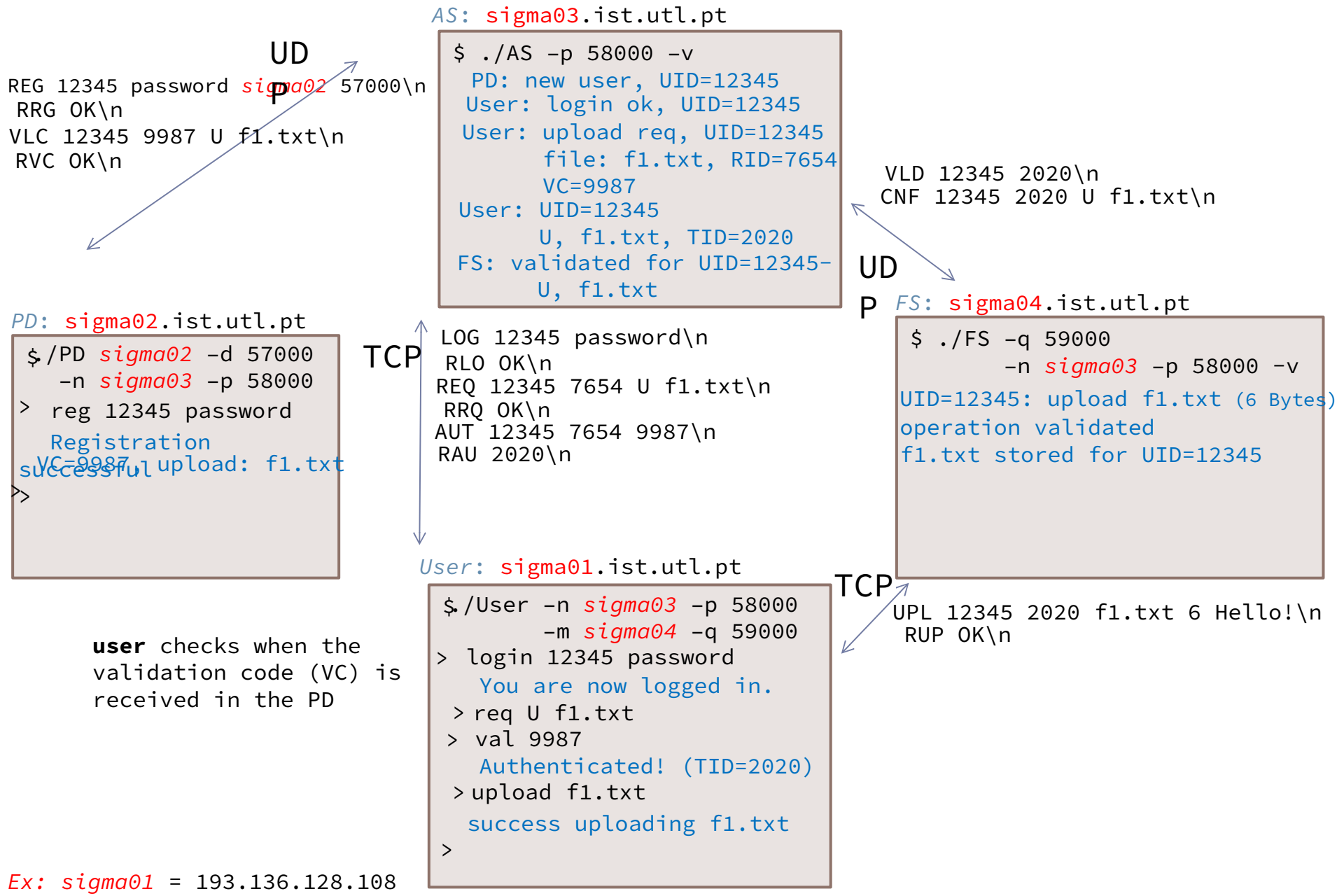
4 componentes:

- *PD*, *User*, *AS*, FS



4 componentes:

- *PD*, *User*, *AS*, FS

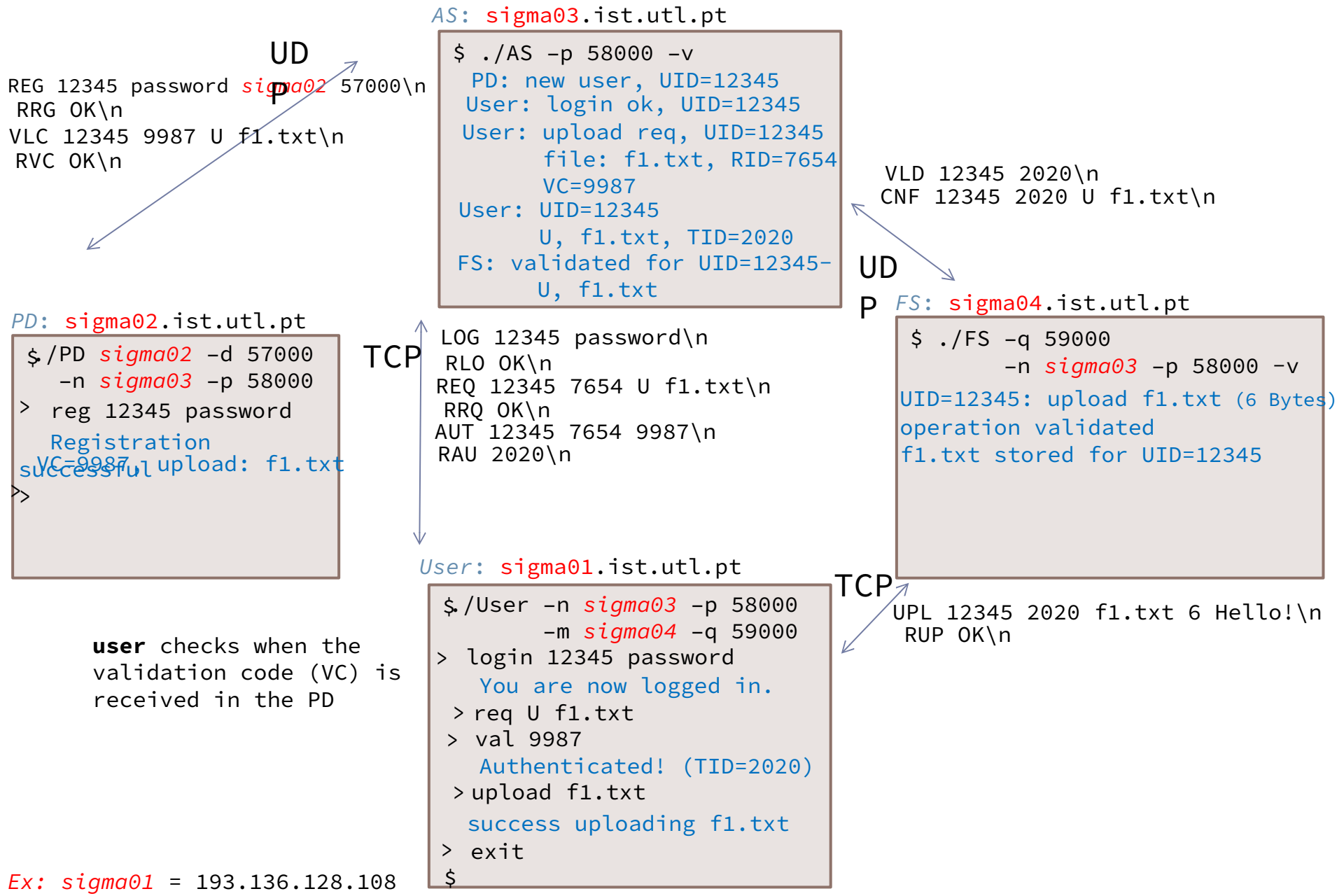


user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

4 componentes:

- *PD*, *User*, *AS*, FS



user checks when the validation code (VC) is received in the PD

Ex: sigma01 = 193.136.128.108

PD command	PD – AS (UDP)
reg UID pass	REG→ ←RRG
exit	UNR→ ←RUN

User command	User – AS (TCP)	AS – PD (UDP)	User – FS (TCP)	AS – FS (UDP)
login UID pass	LOG→ ←RLO			
req Fop [Fname]	REQ→ ←RRQ	VLC→ ←RVC		
val VC	AUT→ ←RAU			
list *l*			LST→ ←RLS	VLD→ ←CNF
retrieve filename *r filename*			RTV→ ←RRT	VLD→ ←CNF
upload filename *u filename*			UPL→ ←RUP	VLD→ ←CNF
delete filename *d filename*			DEL→ ←RDL	VLD→ ←CNF
remove *x*			REM→ ←RRM	VLD→ ←CNF
exit	(close TCP connection)			

RC two factor authentication

O código desenvolvido em C ou C++ deve funcionar no cluster ***sigma*** e estar convenientemente **estruturado** e **comentado**.

As chamadas de sistema ***read()*** e ***write()*** podem ler e escrever, respetivamente, um numero de bytes inferior ao que lhes foi solicitado – deve garantir que ainda assim a sua implementação funciona corretamente.

Os processos (clientes e servidores) devem terminar graciosamente pelo menos nas seguintes situações de falha:

- **mensagens do protocolo erradas** vindas da entidade par correspondente;
- condições de **erro das chamadas de sistema**

O código a entregar: ficheiros fonte dos programas (*User*, *PD*, *AS*, *FS)*, *Makefile, e ficheiros auxiliares*.

Entrega por e-mail, **até dia 13 de Novembro de 2020, às 23h59mn**.
Deve criar um único ficheiro de arquivo **zip** com todos os ficheiros fonte e outros ficheiros necessários à execução das aplicações. O arquivo deve estar preparado para ser aberto para o diretório corrente e compilado com o comando make. O nome do ficheiro submetido deve ter o seguinte formato: **proj<número do**