Möglichkeiten der Nebenläufigkeit in der Programmiersprache Perl

Oder auch:

Node.JS in Perl

06.03.2019
German Perl Workshop 2019

Markus Schräder
CryptoMagic GmbH

Synchron

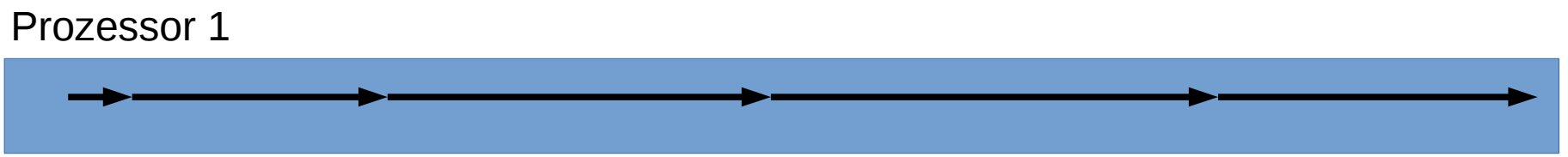
CRYPTO MAGIC einfach. sicher. verbunden.

Synchron

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

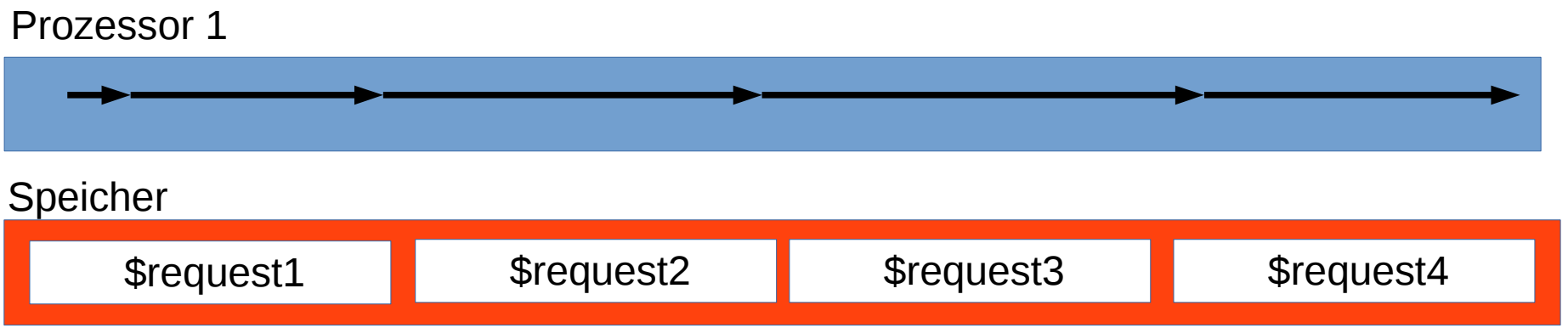
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron



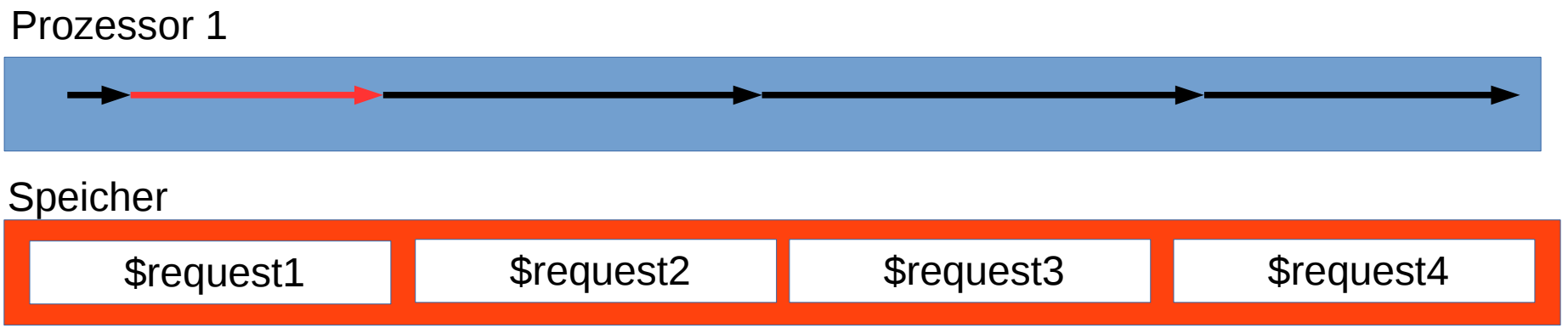

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

Synchron


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

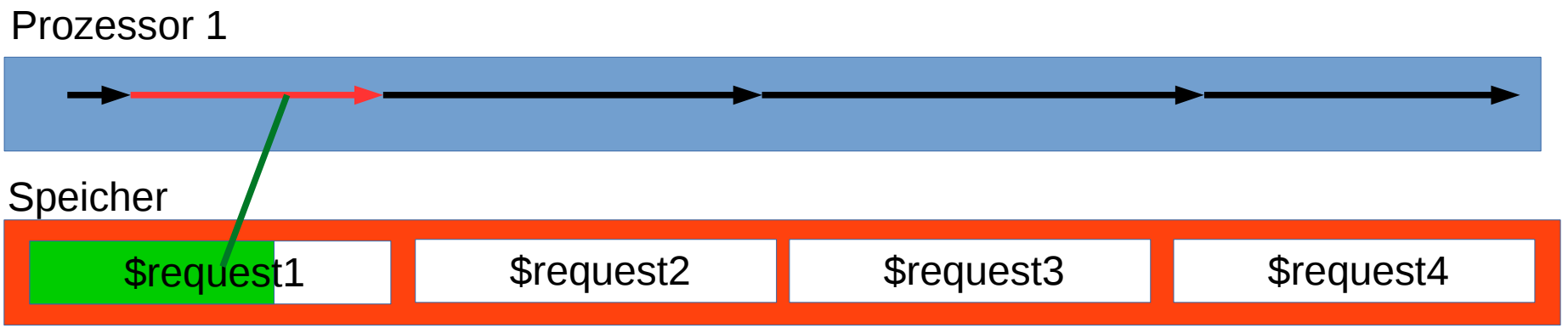
CRYPTO MAGIC einfach. sicher. verbunden.

Synchron


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

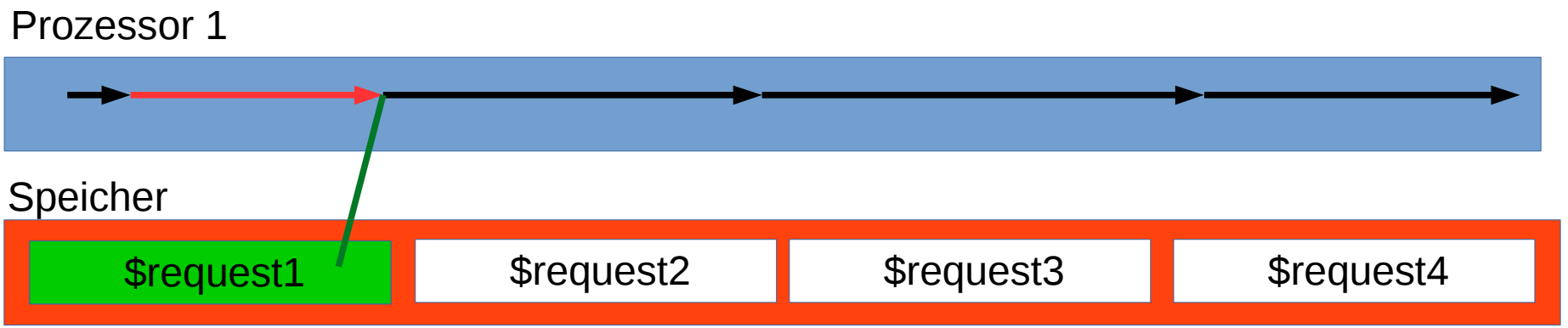
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

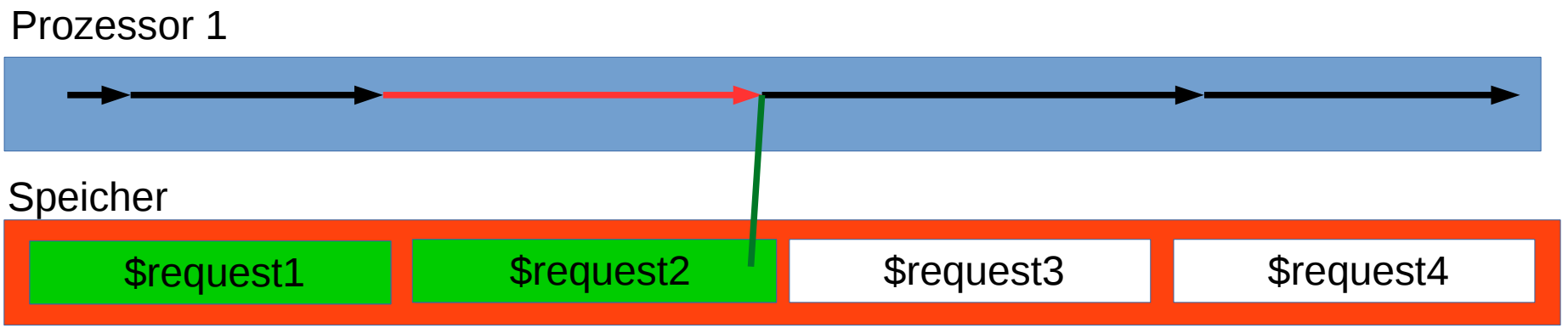
CRYPTO MAGIC einfach. sicher. verbunden.

Synchron


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

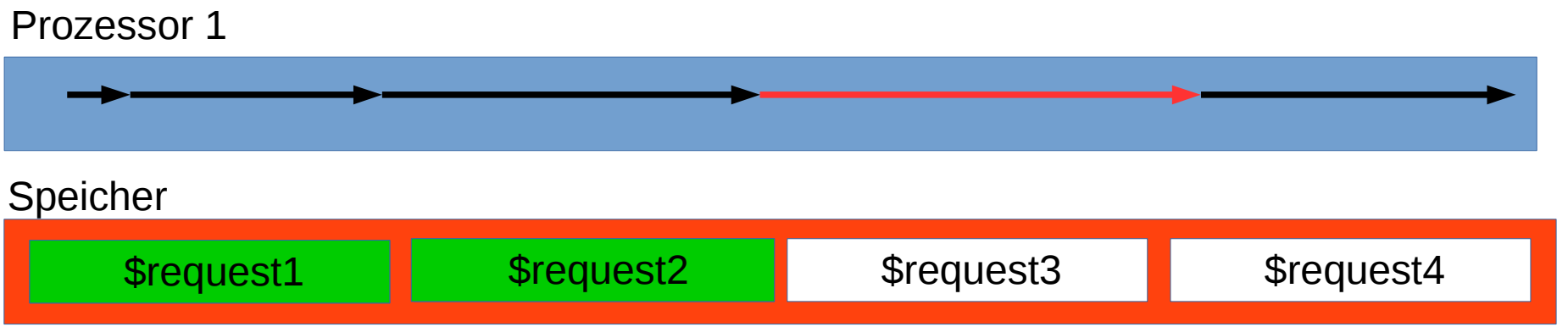
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

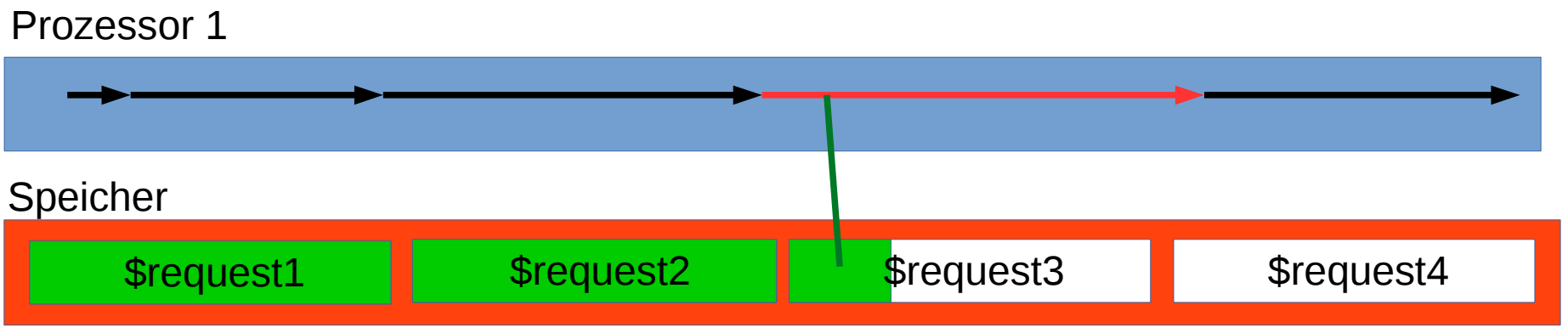
CRYPTO MAGIC einfach. sicher. verbunden.

Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

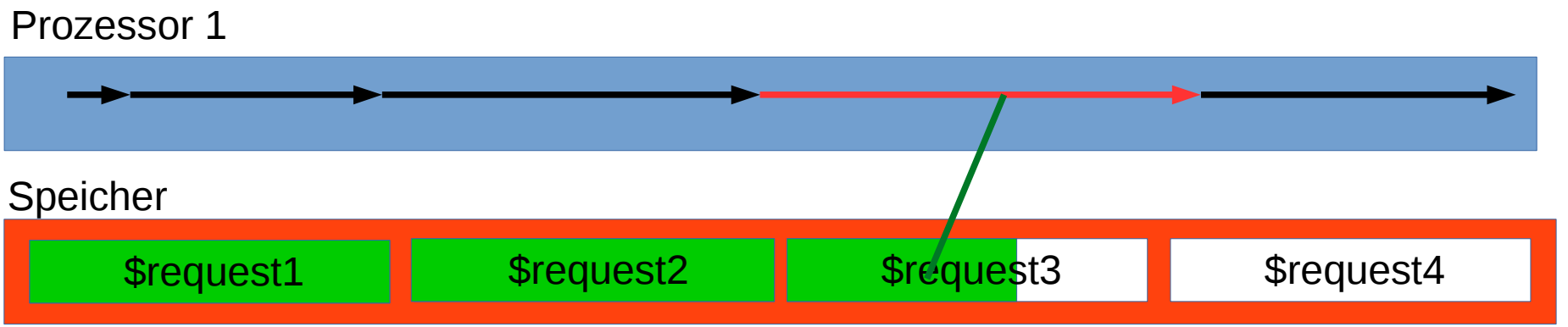
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

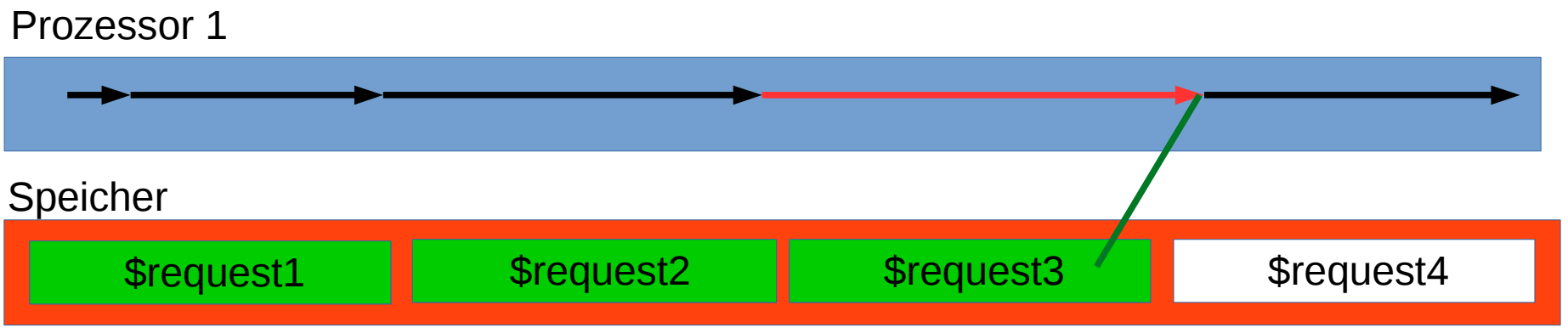
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

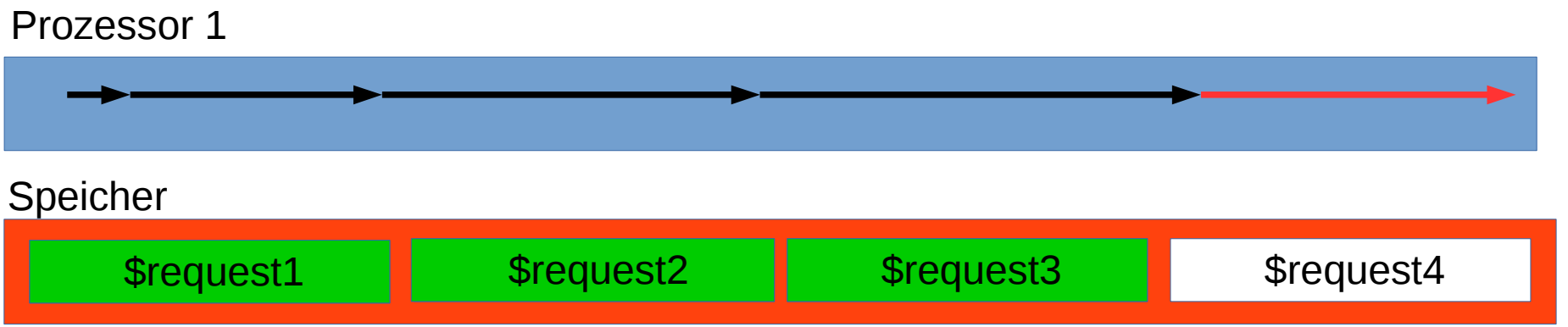
CRYPTO MAGIC einfach. sicher. verbunden.

Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

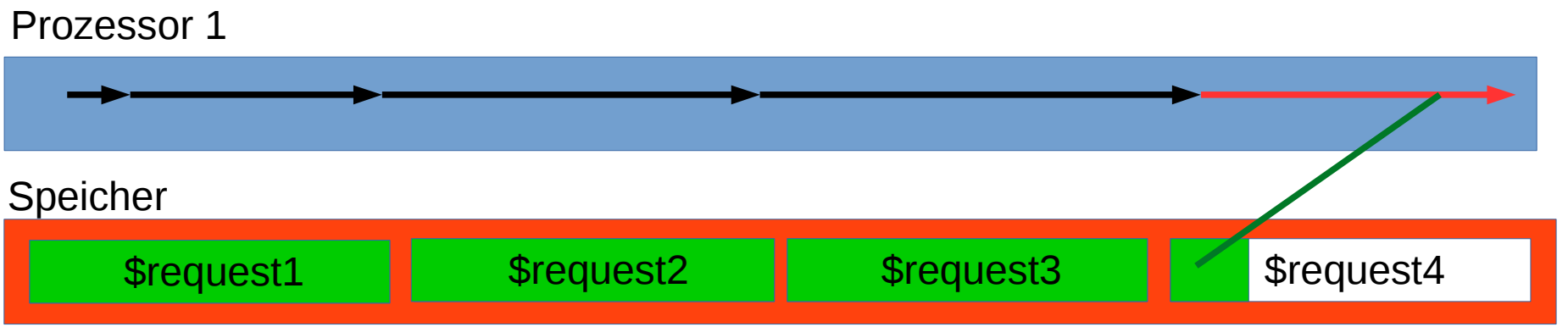
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

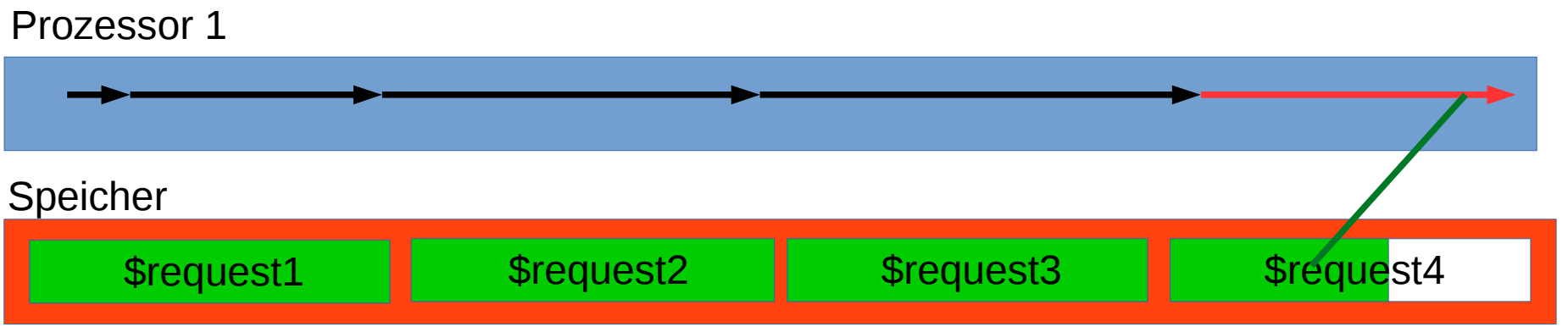
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

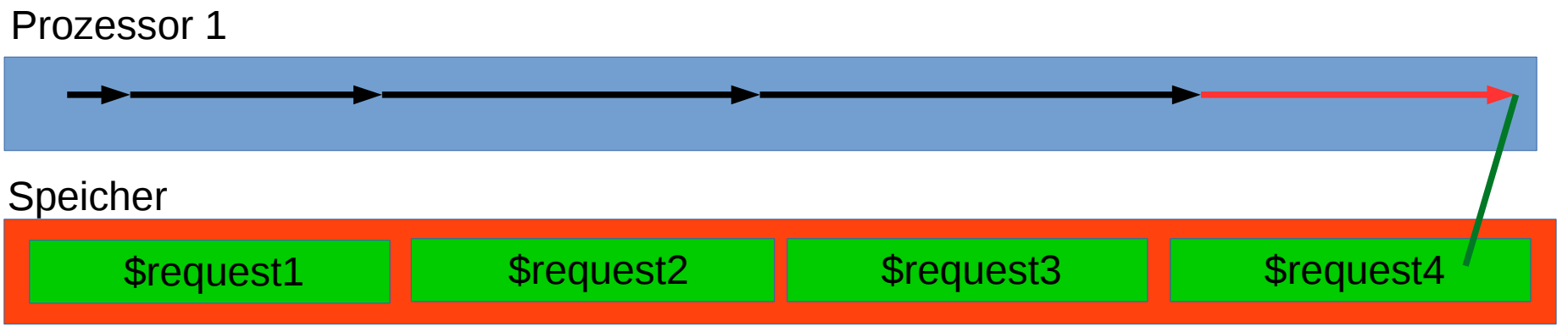
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

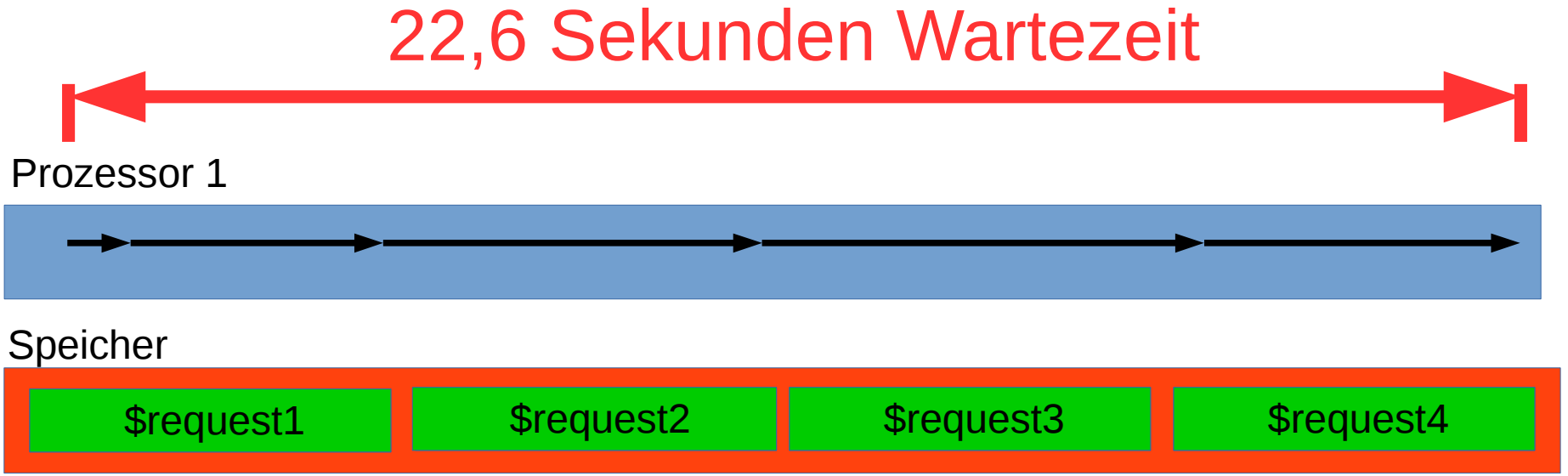
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

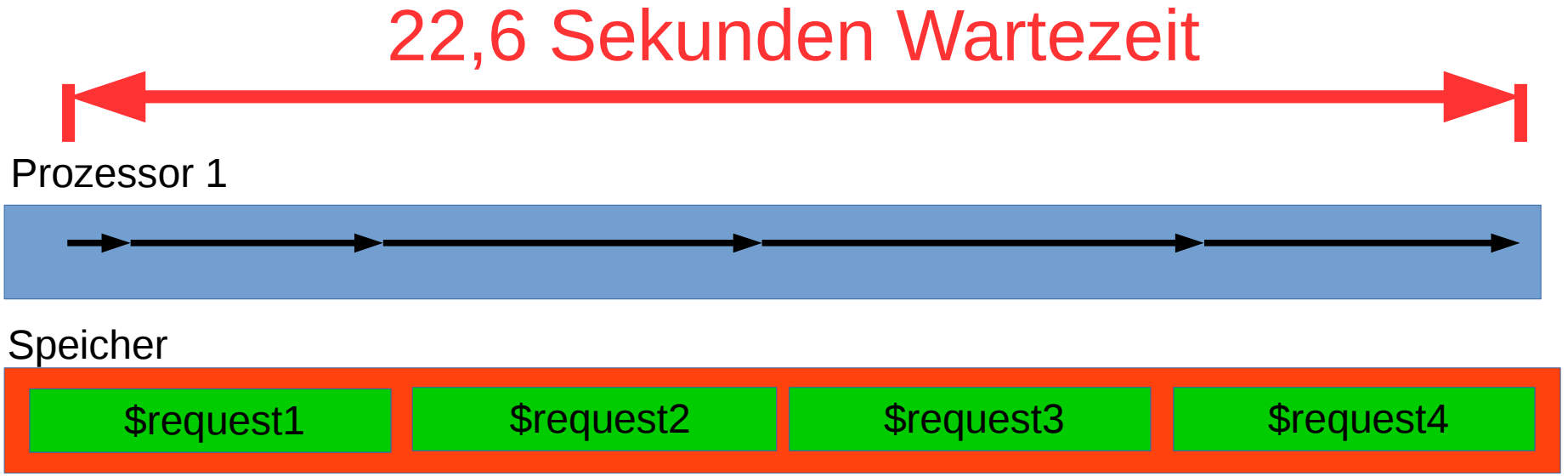
Synchron




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

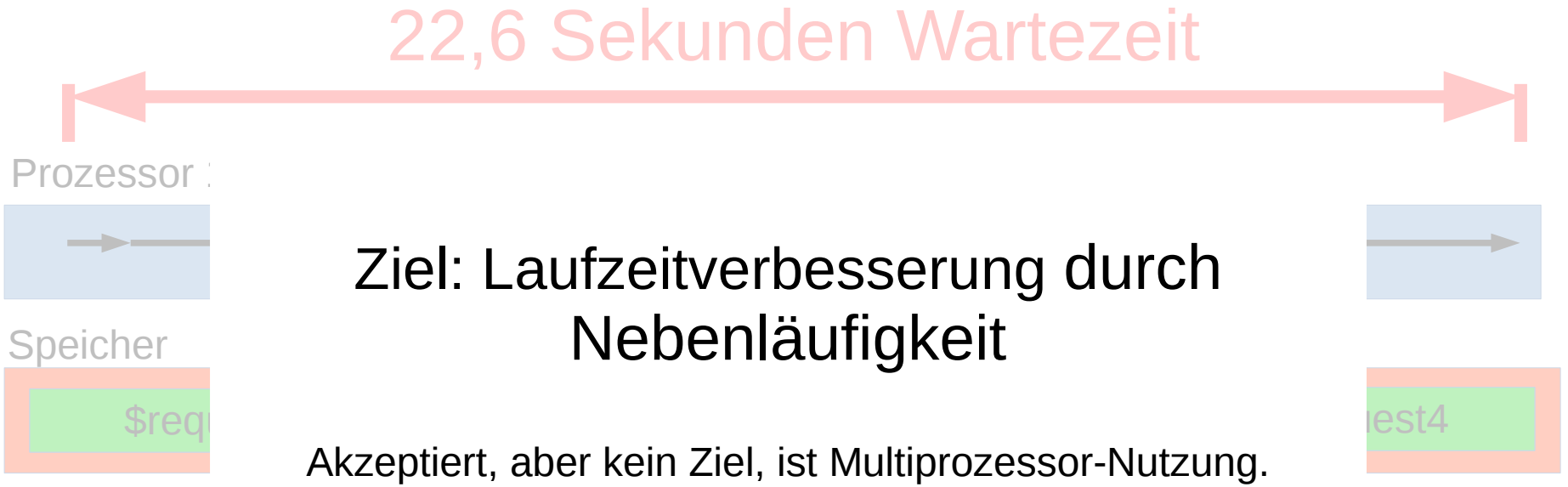
Synchron?




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron?



Ziel: Laufzeitverbesserung durch Nebenläufigkeit

Akzeptiert, aber kein Ziel, ist Multiprozessor-Nutzung.

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

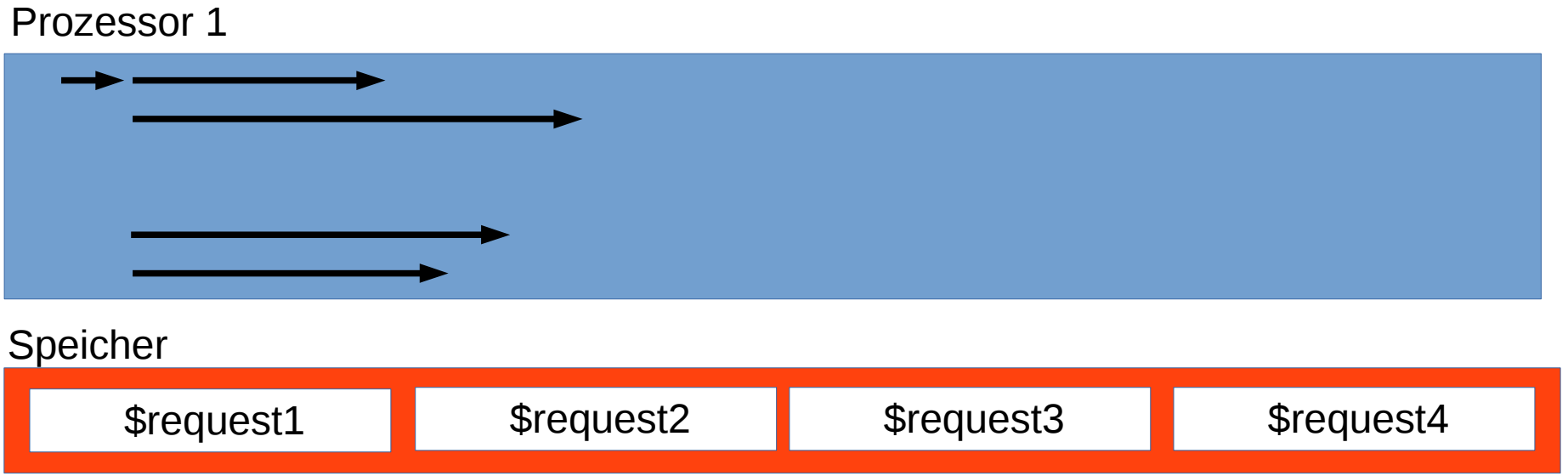
Multithreading

Prozessor 1

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

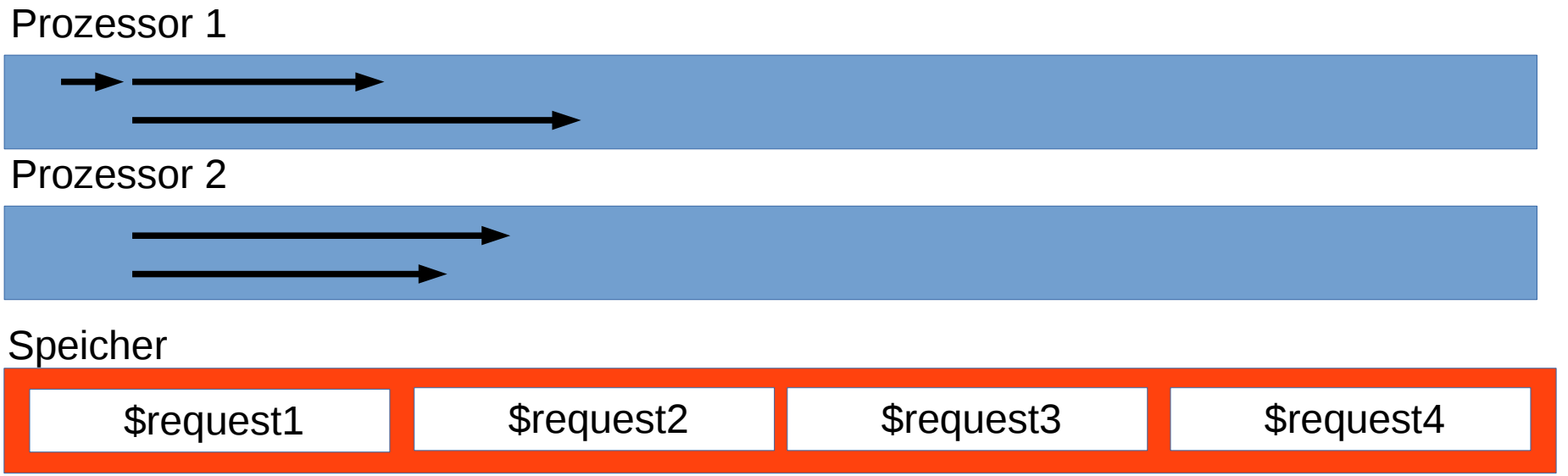
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

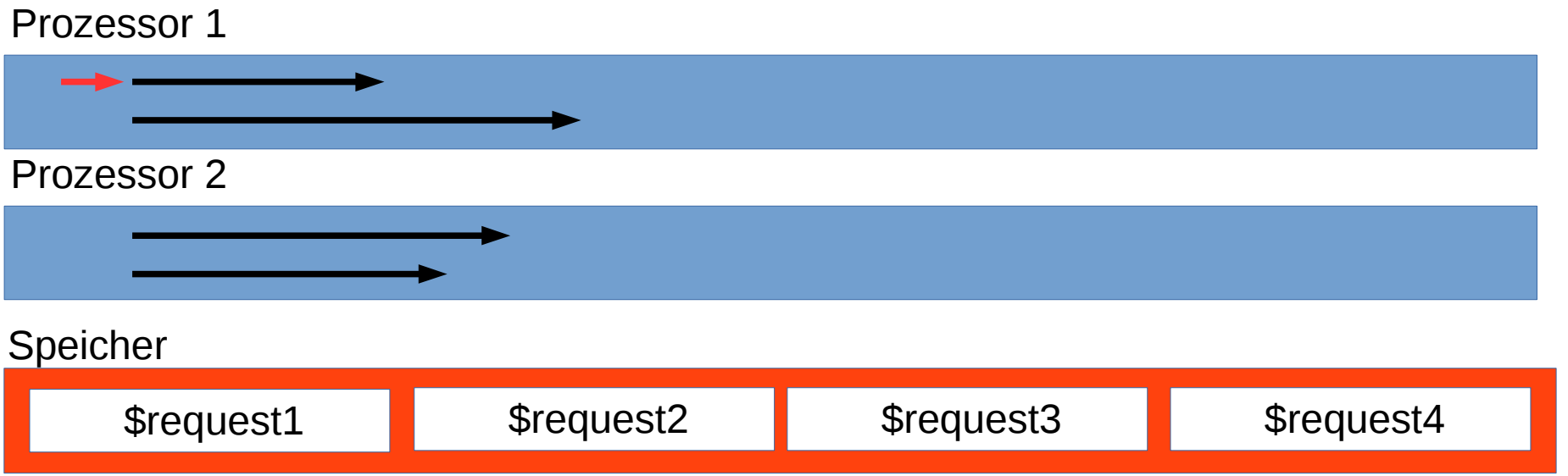
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

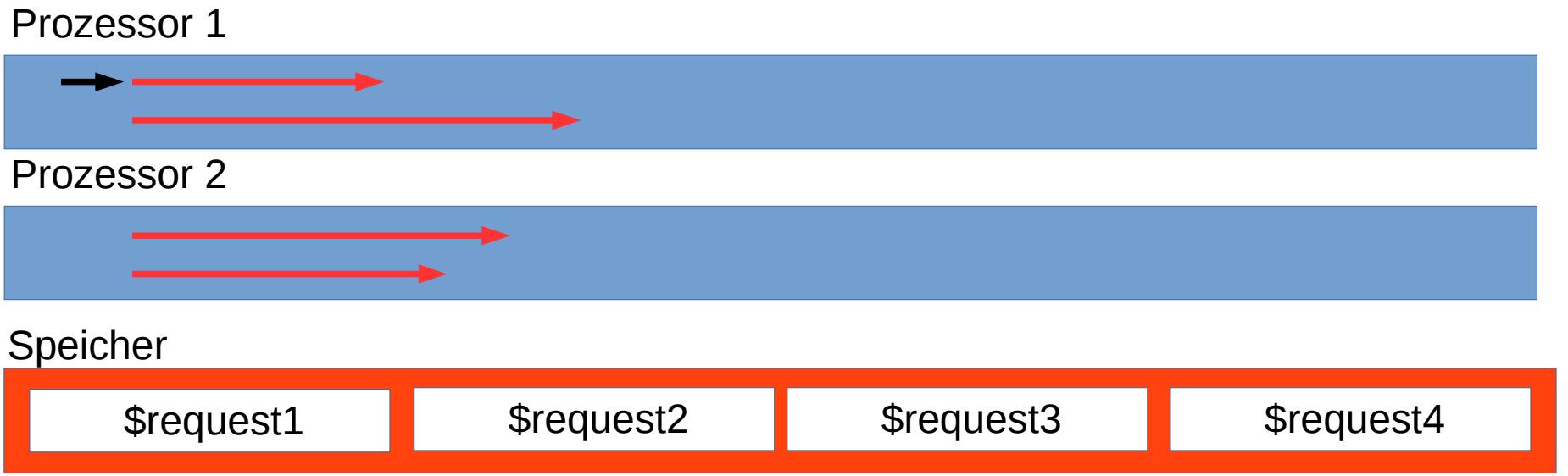
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

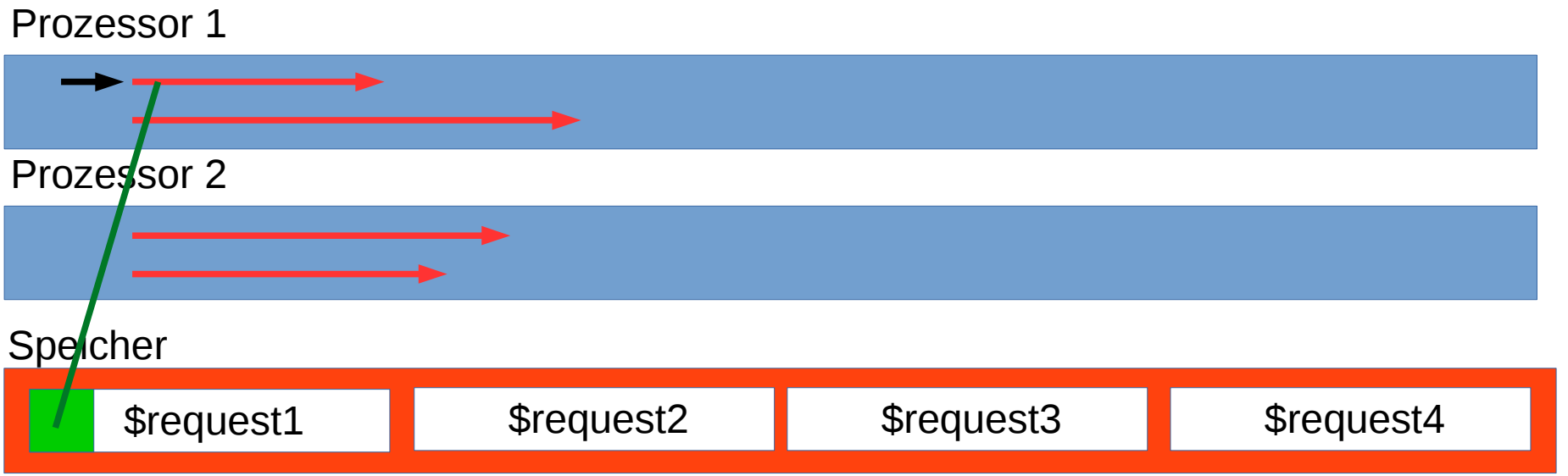
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

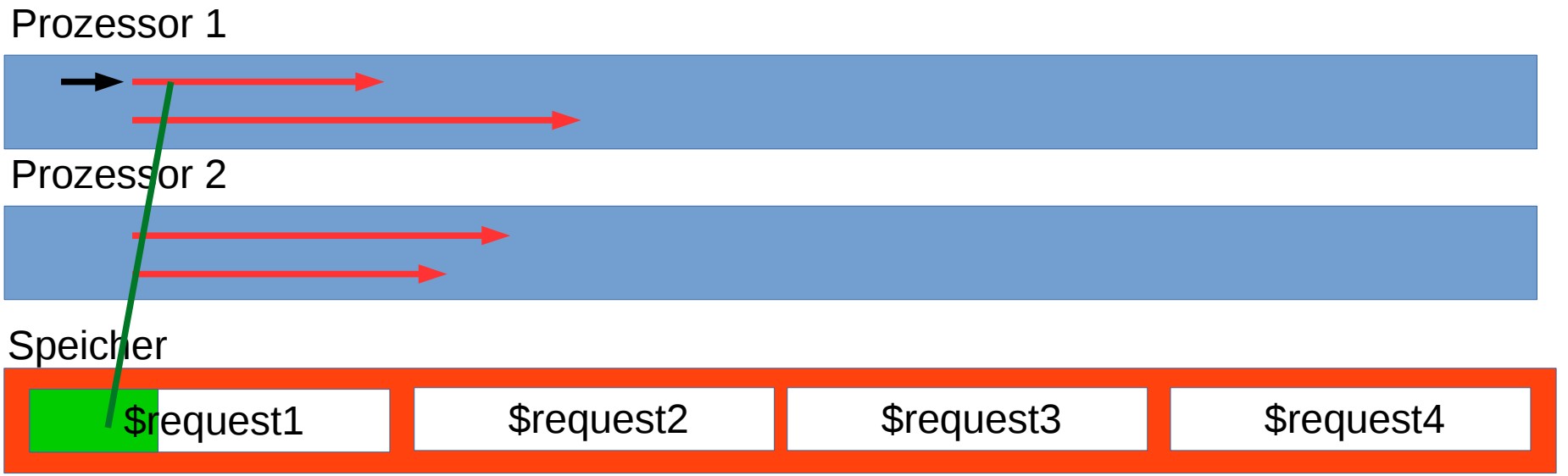
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

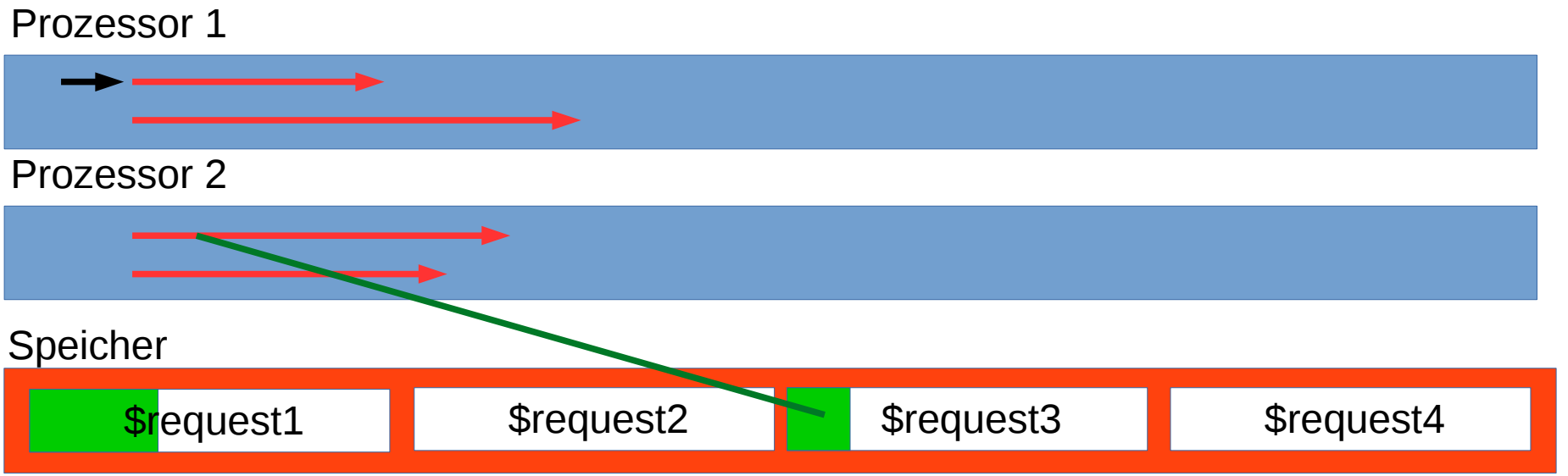
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

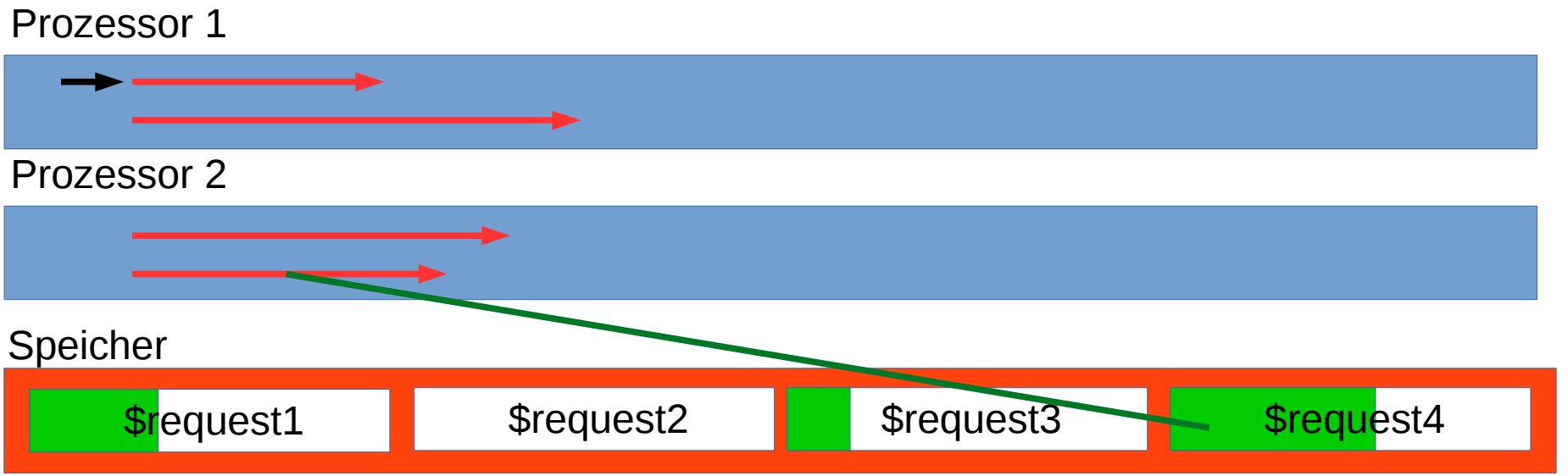
Multithreading




```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.

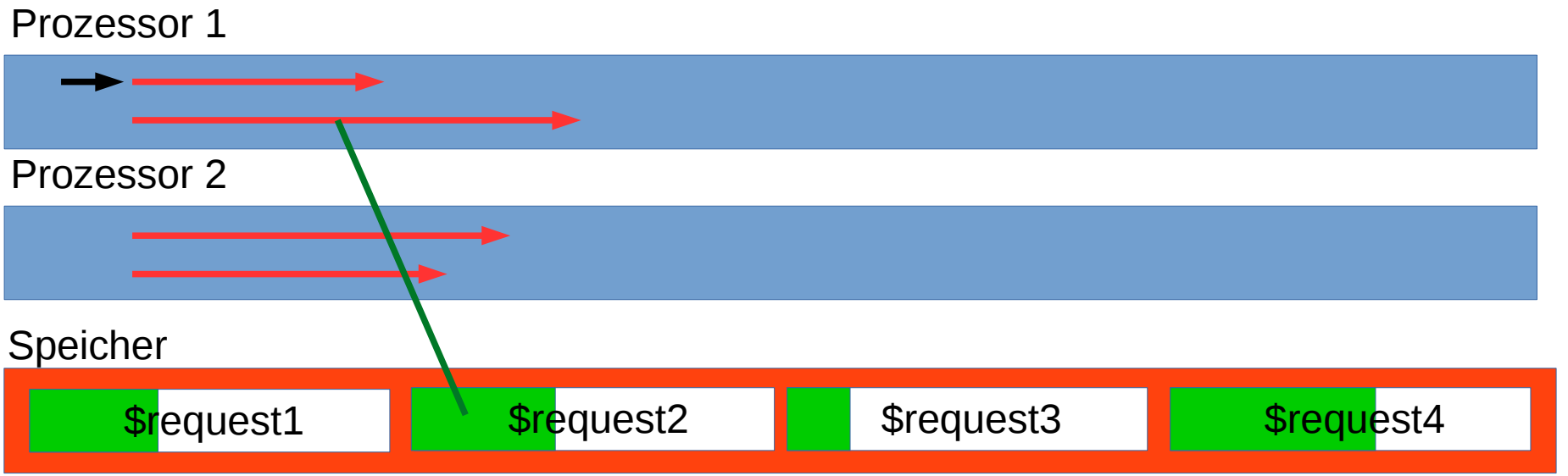
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

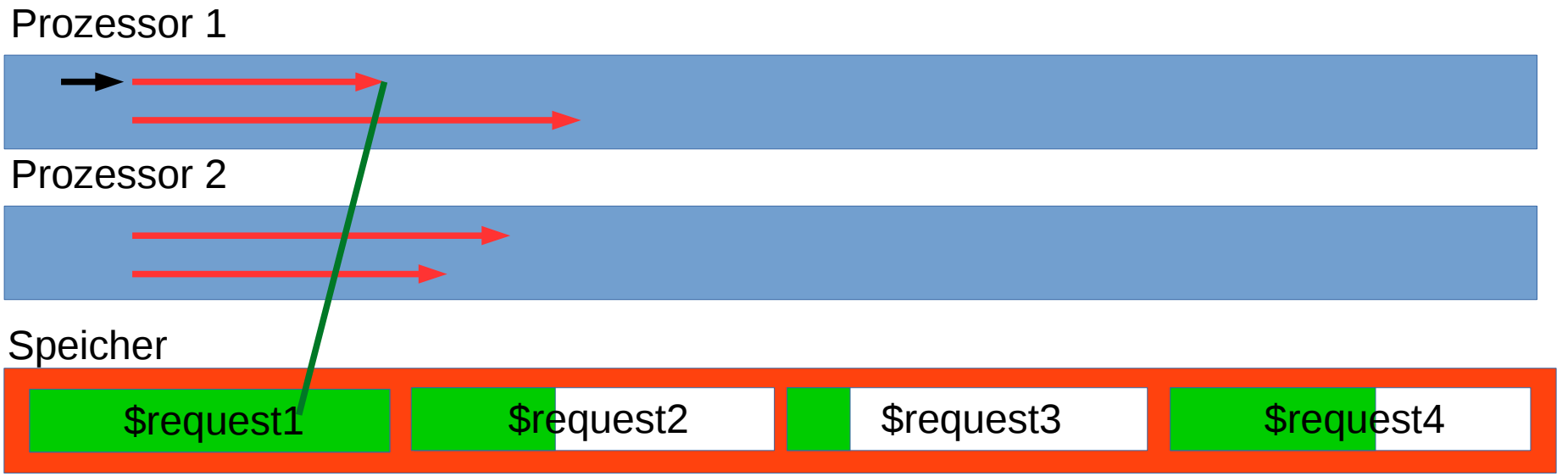
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

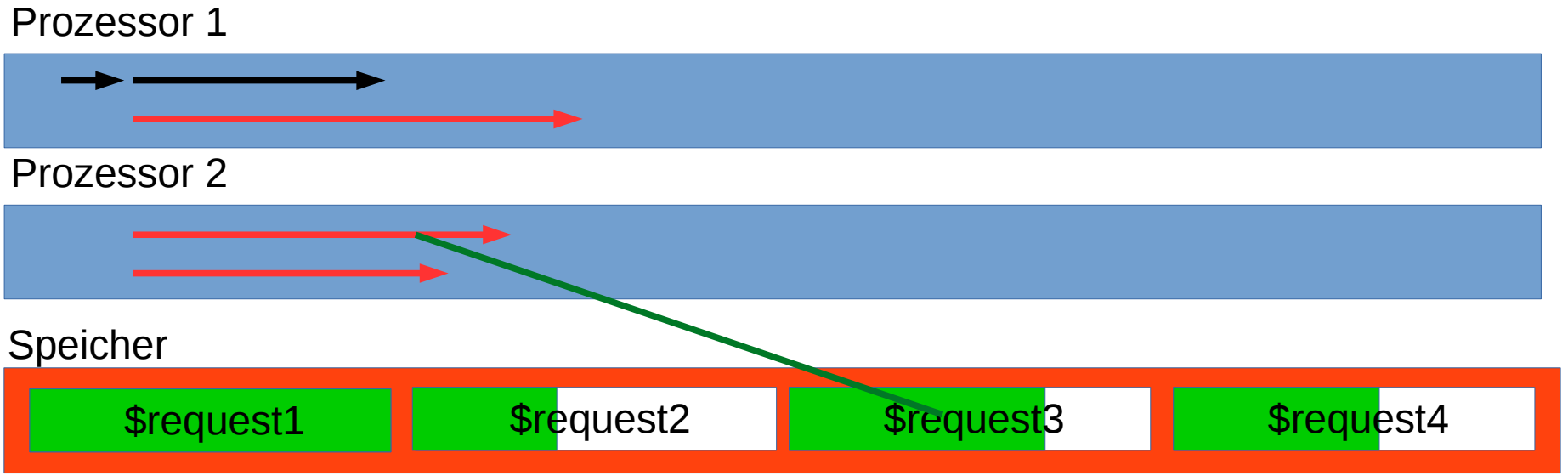
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

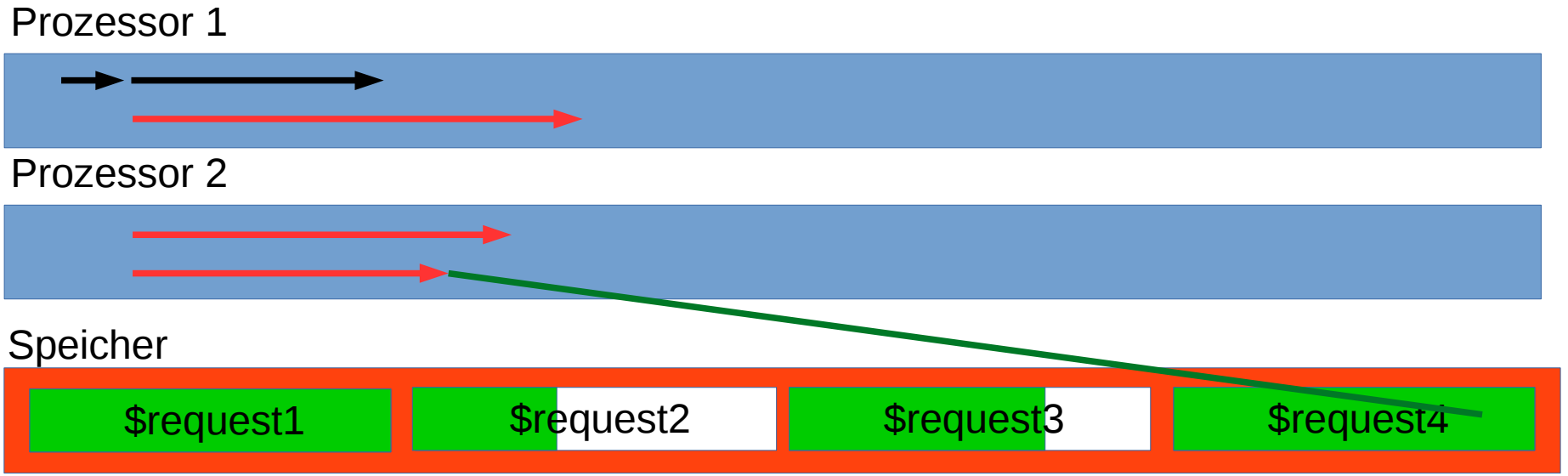
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

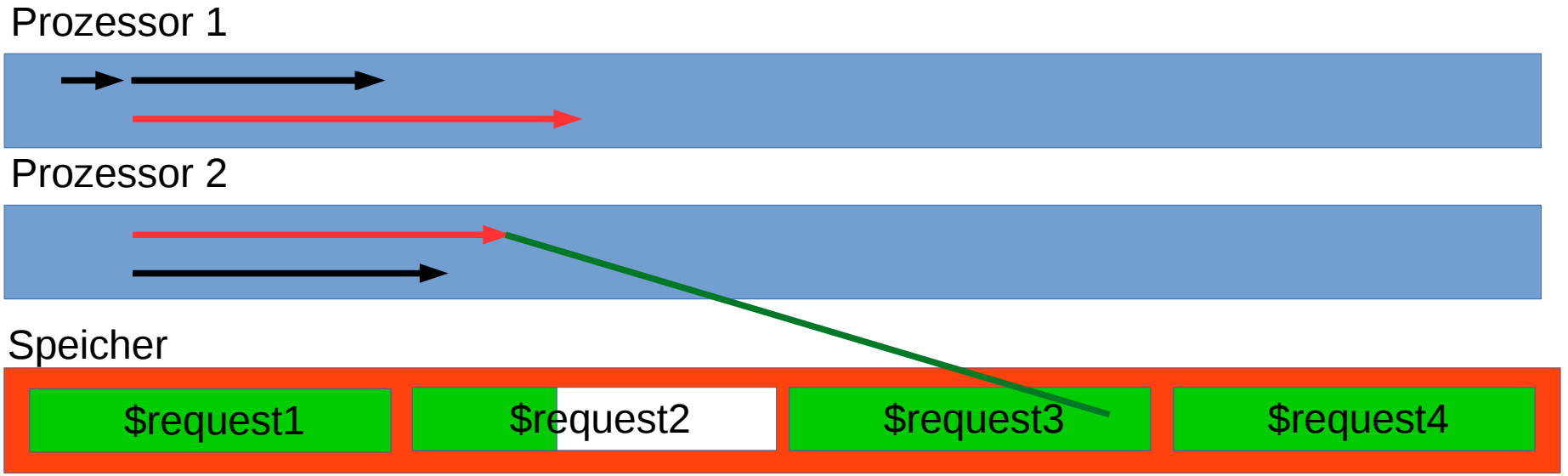
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

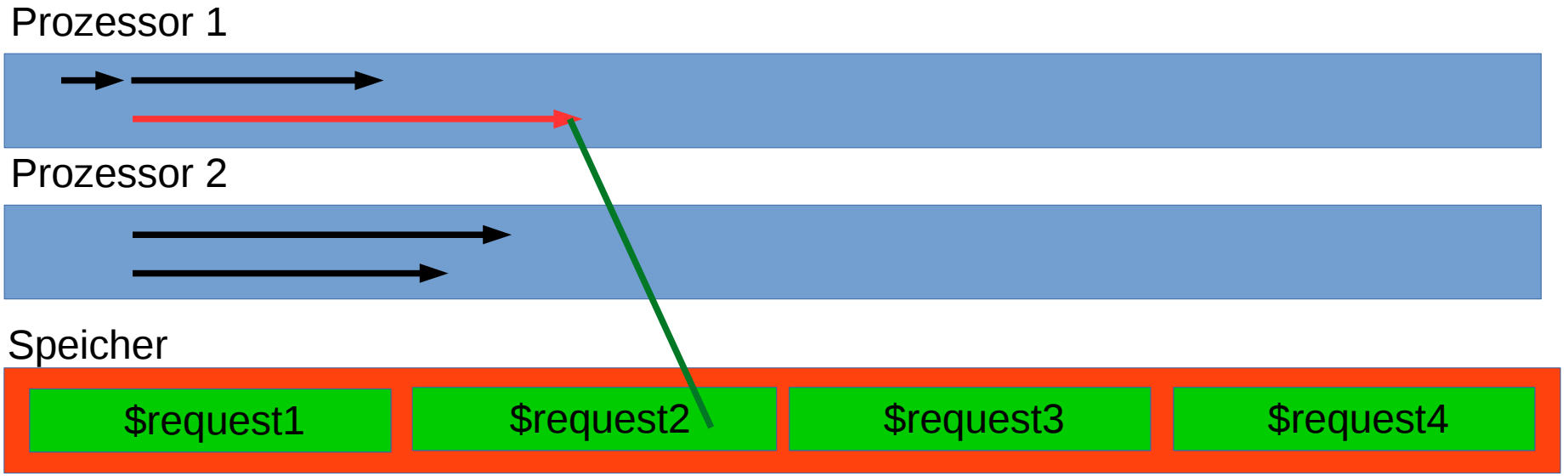
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

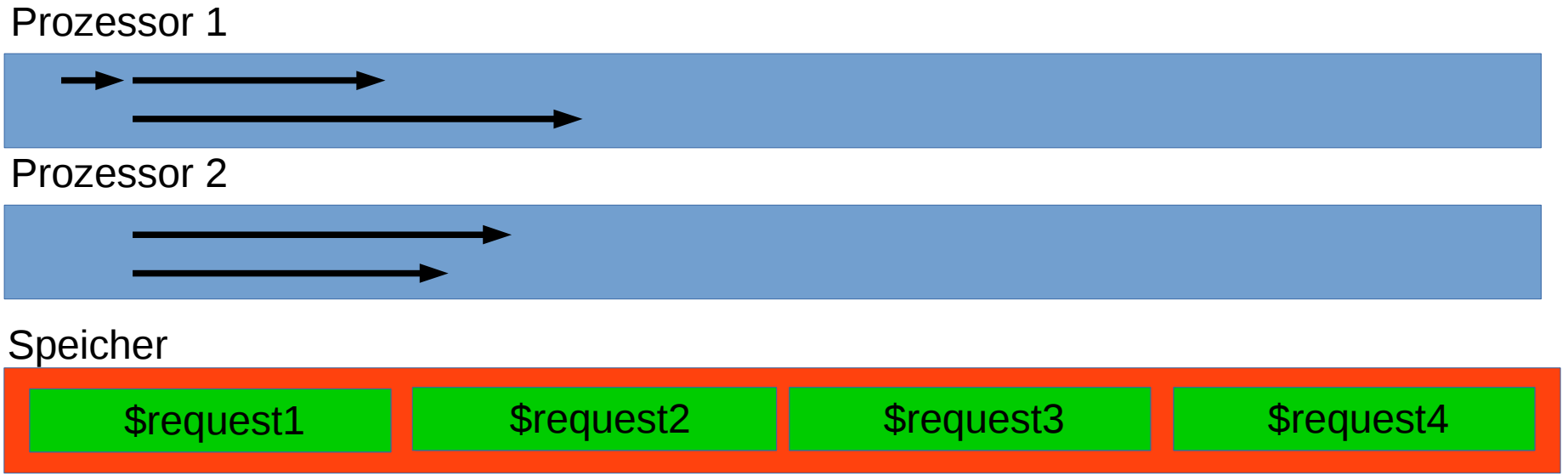
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Multithreading


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

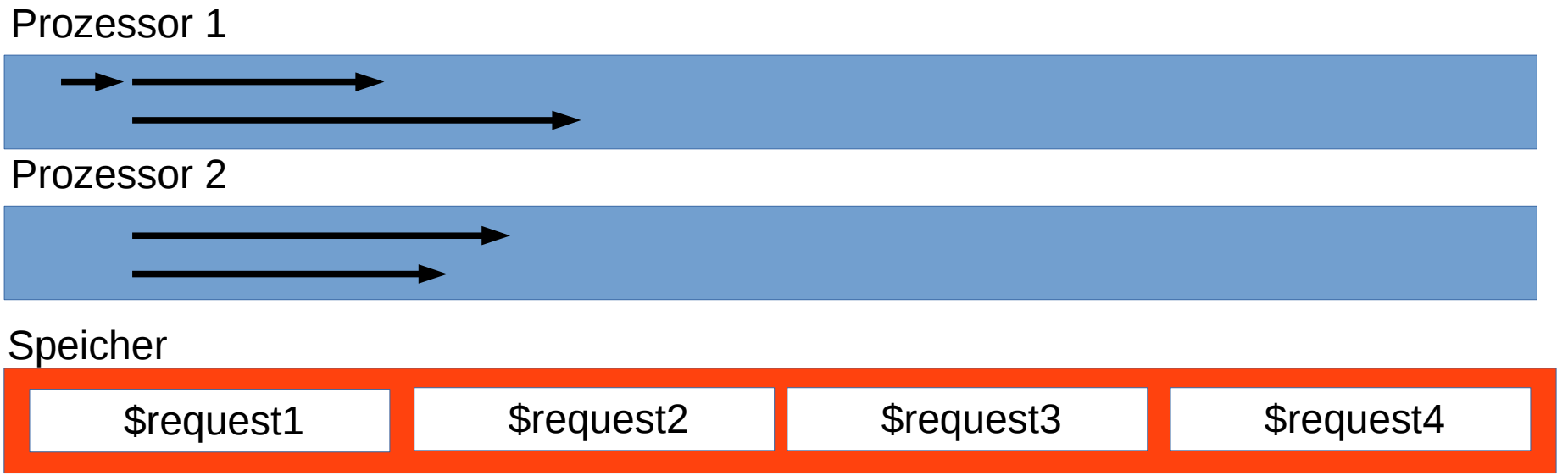
CRYPTO MAGIC einfach. sicher. verbunden.

Multithreading


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

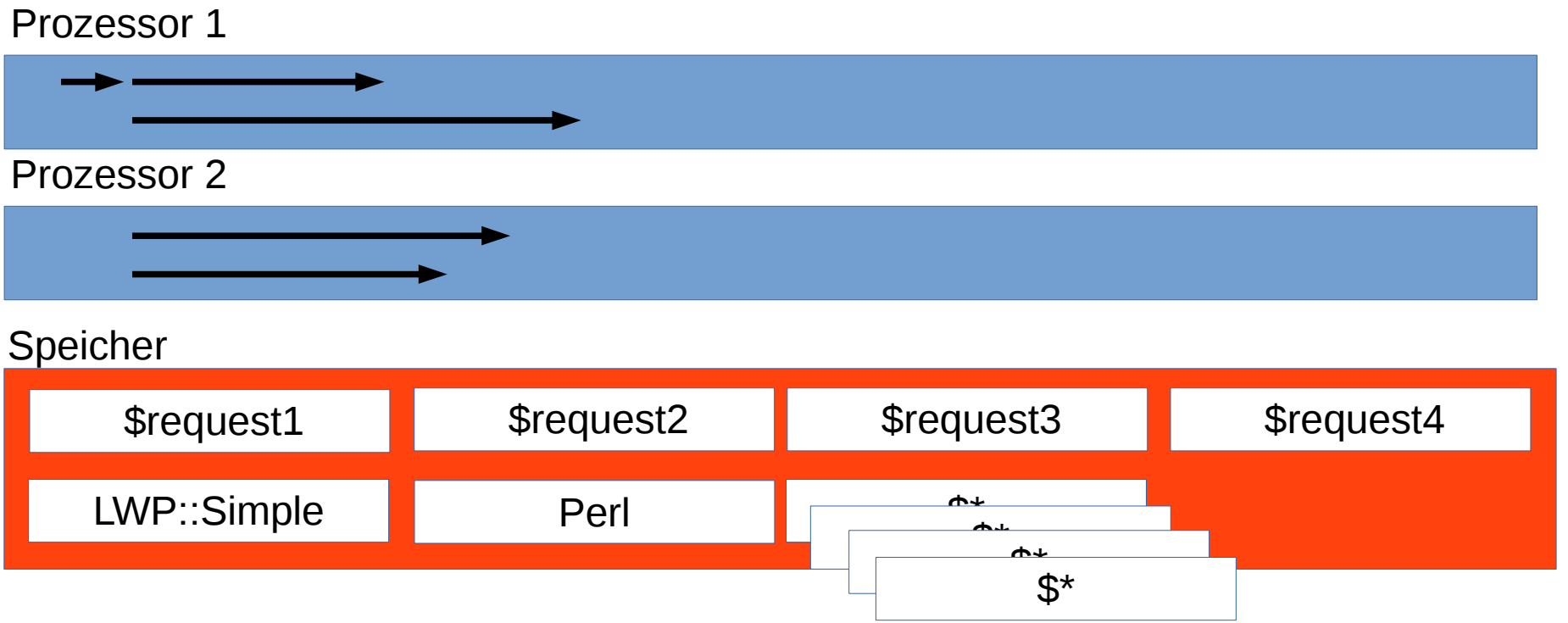
Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Multithreading




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

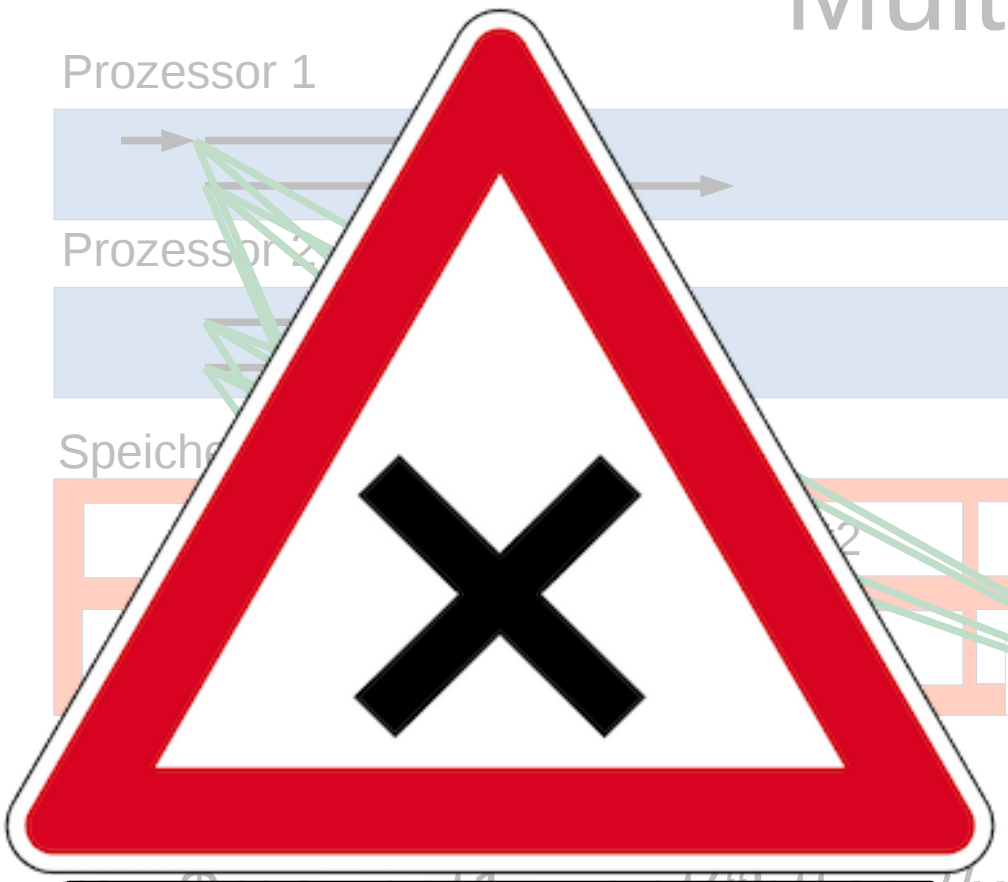
Multithreading


```
use LWP::Simpl
my $request1 = g                                    ic.eu/");
my $request1 = g                                    ;
my $request3 = g
my $request4 = g                                    l/“);
```

Parallelisierung mit

Locking

Multithreading / Fork



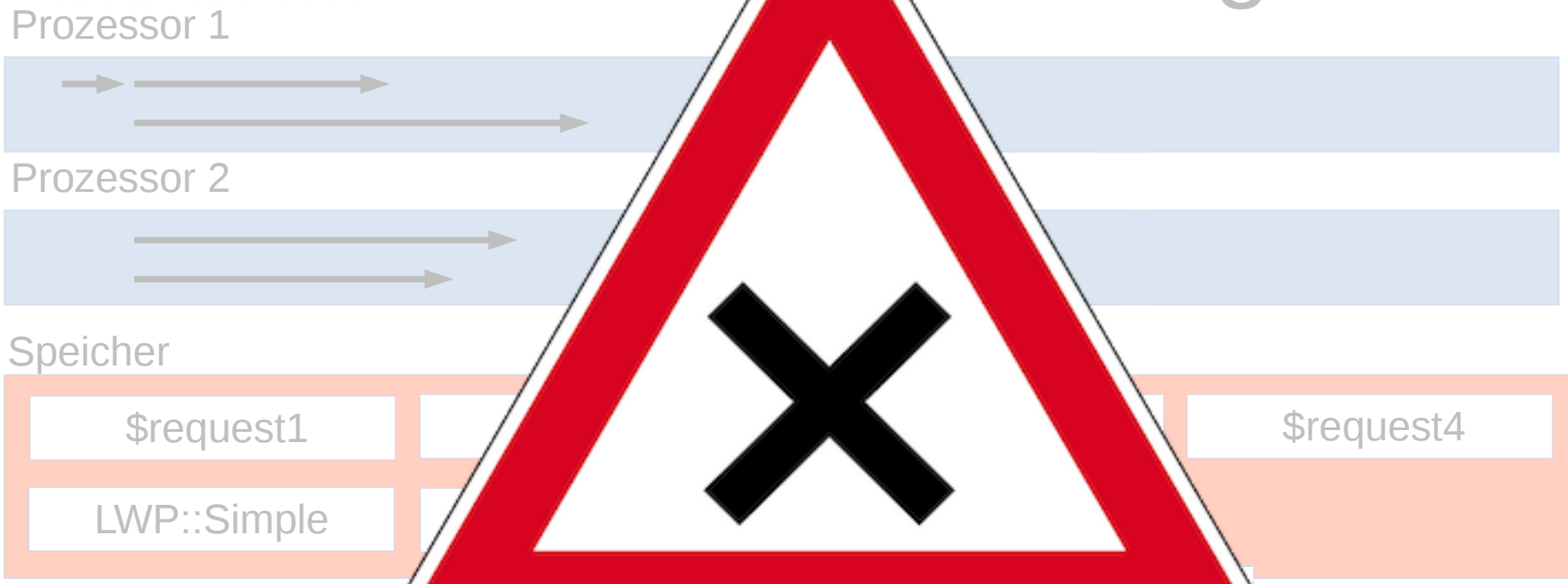
MT: Parallelisierung mit

Locking

Fork: Parallelisierung mit

Kopieren

Perl-Multithreading / Fork




```
use LWP::Sim
my $request1
my $request1
my $request3
my $request4
```

```
c.eu/");
/“);
```

Perl-MT: Parallelisierung mit

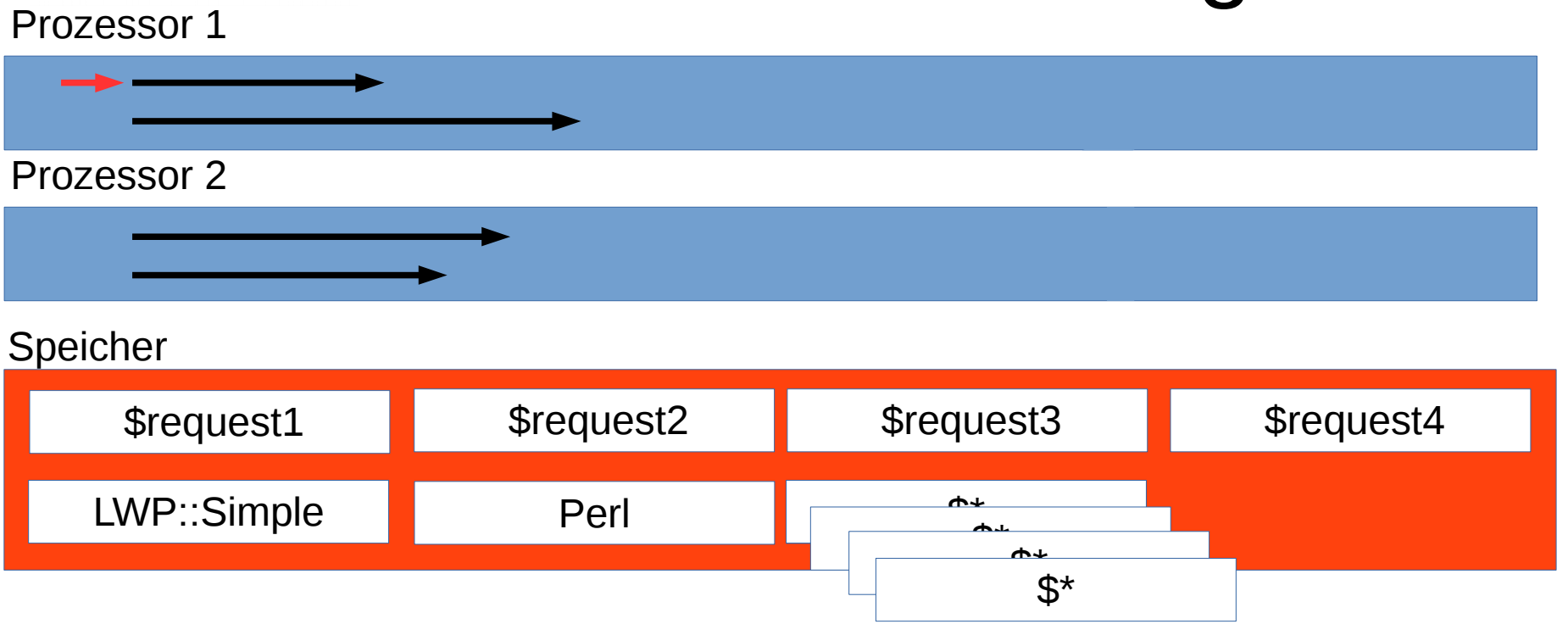
teils

Locking

und teils

Kopieren

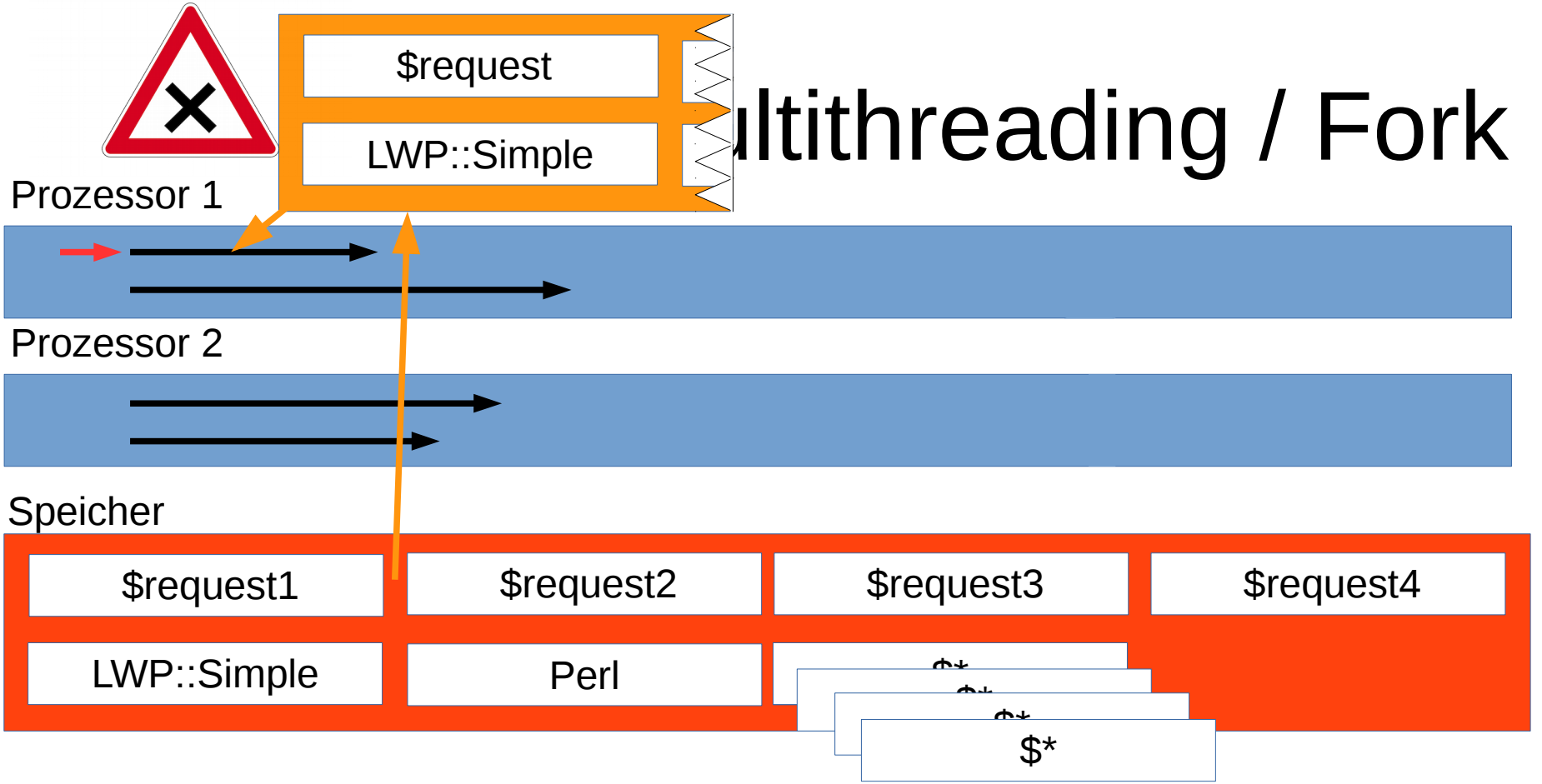
Perl-Multithreading / Fork




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

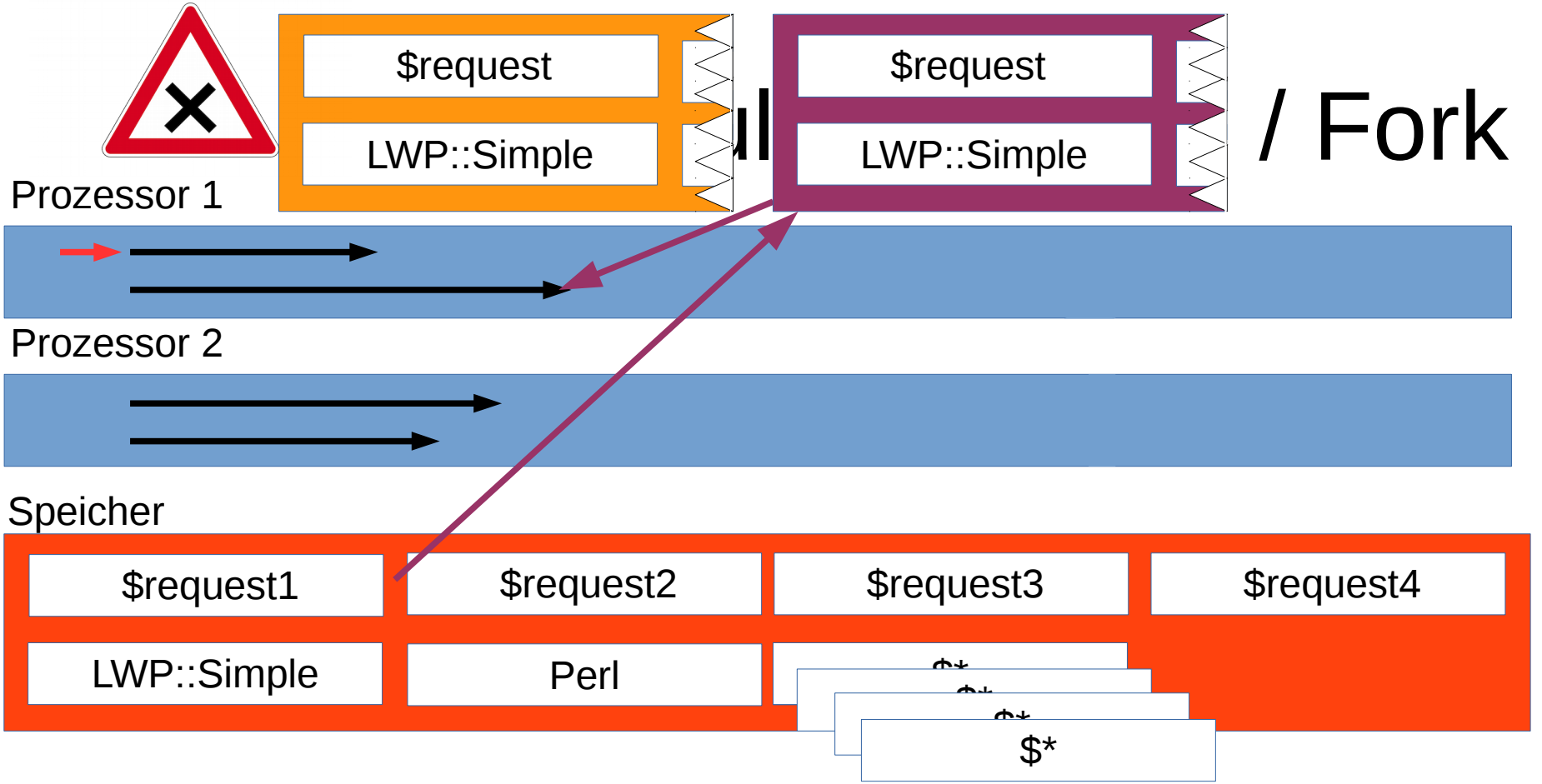
CRYPTO MAGIC einfach. sicher. verbunden.

ltithreading / Fork




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

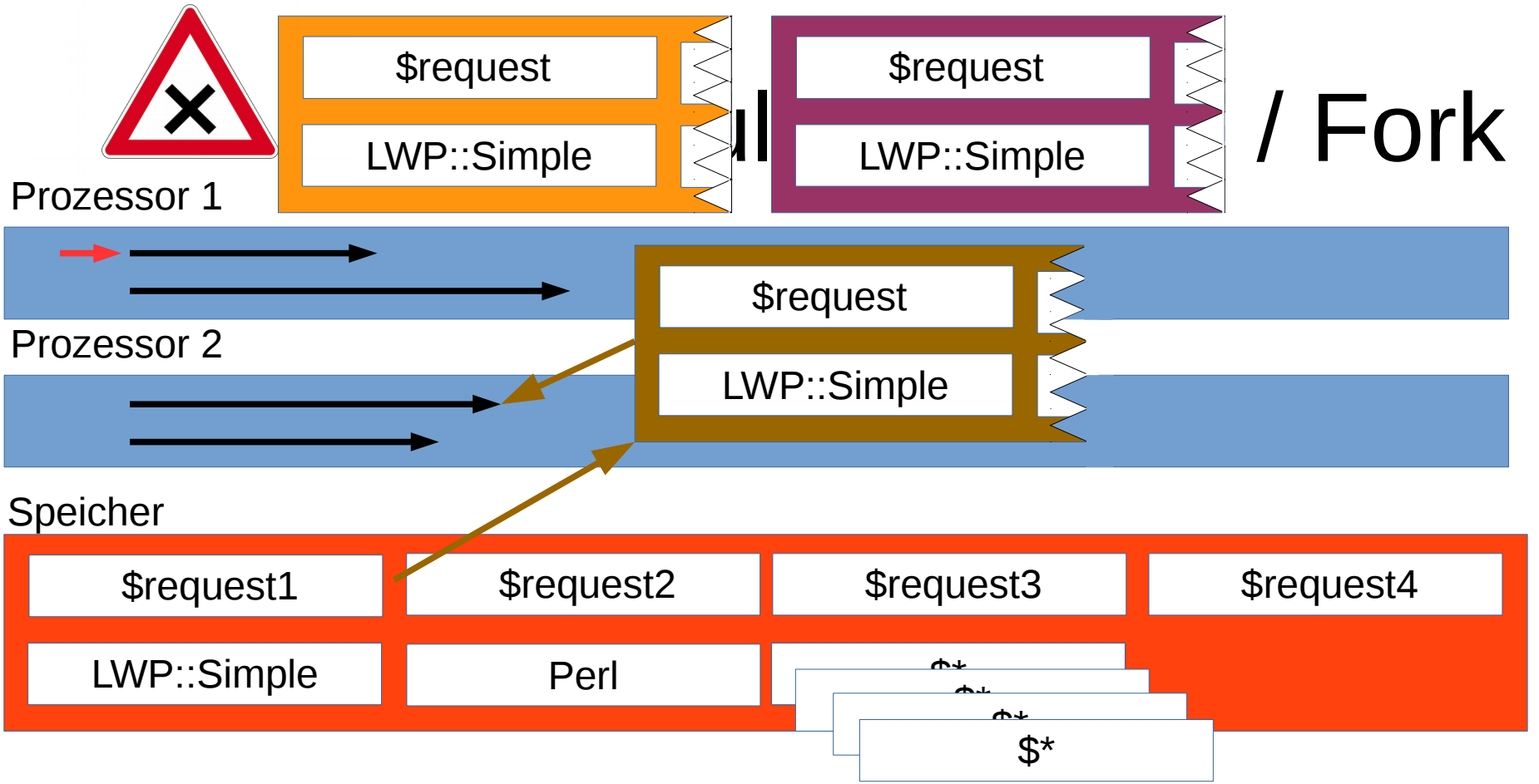
CRYPTO MAGIC einfach. sicher. verbunden.



/ Fork

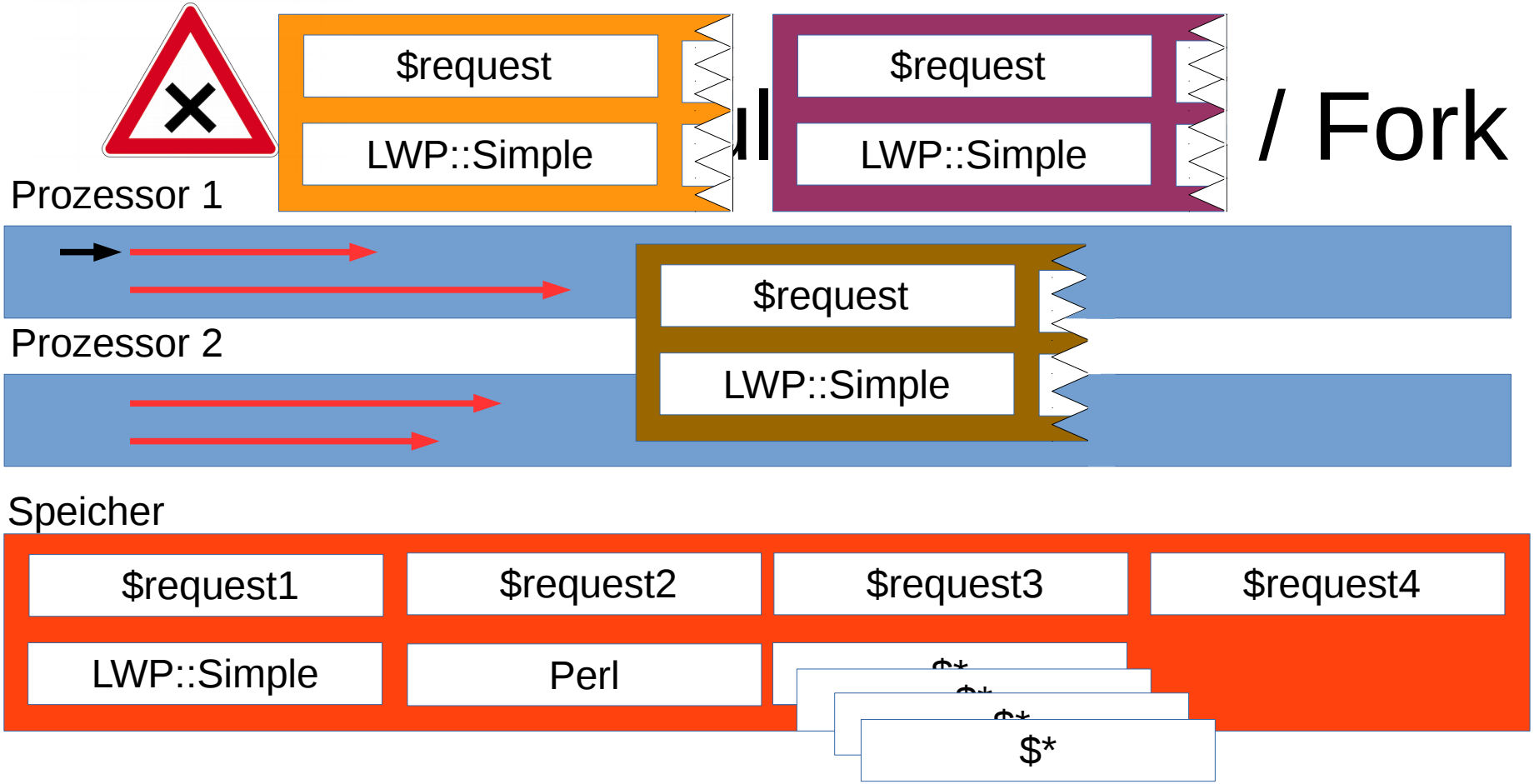
```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



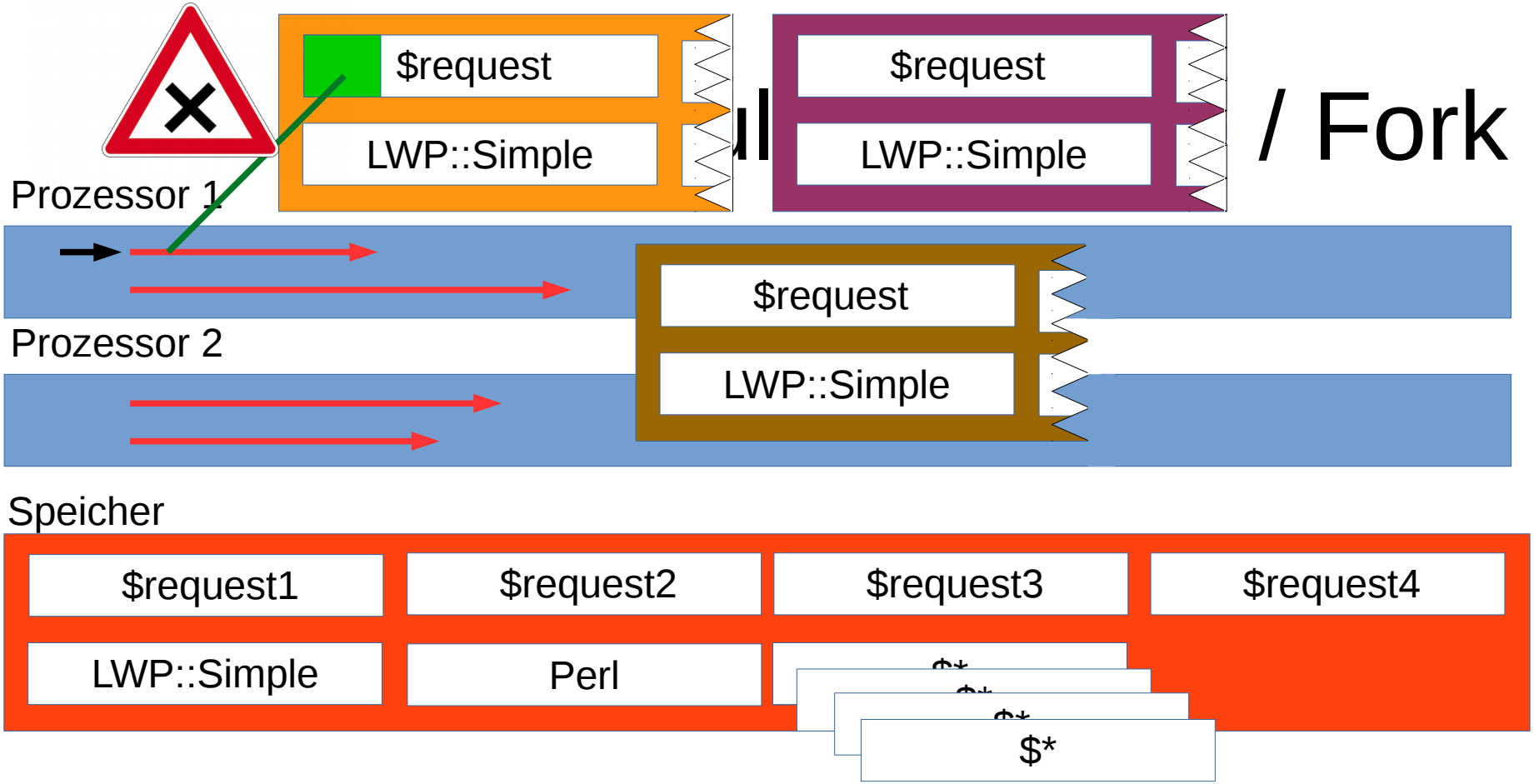

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



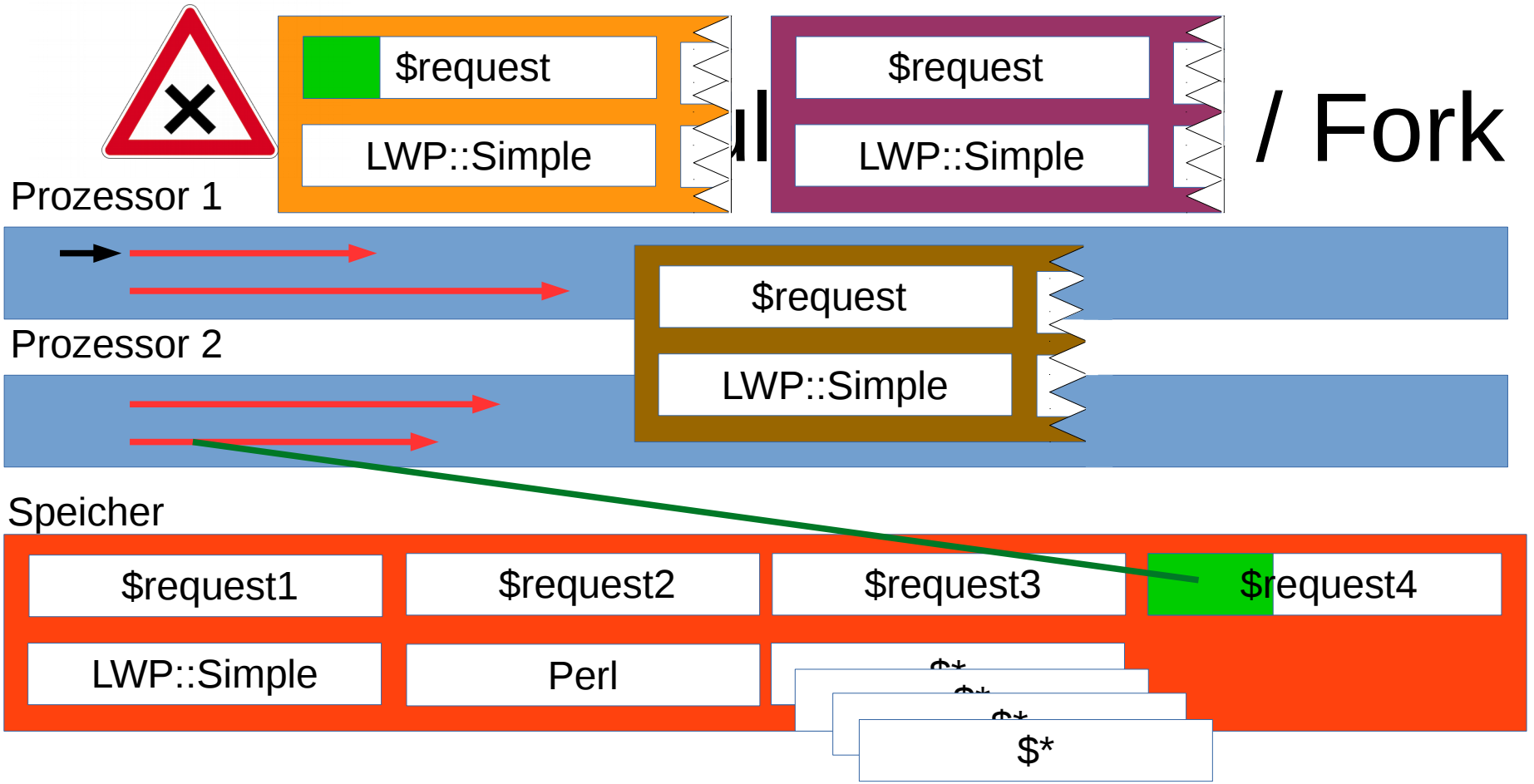

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



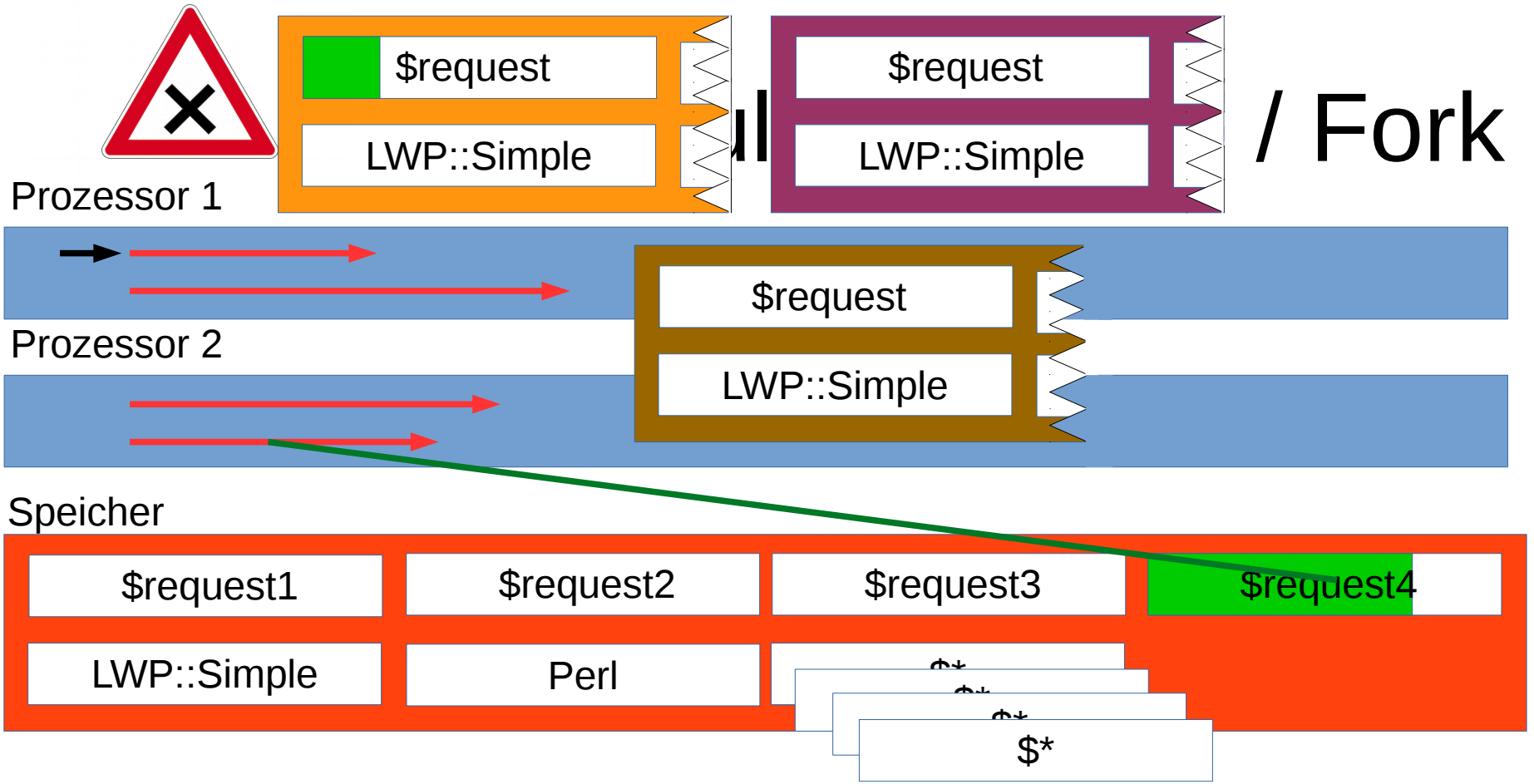

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



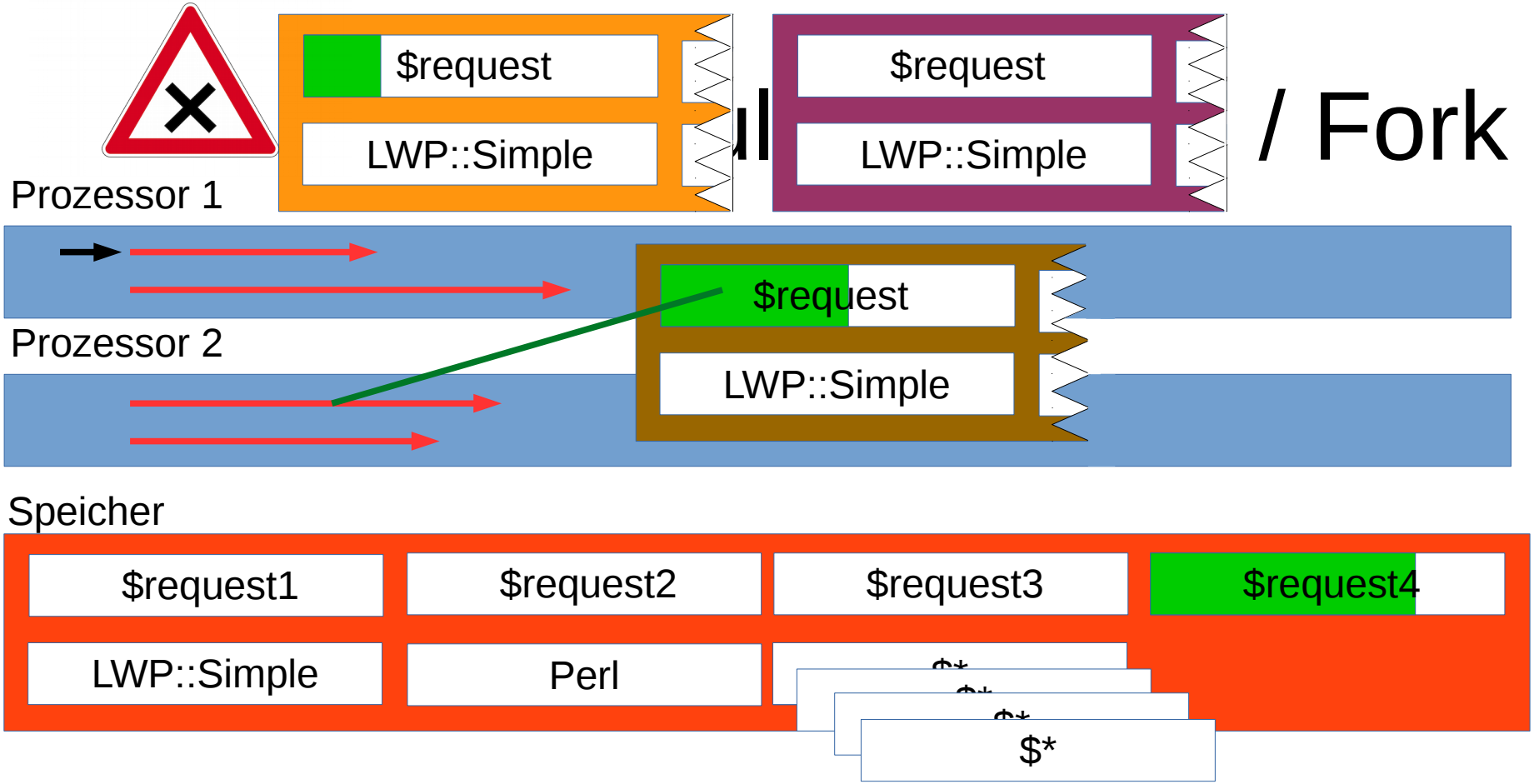

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.



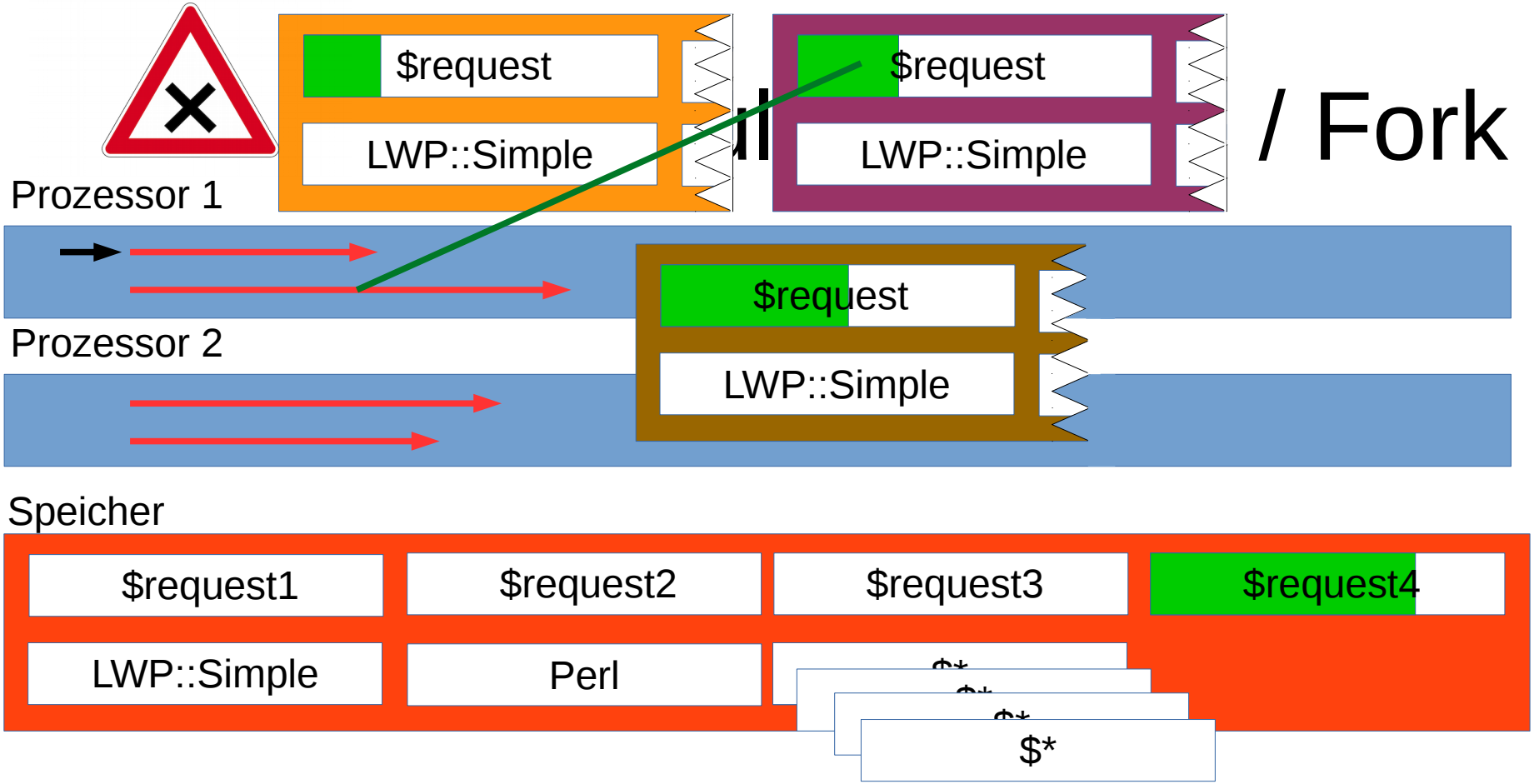

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.



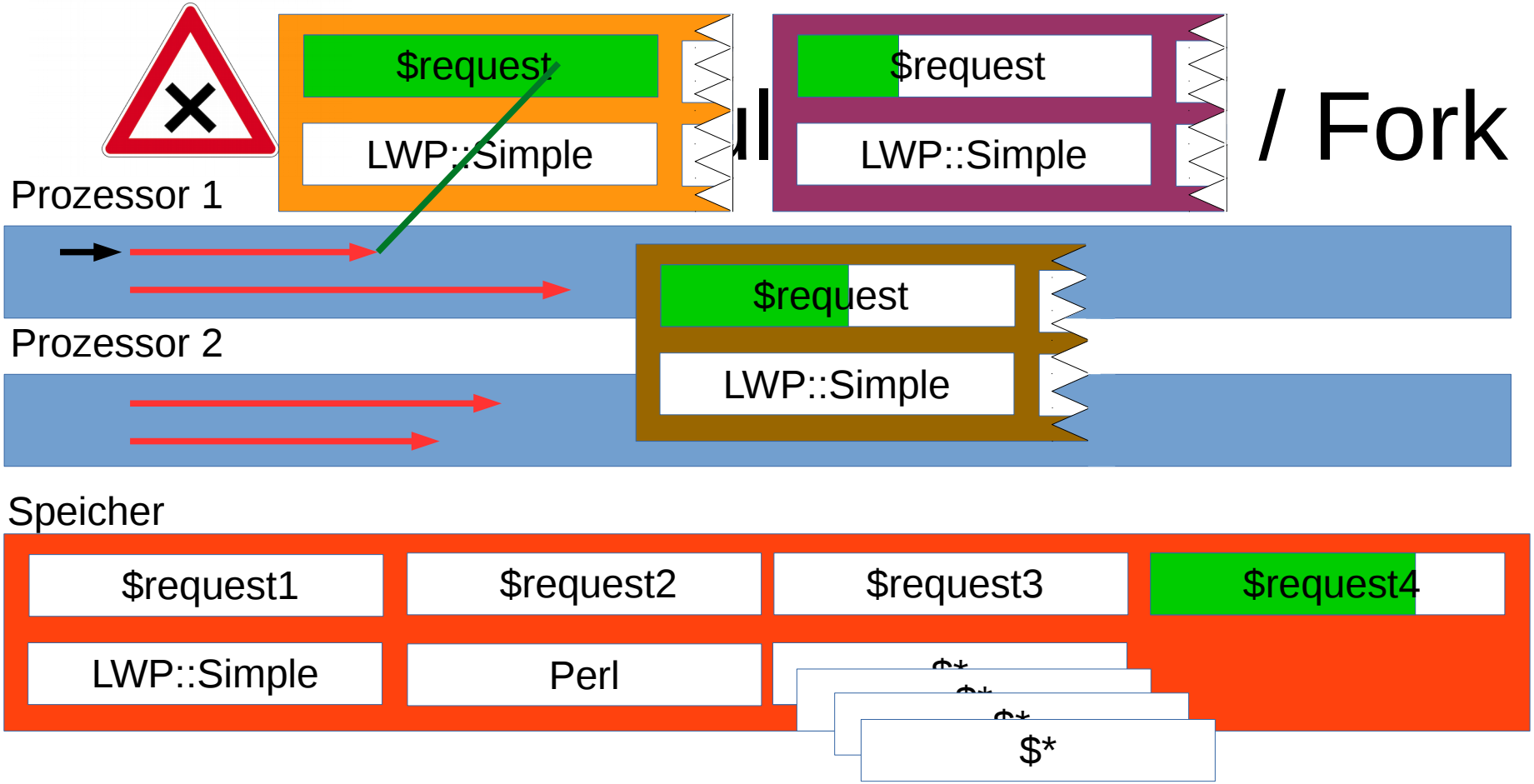

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



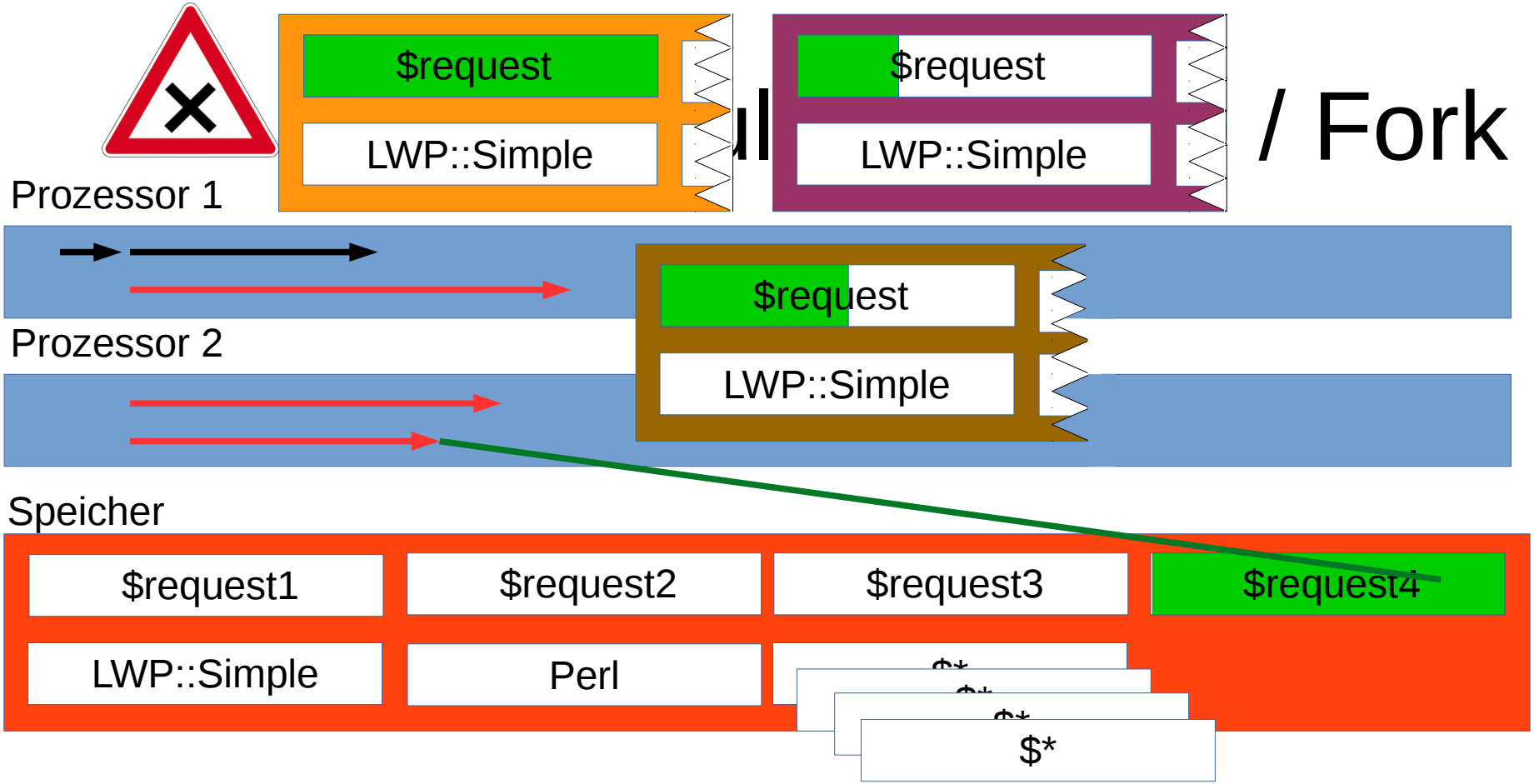

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

CRYPTO MAGIC einfach. sicher. verbunden.



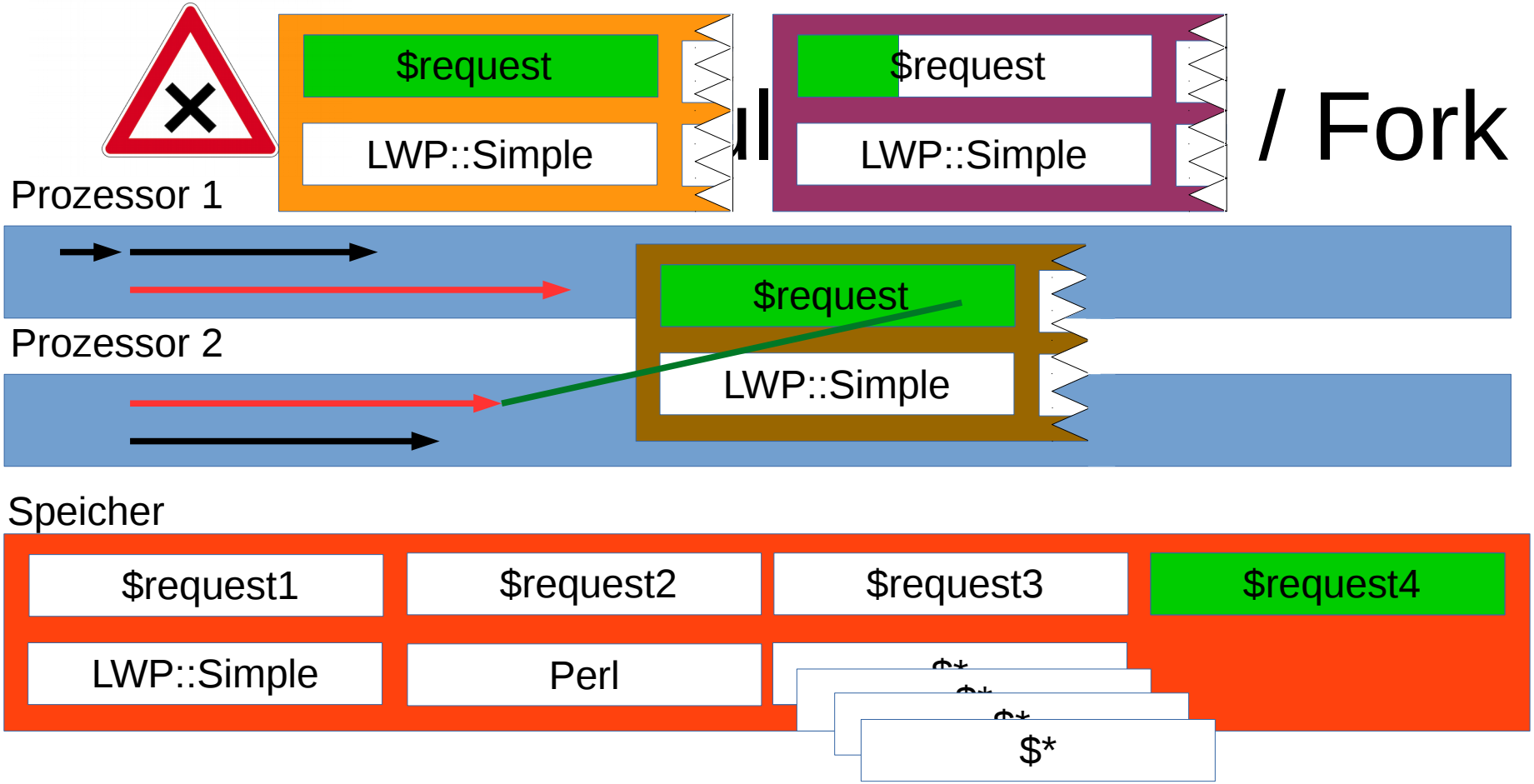

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.




```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

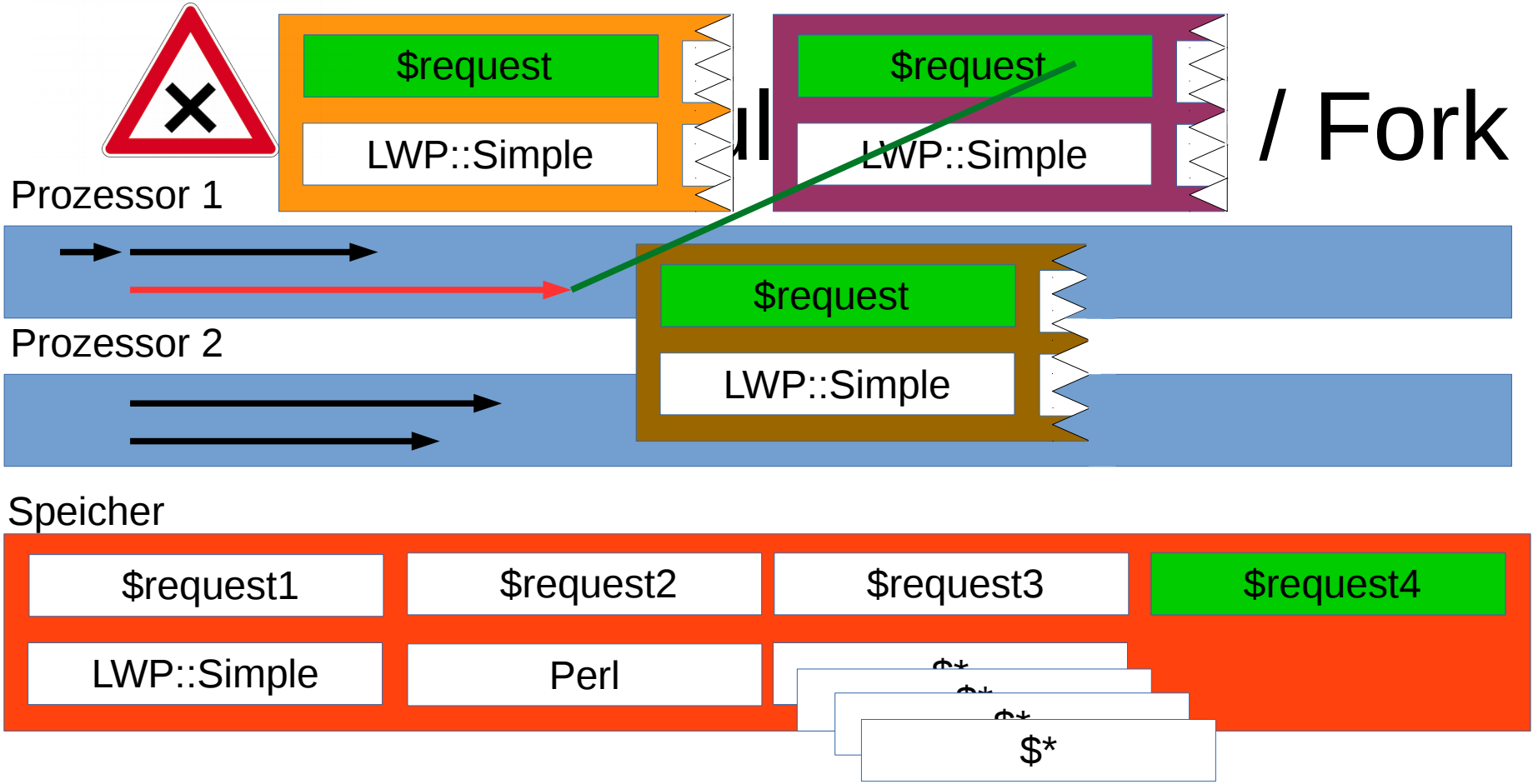
CRYPTO MAGIC einfach. sicher. verbunden.



/ Fork

```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

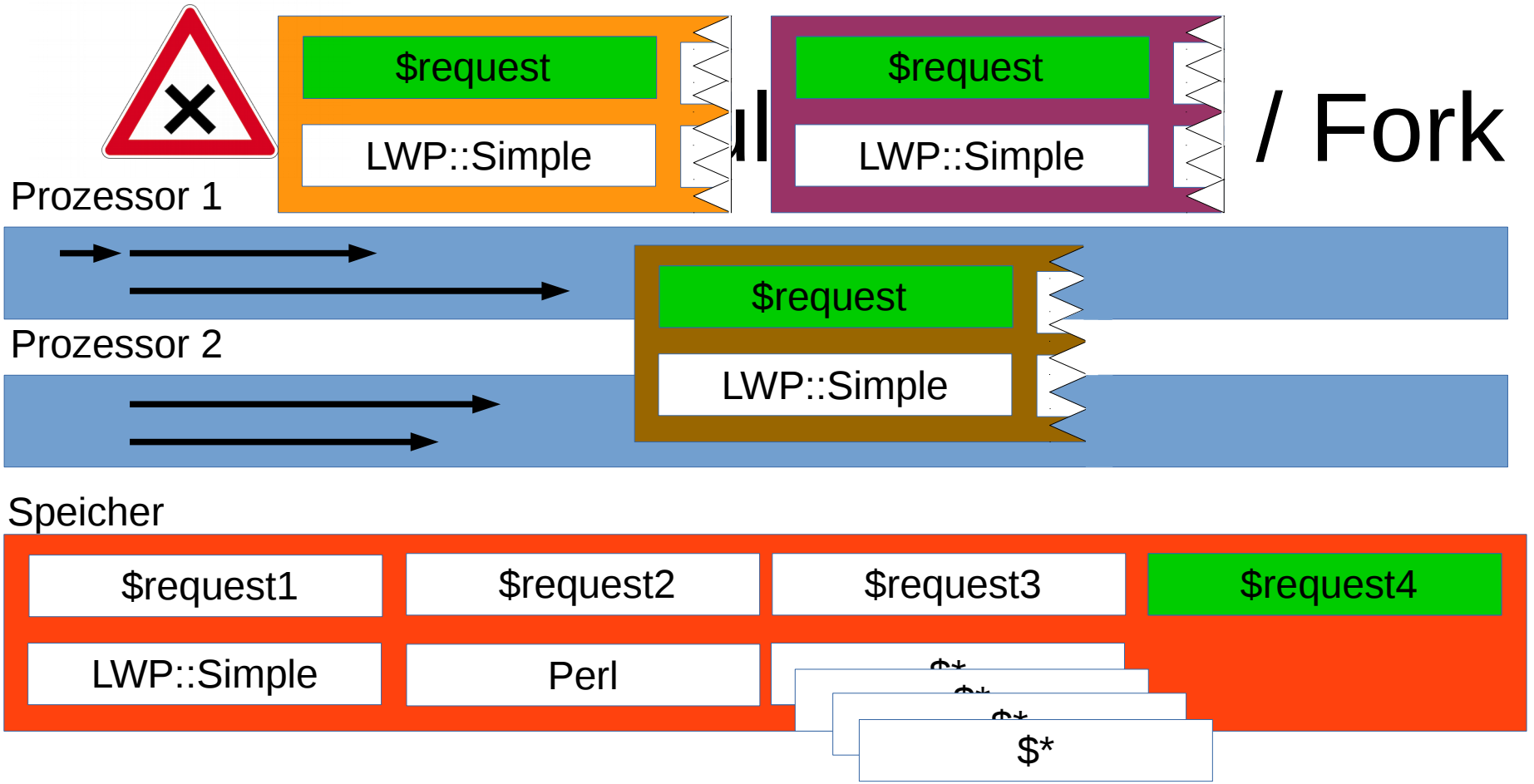
CRYPTO MAGIC einfach. sicher. verbunden.



/ Fork

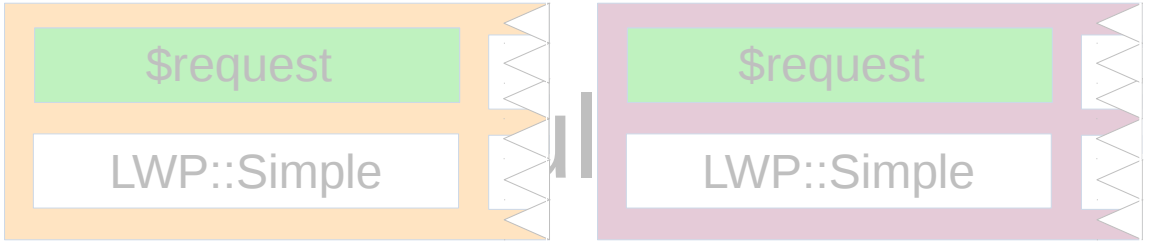
```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.



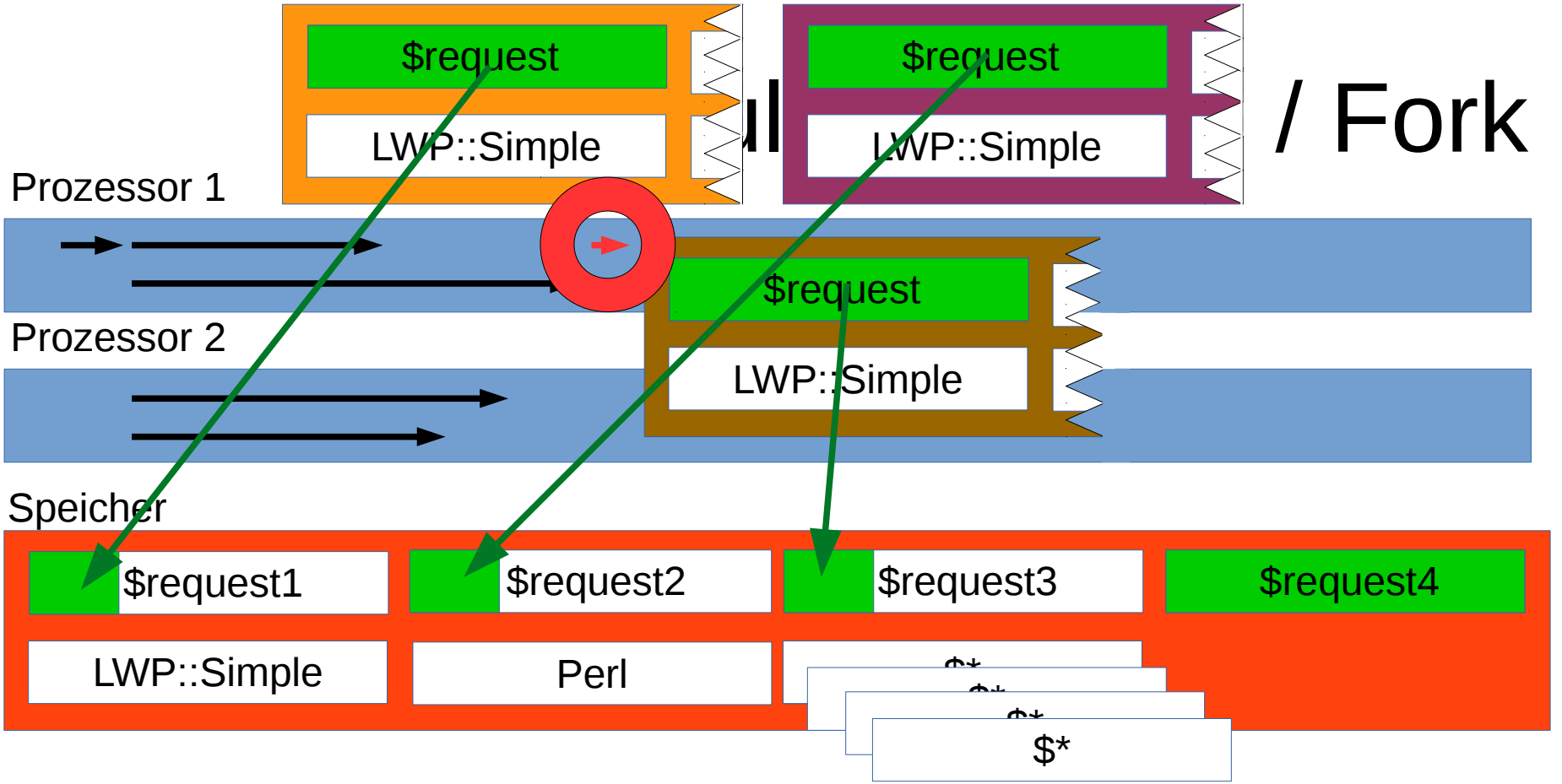
/ Fork

Prozessor 1

Multithreading-Perl / Fork
= Laufzeit-Superhero?

```
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

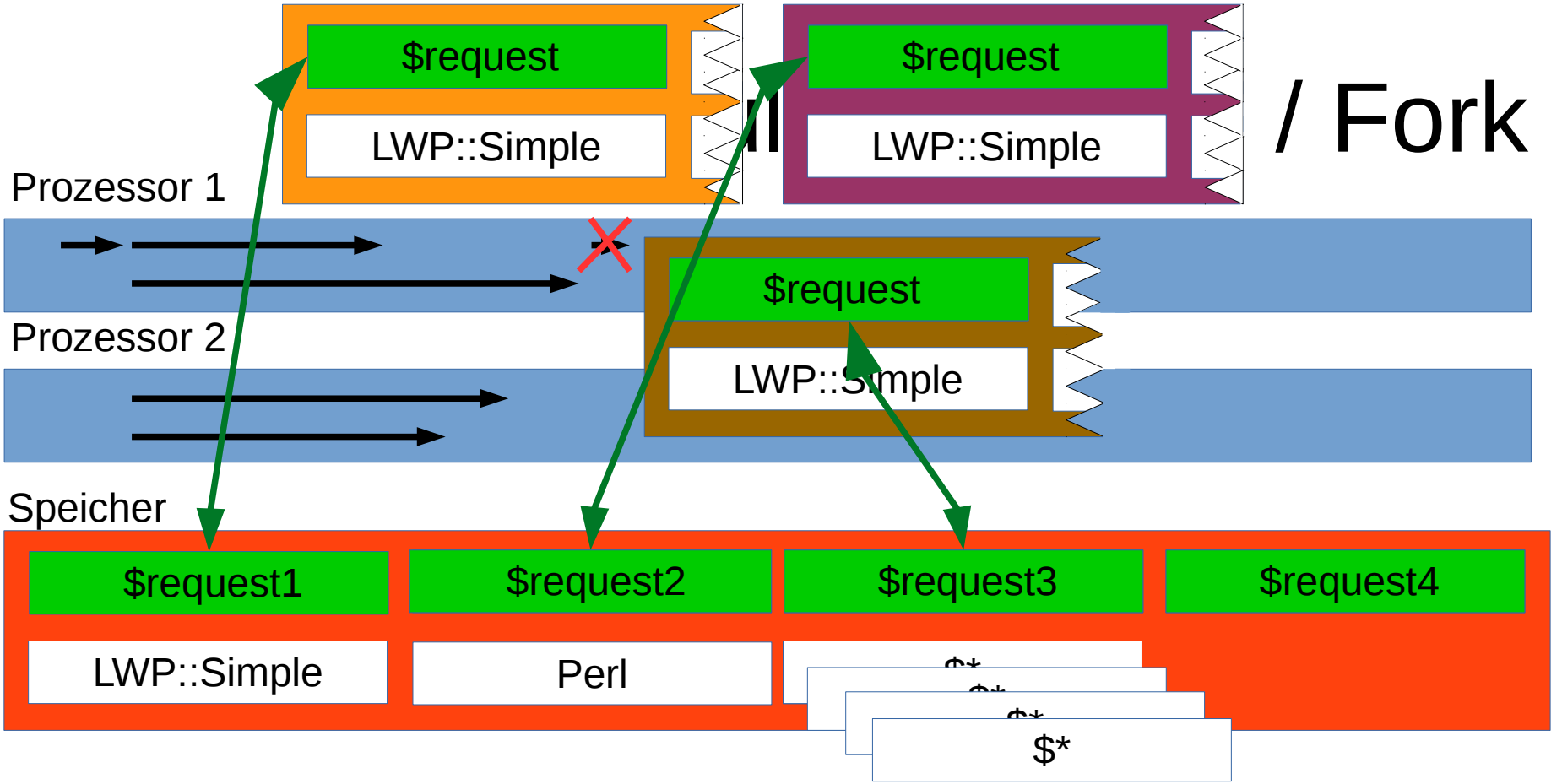
/ Fork




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

Datenübertragung?

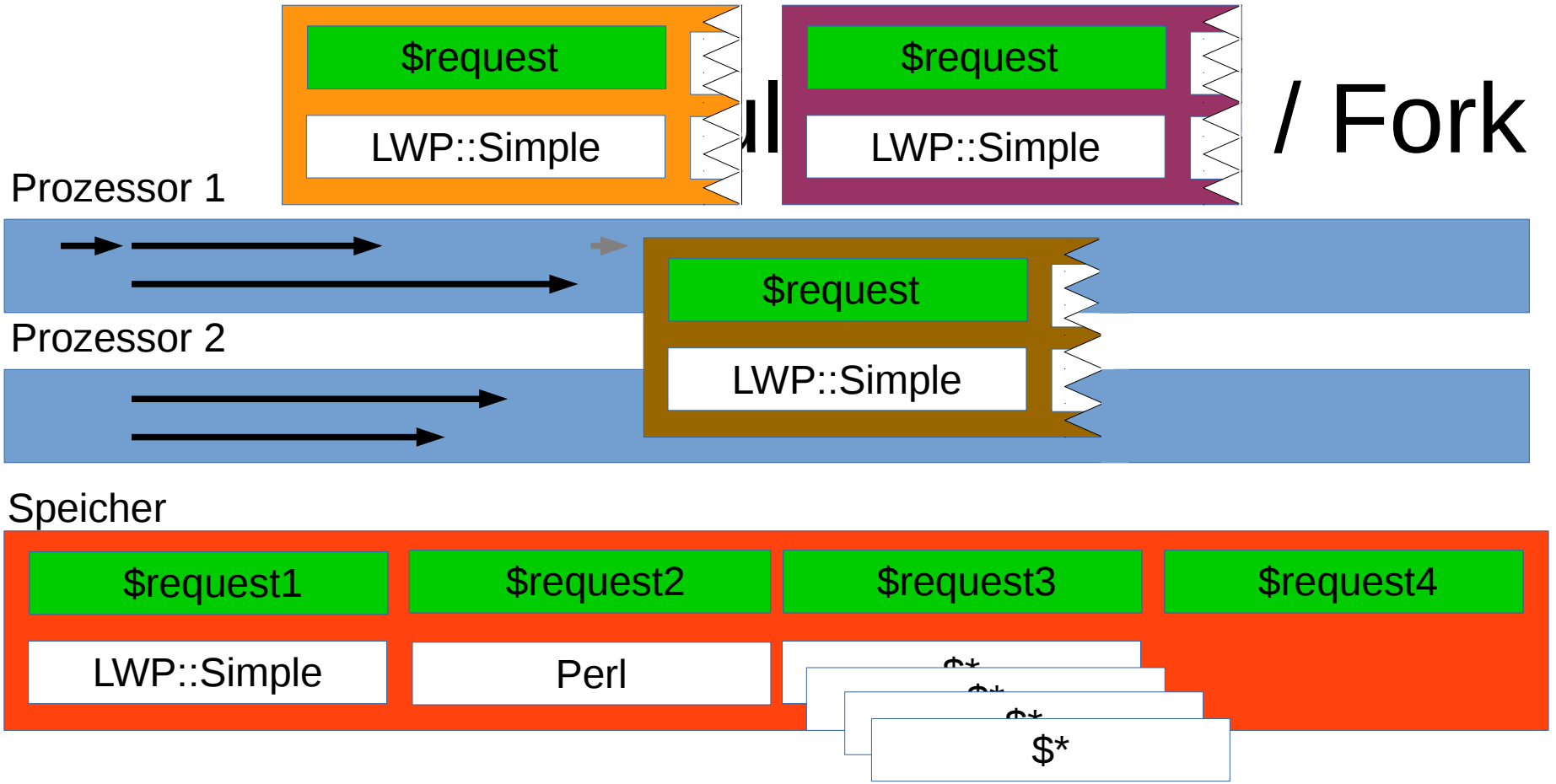
CRYPTO MAGIC einfach. sicher. verbunden.




```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

Datenübertragung? → Shared Memory

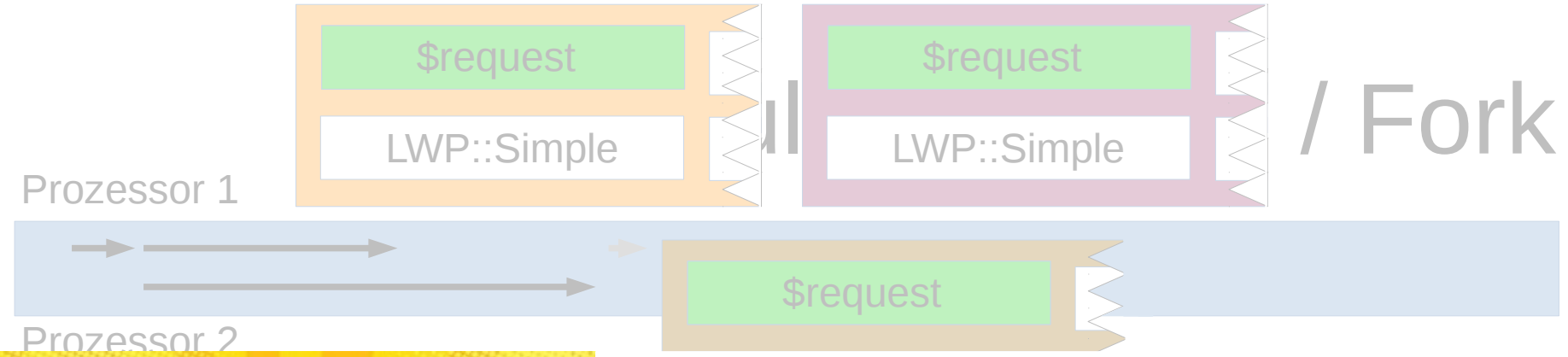
CRYPTO MAGIC einfach. sicher. verbunden.




```
use LWP::Simple;
my $request1 = get("https://www.cryptomagic.eu/");
my $request1 = get("https://www.heise.de/");
my $request3 = get("https://blog.fefe.de/");
my $request4 = get("https://gesicherte.email/");
```

Datenübertragung? → Shared Memory

CRYPTO MAGIC einfach. sicher. verbunden.

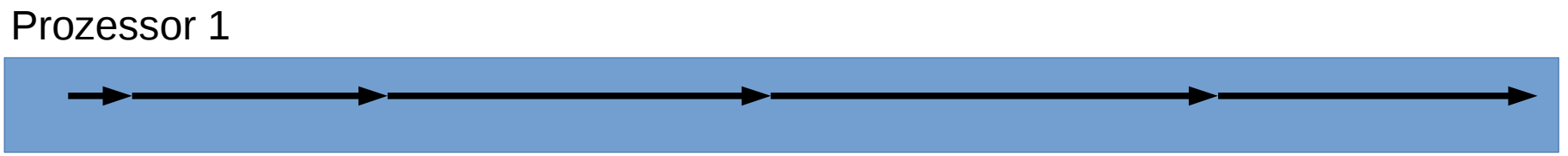


Multithreading-Perl / Fork = Laufzeit-Superhero?

- Aufpassen mit allokierten Resourcen (z.B. Filedescriptoren)
- Fork: Keine Unterstützung unter Windows → MT-Perl
- MT-Perl: Kopieraufwand für Perl pro erzeugten Thread
- Fork: Kopieraufwand für Daten wenn kein SharedMem.
- MT-Perl: chdir, chroot, … problematisch
- Fork: Shared-Memory: Nur Low-Level API?
- Fork: Copy-on-write ist kein wirkliches kopieren
- Same/Shared-Memory: Locking !

my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
Datenübertragung? → Shared Memory

Synchron?

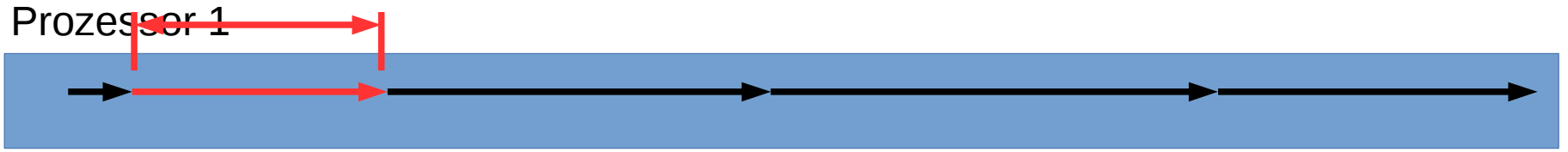



```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron?

3,8 Sekunden




```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

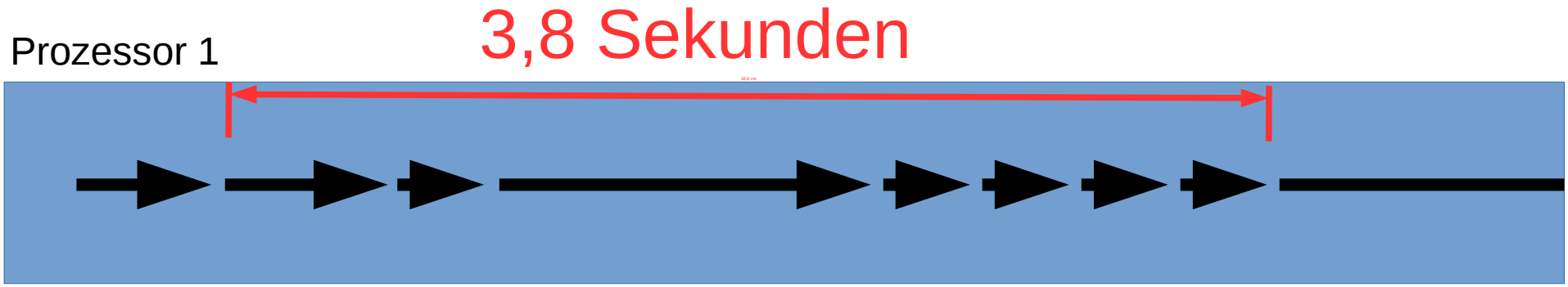
Synchron?


```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
...

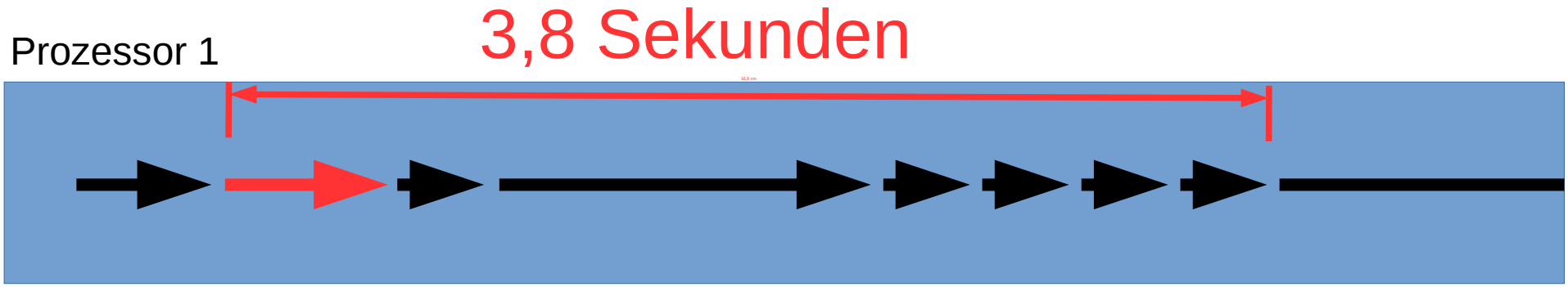



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN

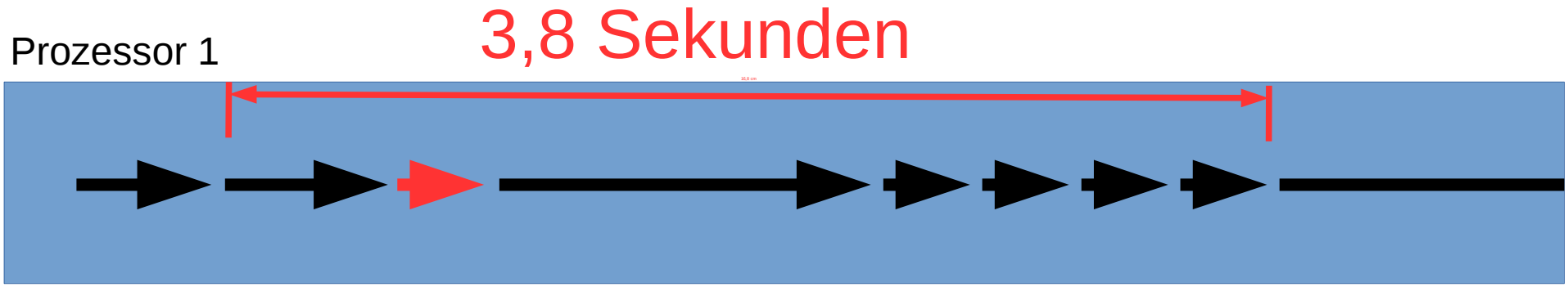



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN

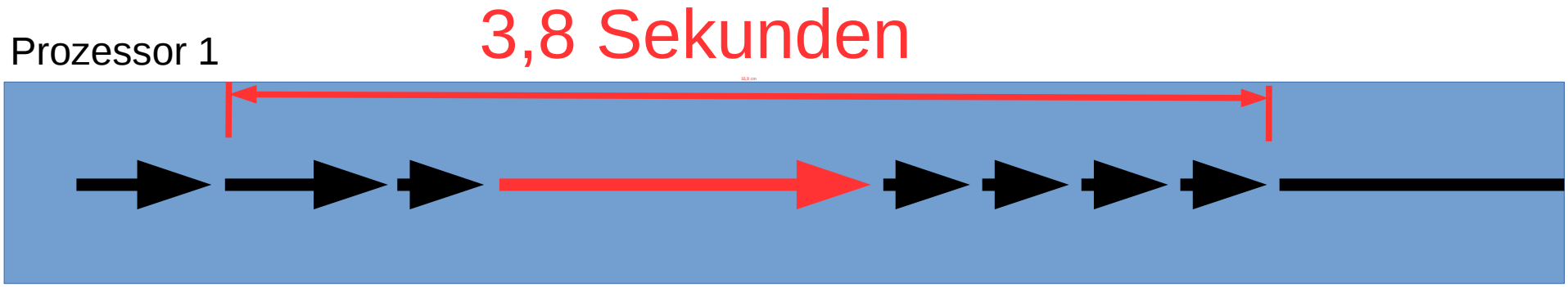



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN

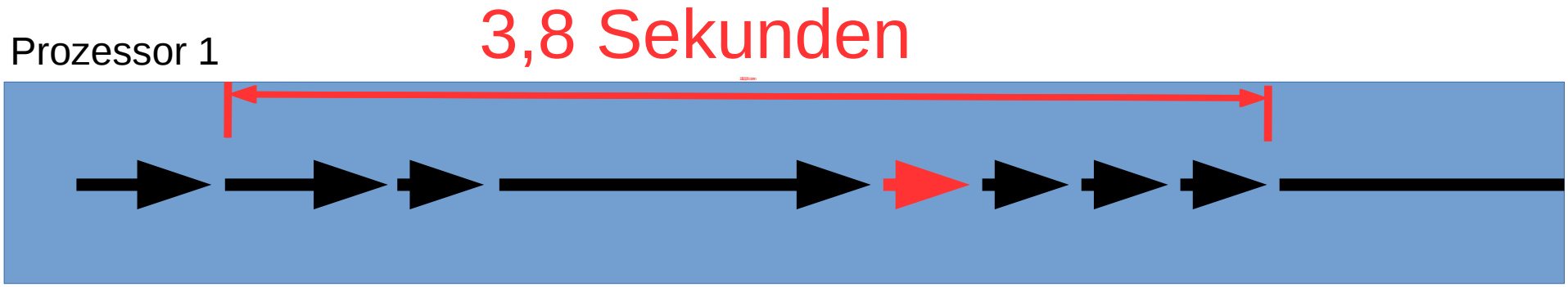



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen

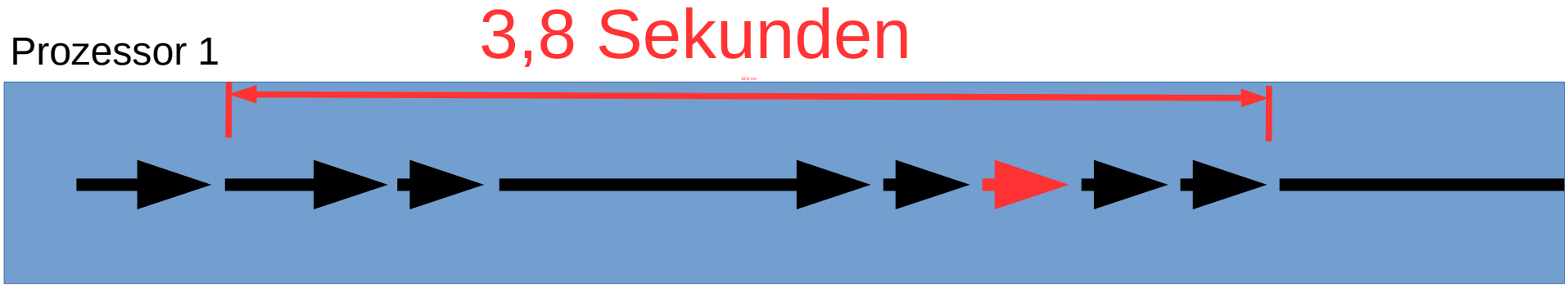



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN

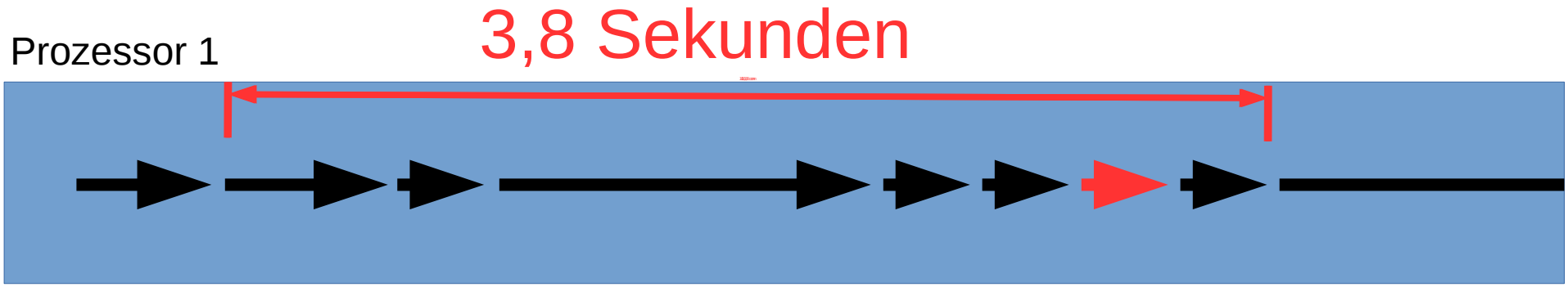



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen

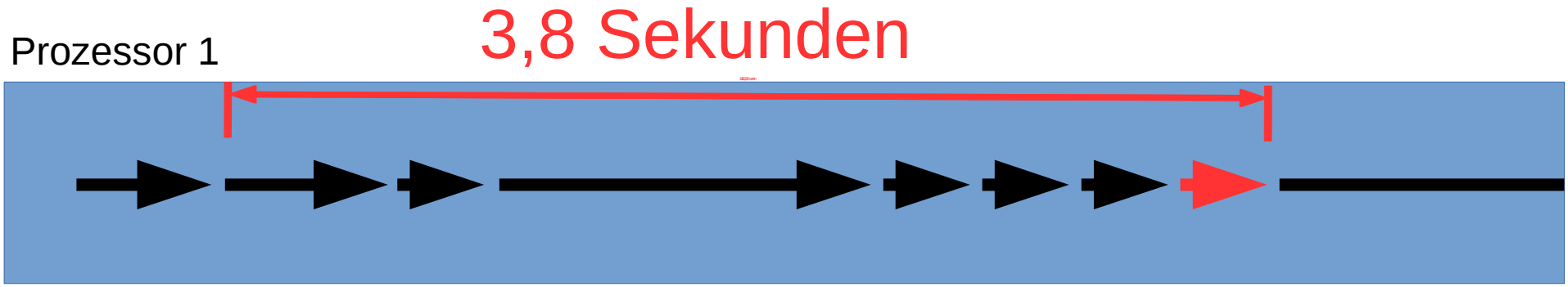



```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN




```
connect($socket, ...);
write($socket, „GET / HTTP/1.0\r\n\r\n“);
while (read($socket, $in) > 0) {
  $request .= $in;
}
```

CRYPTO MAGIC einfach. sicher. verbunden.

Synchron? → EventLoop

Ausführen
und
WARTEN

Prozessor 1

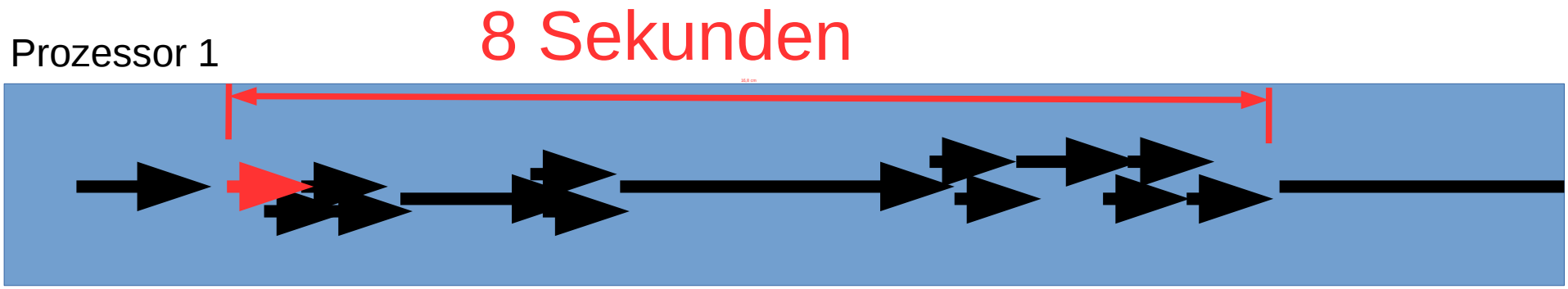
3,8 S

```
connect($socket
write($socket, „G
while (read($soc
  $request .= $in;
}
```

Asynchron non-blocking
oder auch:
EventLoop

EventLoop

Ausführen

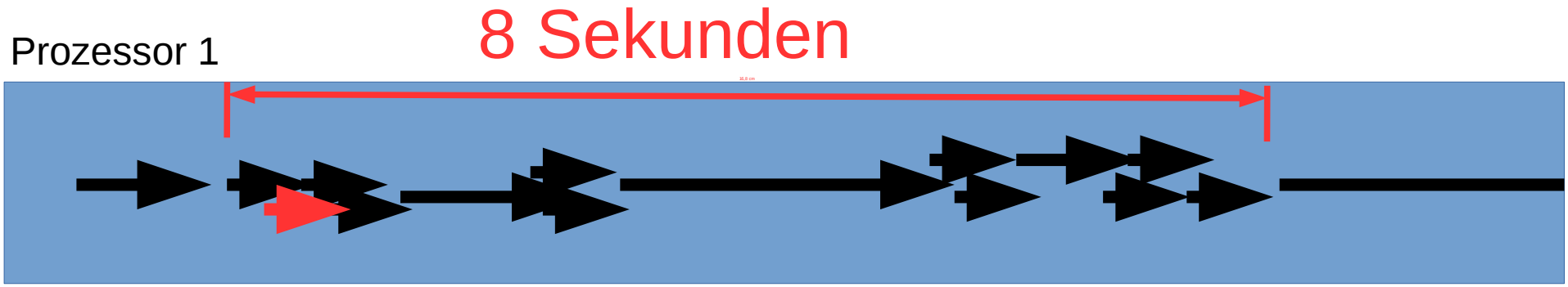



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

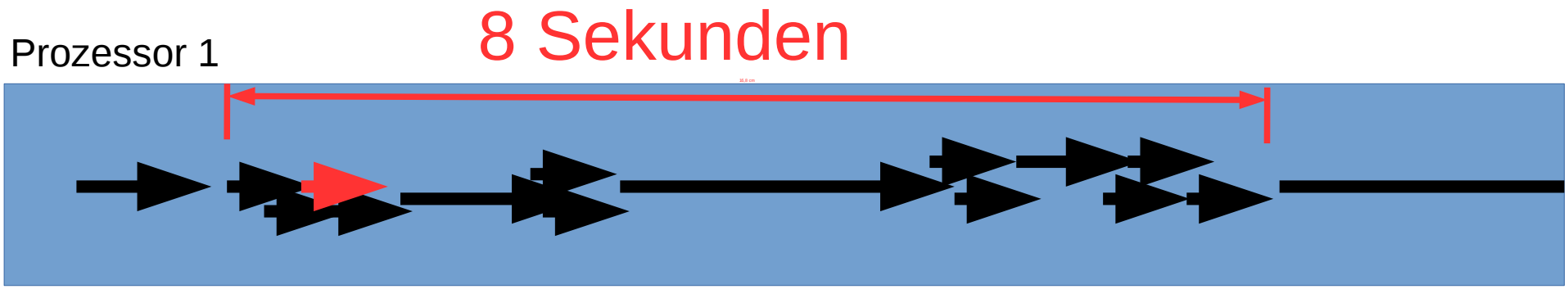
Ausführen




```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

EventLoop

Ausführen

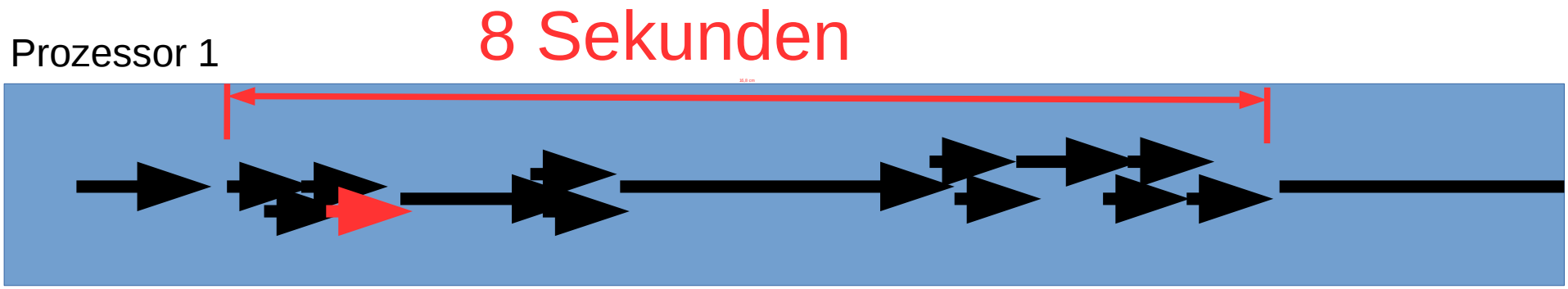



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen

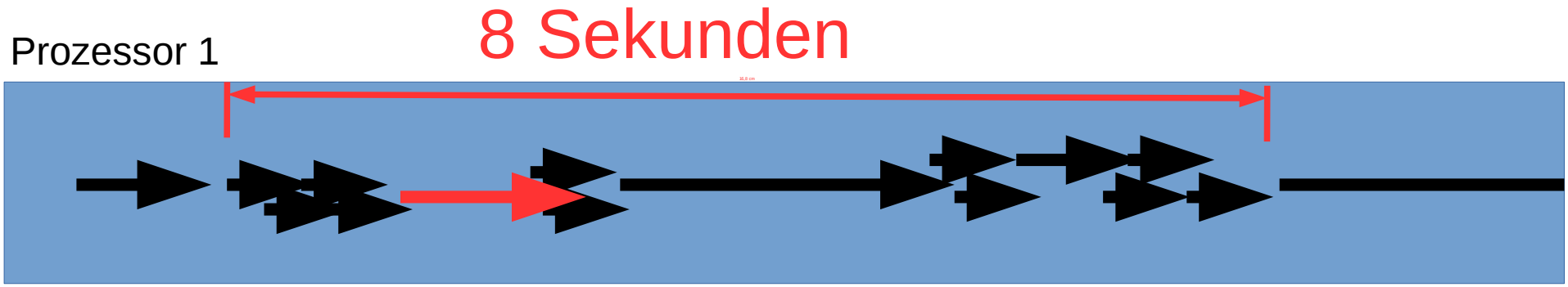



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen
und gesteuert
WARTEN

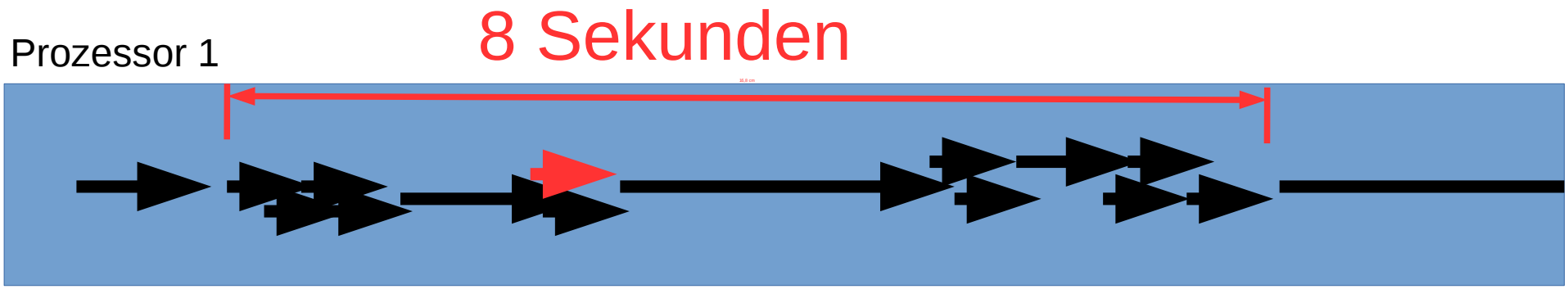



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

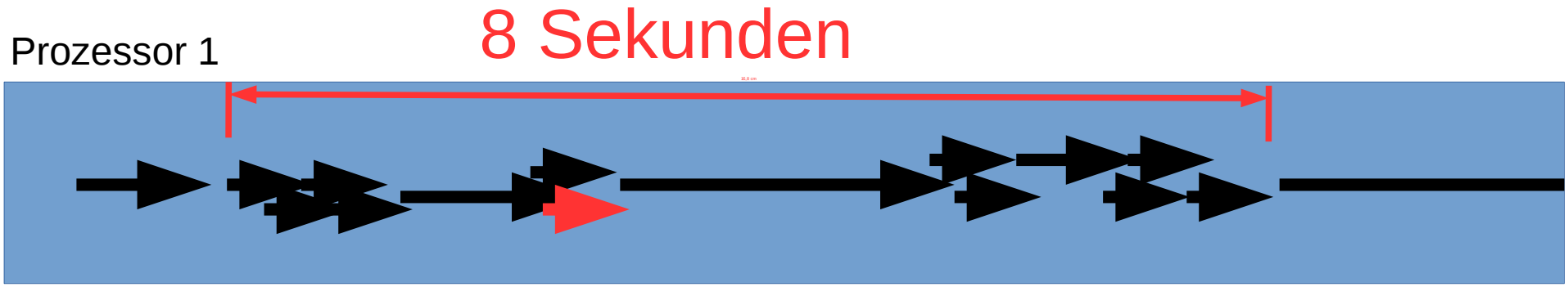



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

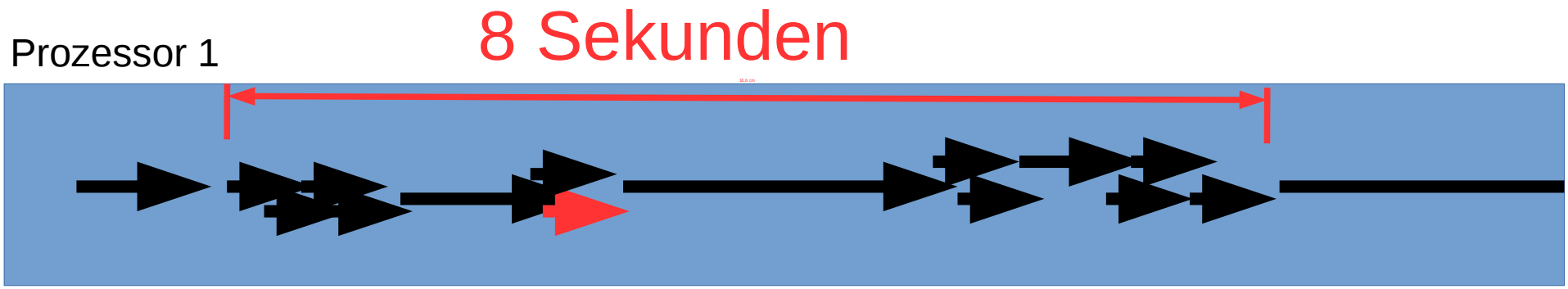



```
fcntl($sock1, F_SETFL, O_NONBLOCK);
fcntl($sock2, F_SETFL, O_NONBLOCK);
connect($sock1, ...);
connect($sock2, ...);
while (select($sock1 | $sock2, undef) != $sock1 & $sock2);
write($sock1, „GET / HTTP/1.0\r\n\r\n“);
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

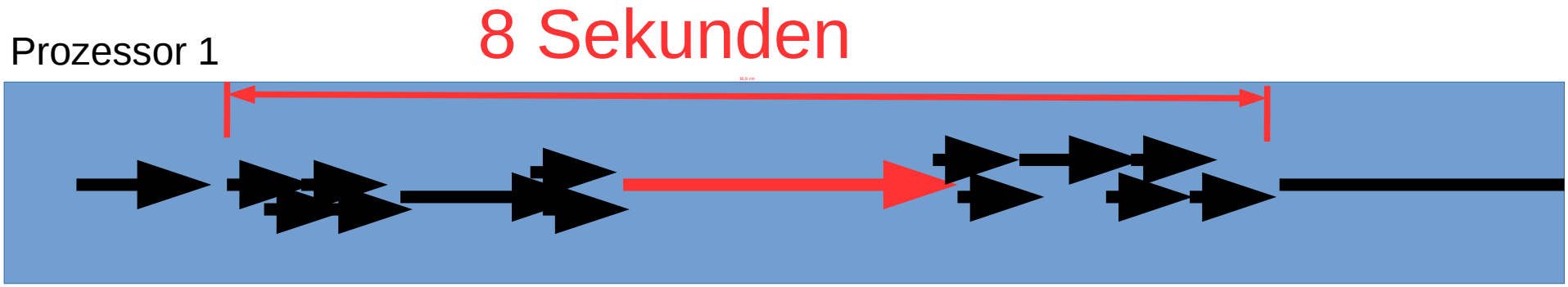



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

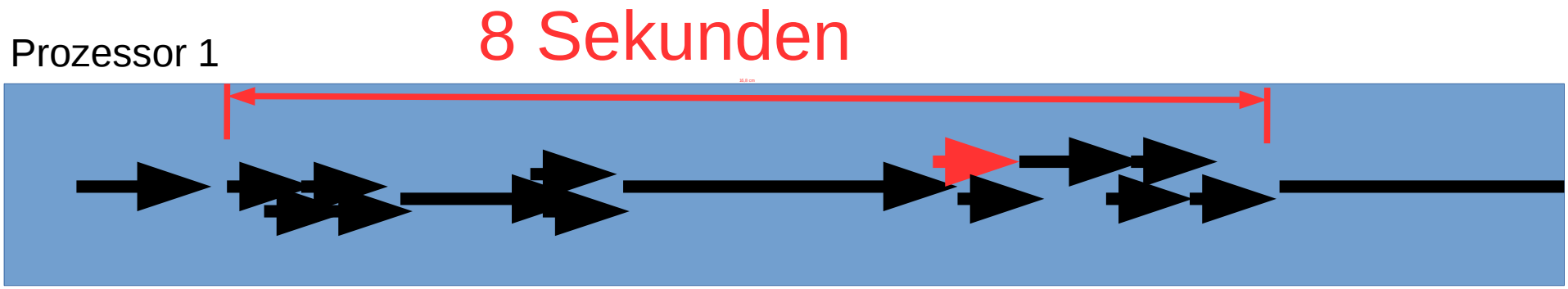
Ausführen
und gezielt
WARTEN




```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

EventLoop

Ausführen ohne WARTEN

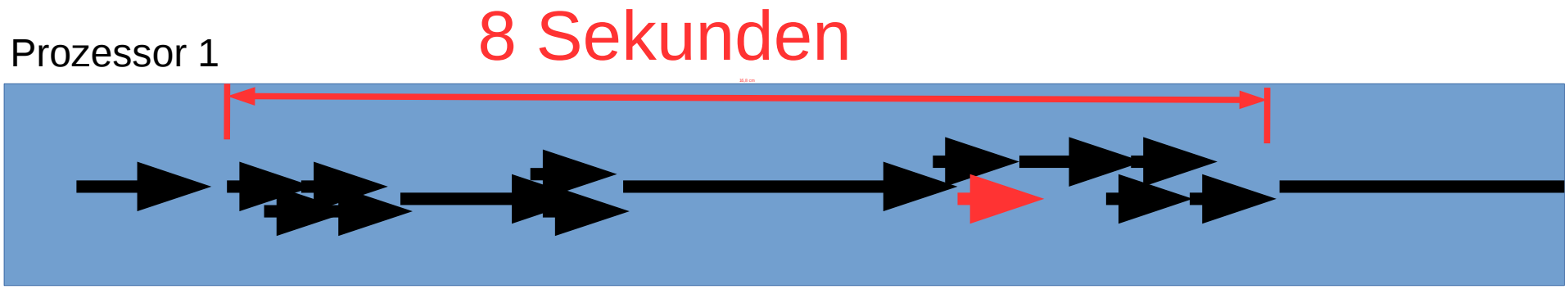



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

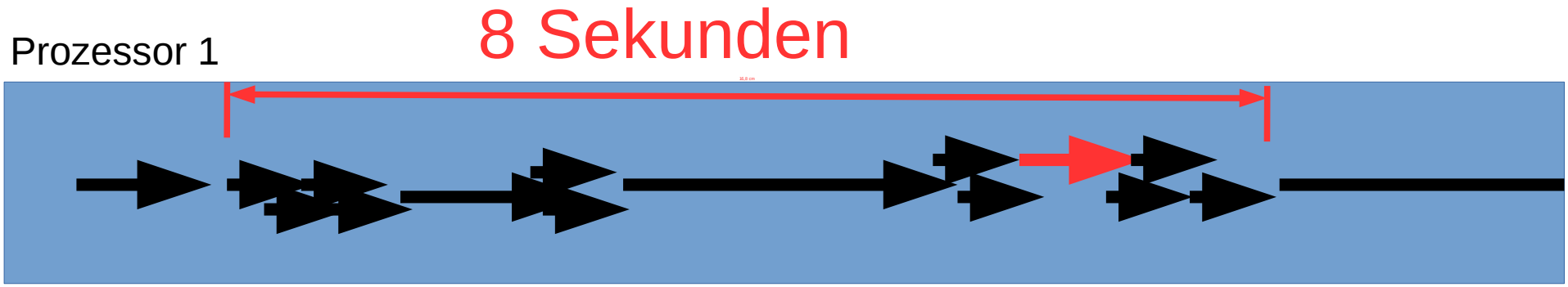
Ausführen ohne WARTEN




```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

EventLoop

Ausführen
und gezielt
WARTEN

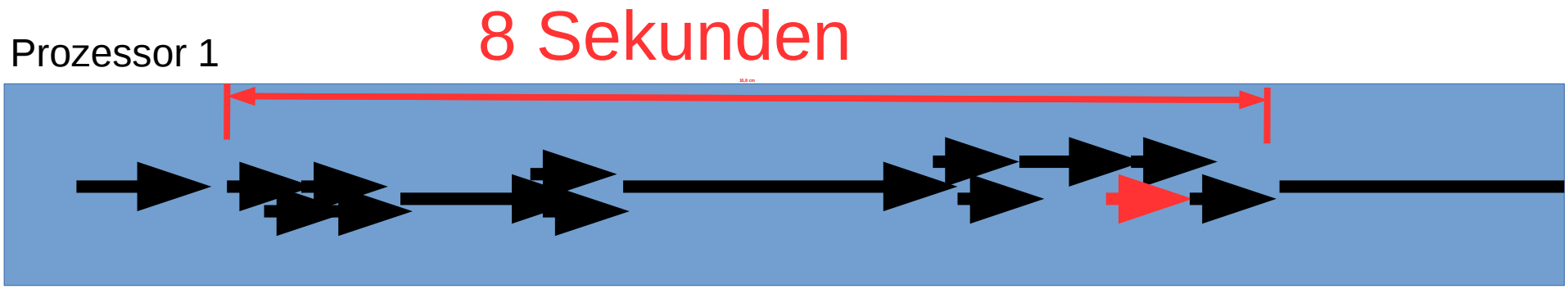



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

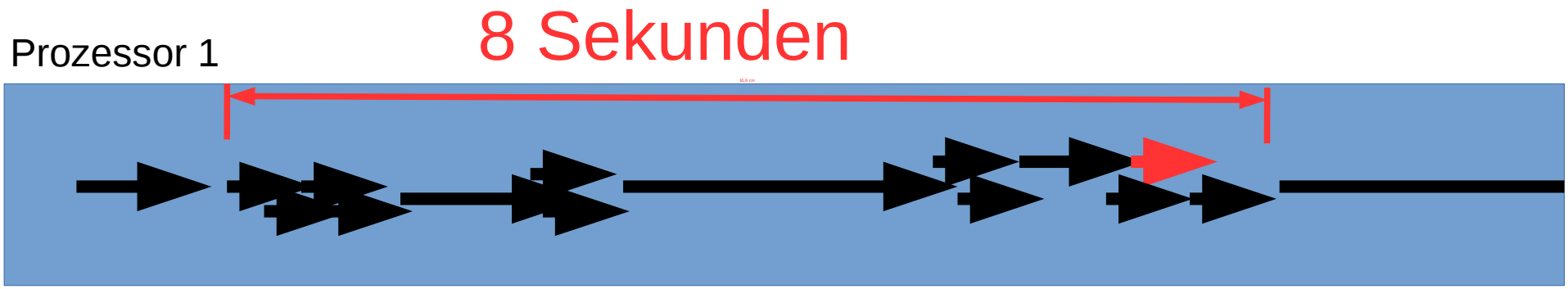



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

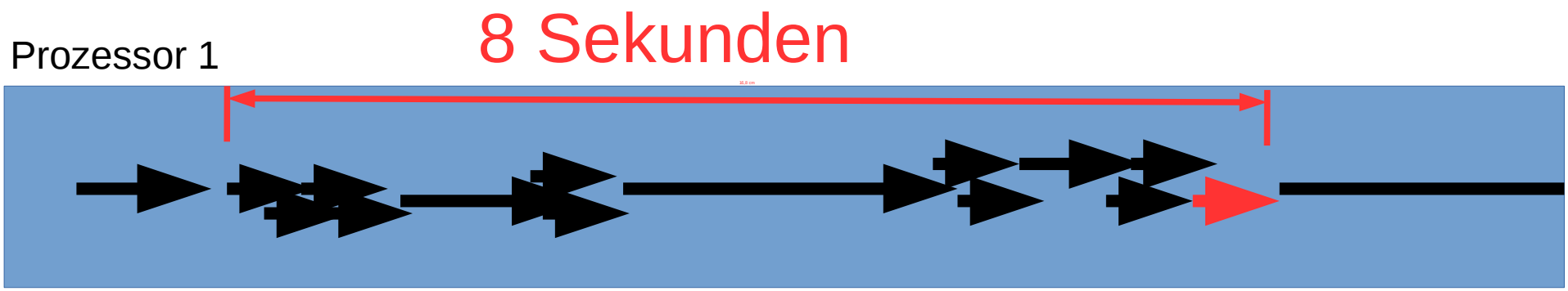



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen ohne WARTEN

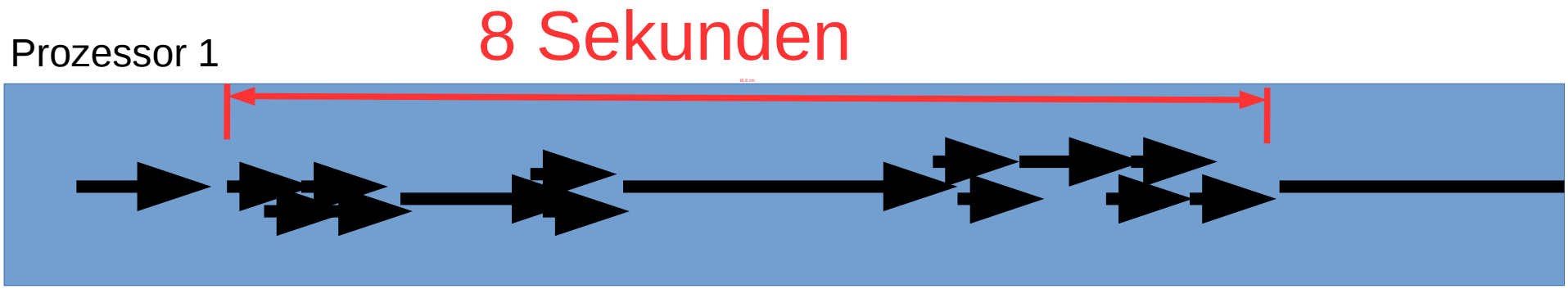



```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

RYPTO MAGIC einfach. sicher. verbunden.

EventLoop

Ausführen
ohne
WARTEN




```
write($sock2, „GET / HTTP/1.0\r\n\r\n“);
while ($ready = select($sock1 | $sock2, undef)) {
   $return1 .= $in
      if (($ready & $sock1) && read($sock1, $in);
   $return2 .= $in
      if (($ready & $sock2) && read($sock2, $in);
}
```

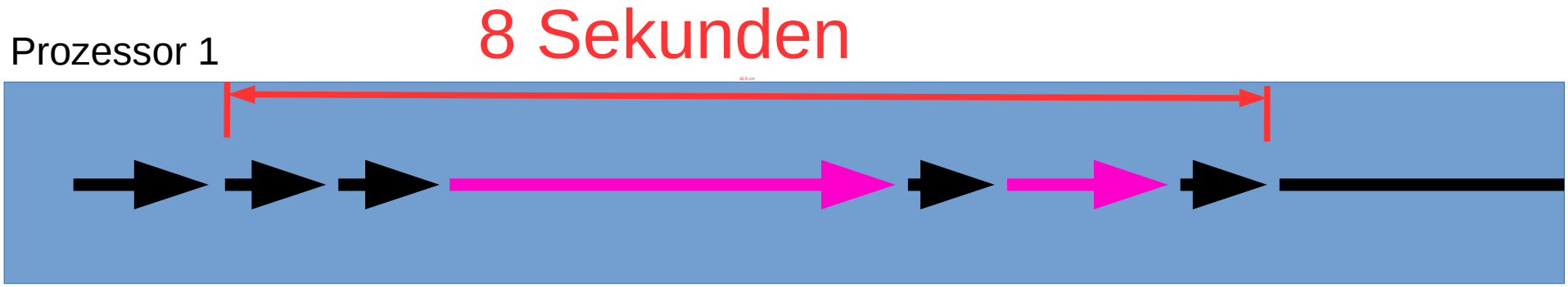
EventLoop

Ausführen ohne WARTEN

Preis: Die totale Zerstückelung

```
write(
while (
    $retur                                    {
        if (($ready & $sock1) && read($sock1, $in);
    $return2 .= $in
        if (($ready & $sock2) && read($sock2, $in);
}
```

EventLoop → Node.JS/POE

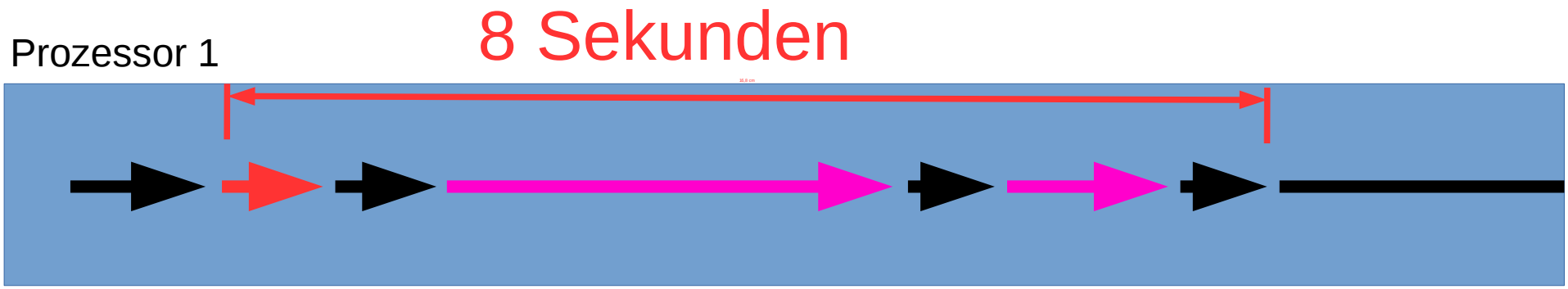



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen ohne WARTEN

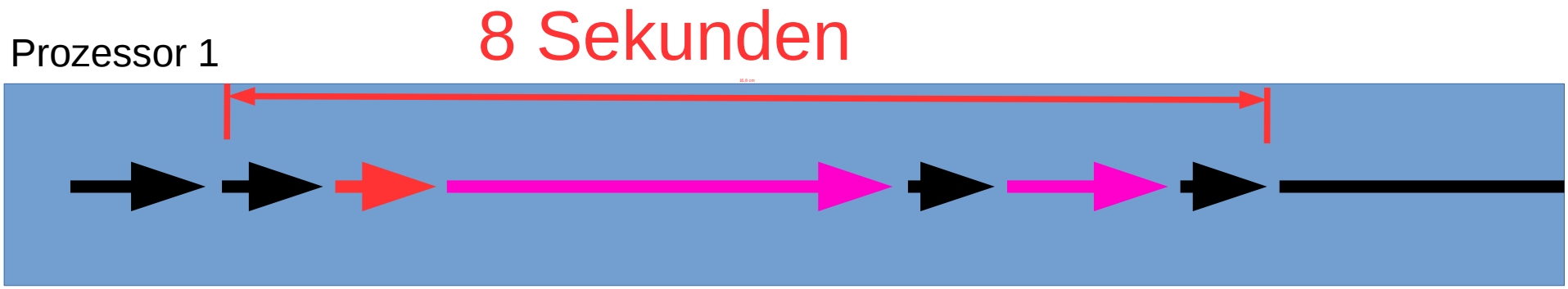



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen ohne WARTEN

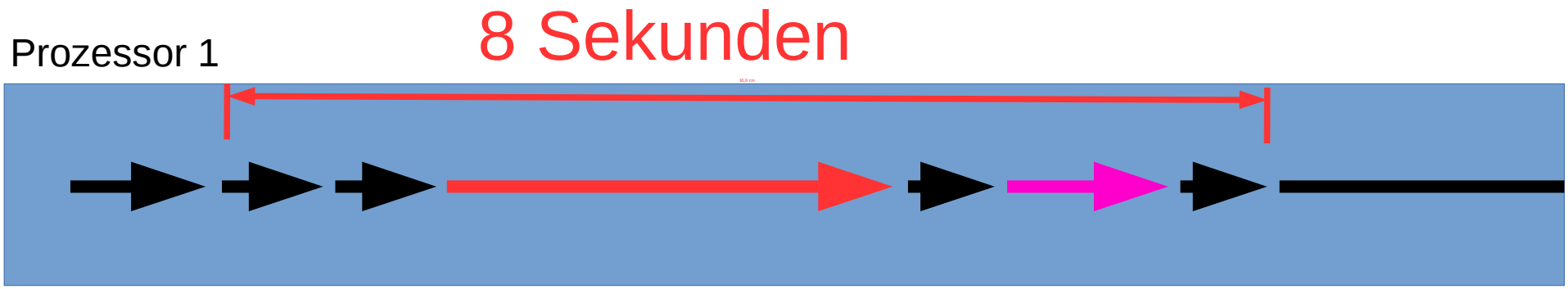



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen
mit gezieltem
WARTEN

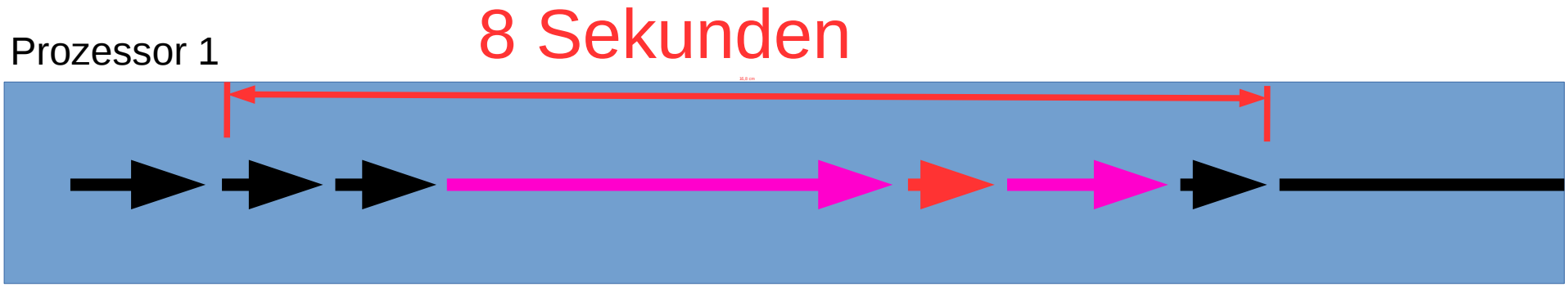



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen ohne WARTEN

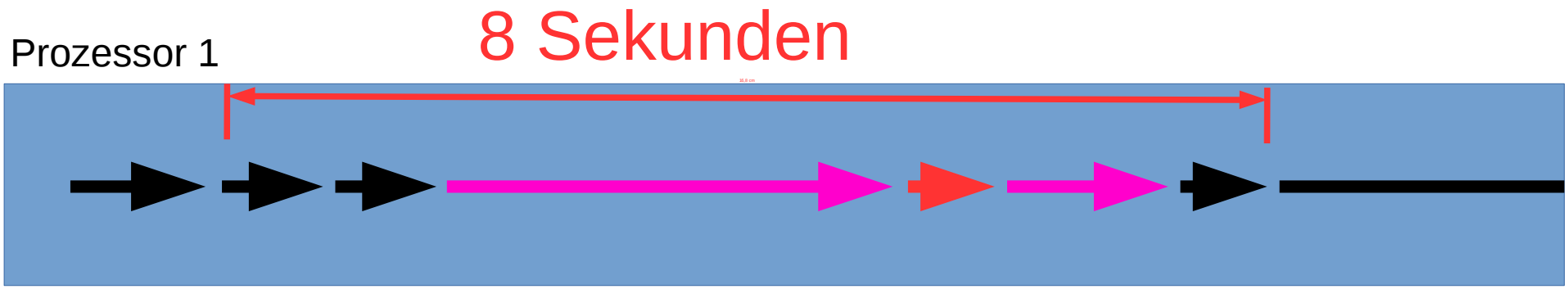



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen ohne WARTEN



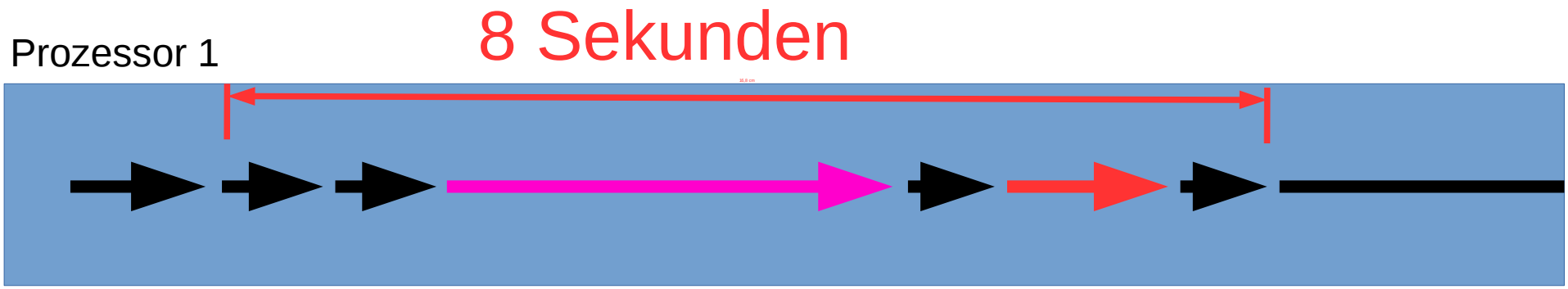

```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

Ausführen ohne WARTEN

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

Ausführen
mit gezieltem
WARTEN

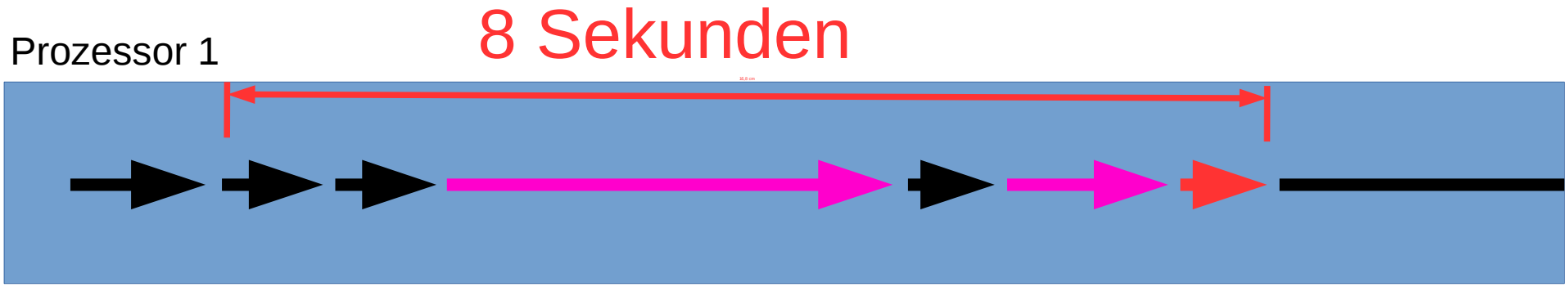



```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE

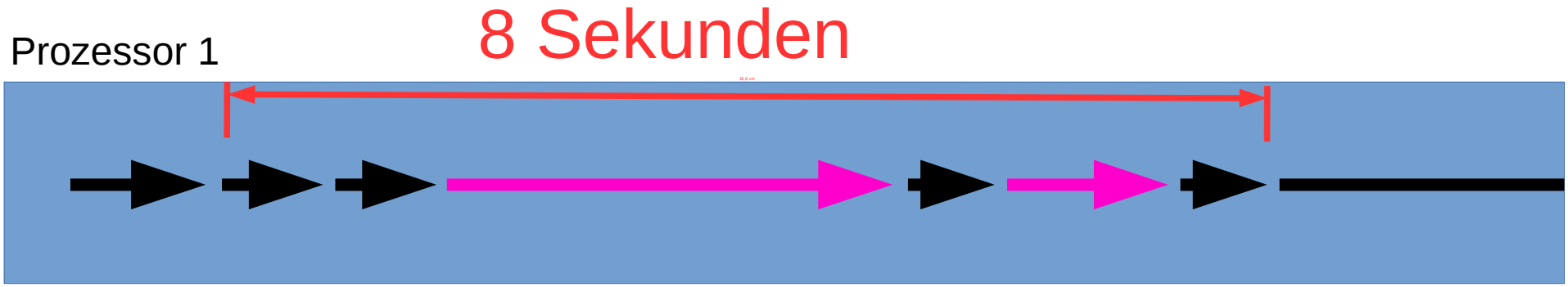
Ausführen ohne WARTEN




```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

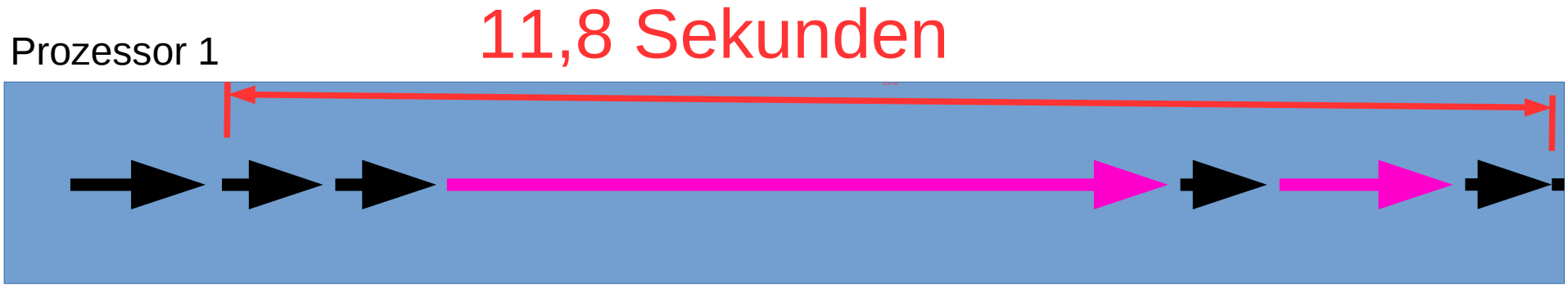
EventLoop → Node.JS/POE




```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
});
get("https://gesicherte.email/", sub {
   $request2 = shift;
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

EventLoop → Node.JS/POE




```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
   get("https://gesicherte.email/?data=".$request1, sub {
      $request2 = shift;
   });
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

Callback-Hölle

```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
   get("https://gesicherte.email/?data=".$request1, sub {
      $request2 = shift;
   });
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

Callback-Hölle

AsyncAwait-Himmel?

```
get("https://www.cryptomagic.eu/", sub {
   $request1 = shift;
   get("https://gesicherte.email/?data=".$request1, sub {
      $request2 = shift;
   });
});
run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

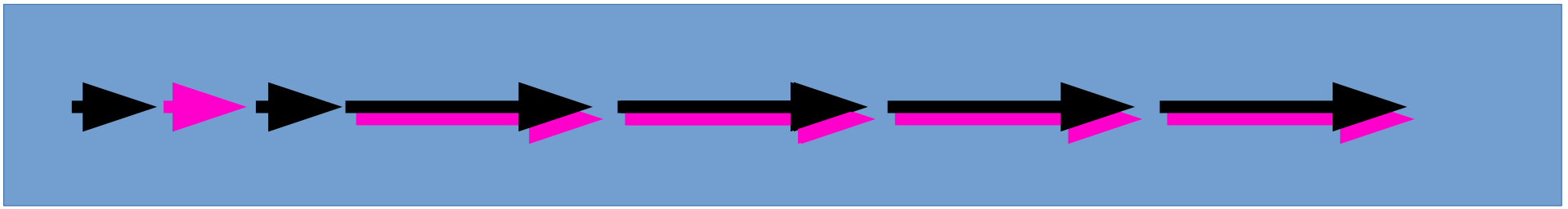
```
POE::Session->create(inline_states => {
  _start => sub {
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   },
});
$poe_kernel->run();
```

```
use LWP::Simple;
my $request1 = get(“https://www.cryptomagic.eu/");
my $request1 = get(“https://www.heise.de/“);
my $request3 = get(“https://blog.fefe.de/“);
my $request4 = get(“https://gesicherte.email/“);
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

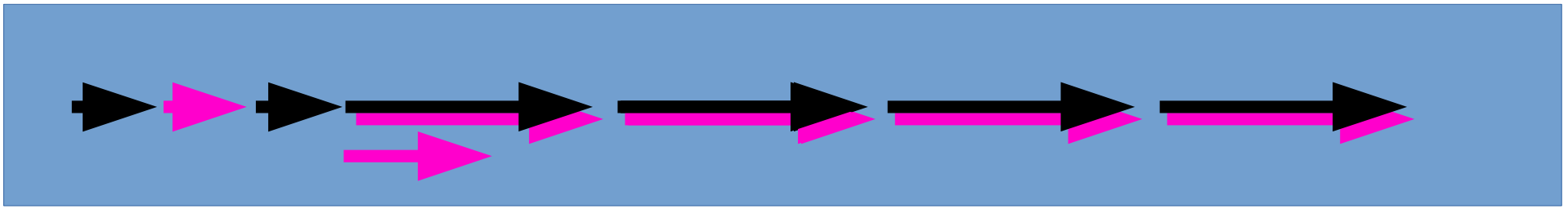


```
POE::Session->create(inline_states => {
  _start => sub {
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   },
});
$poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

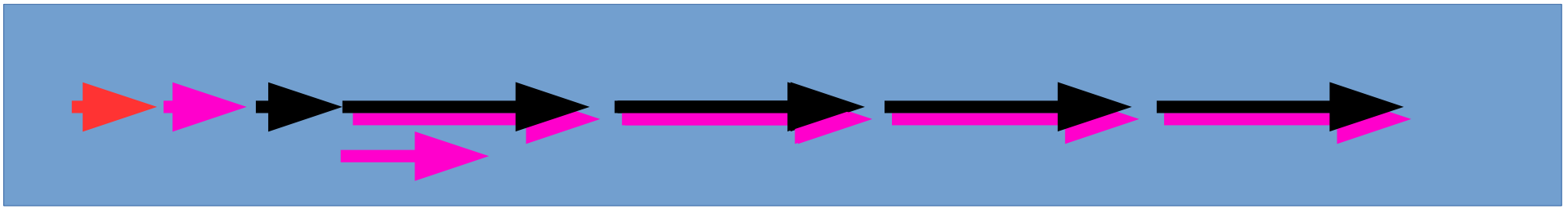


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

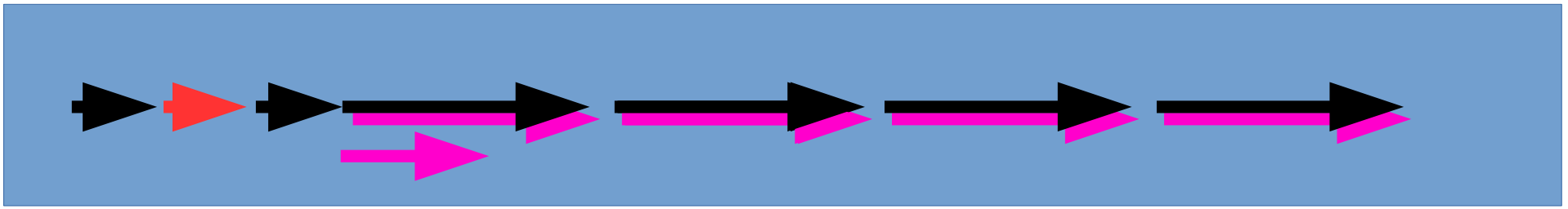


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

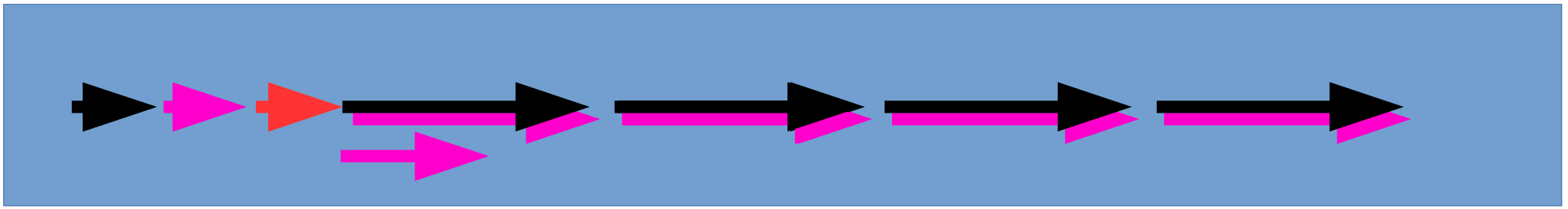


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

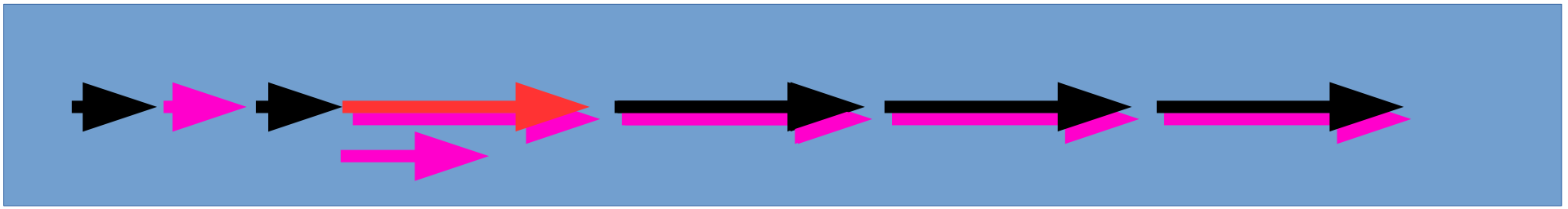


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

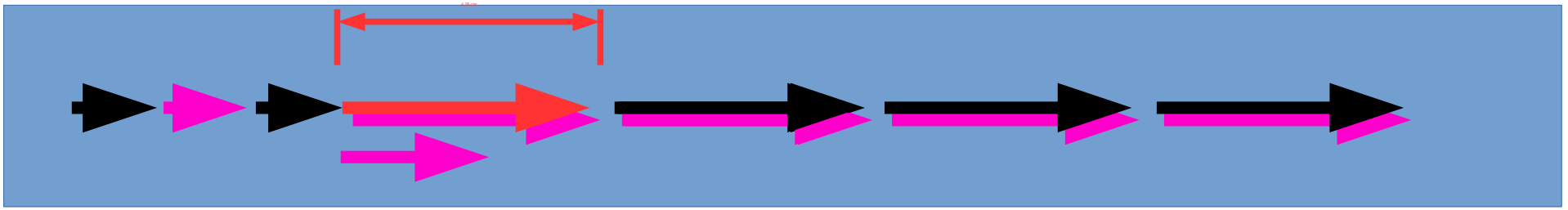


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

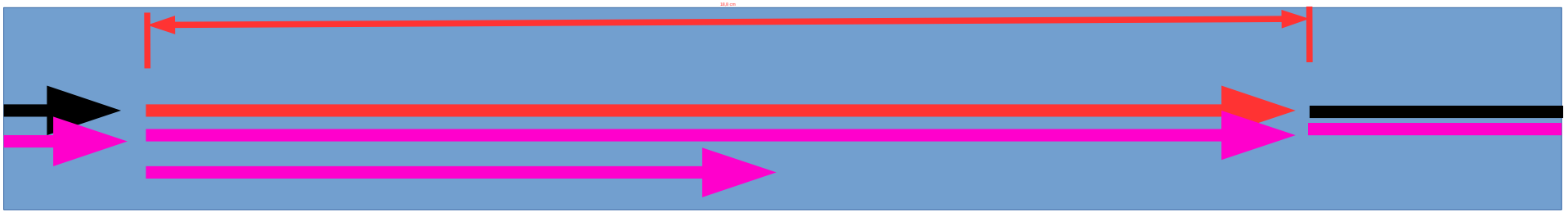


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

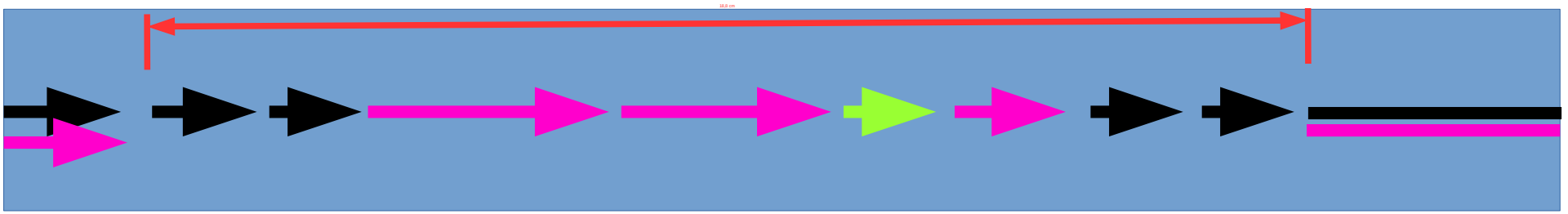


```
POE::Session->create(inline_states => {
  _start => sub {
      $poe_kernel->delay(sub {
          print „Hello World!\n“;
      } => 15);
      $request1 = syncget("https://cryptomagic.eu");
      $request2 = syncget("https://www.heise.de/");
      $request3 = syncget("https://blog.fefe.de/");
      $request4 = syncget("https://gesicherte.email/");
   }, … $poe_kernel->run();
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

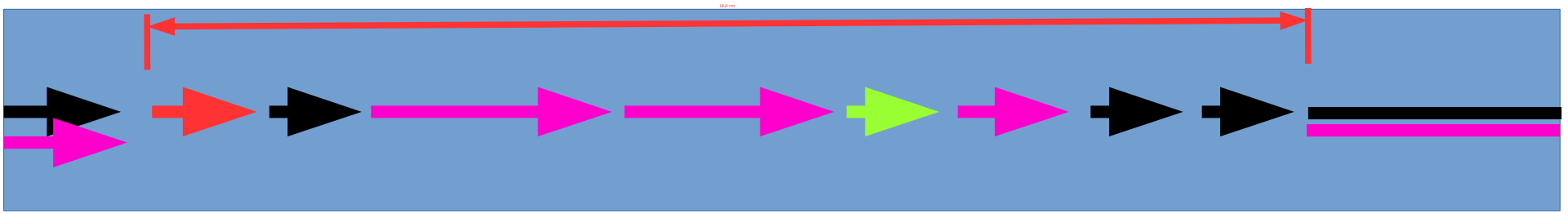


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   }
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

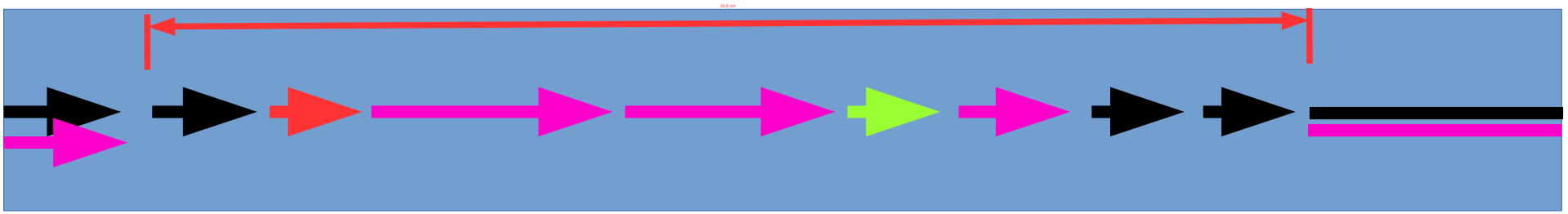


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   }
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

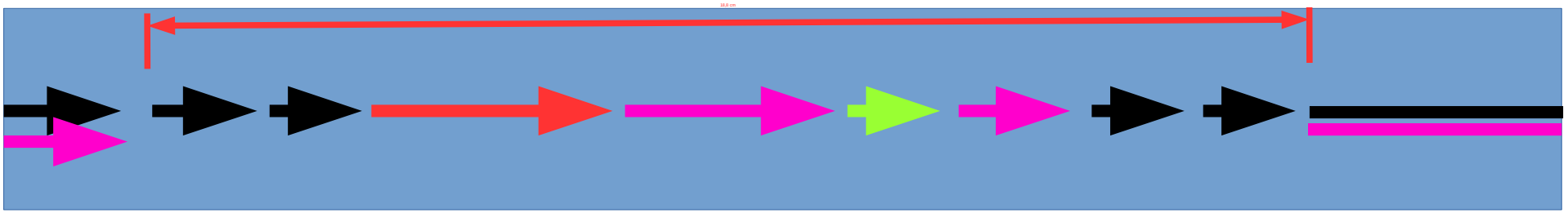


```
sub syncget {
    my $result = undef;
    get(shift, sub {
        $result = shift;
    });
    while (!defined($result)) {
        $poe_kernel->run_one_timeslice;
    };
    return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

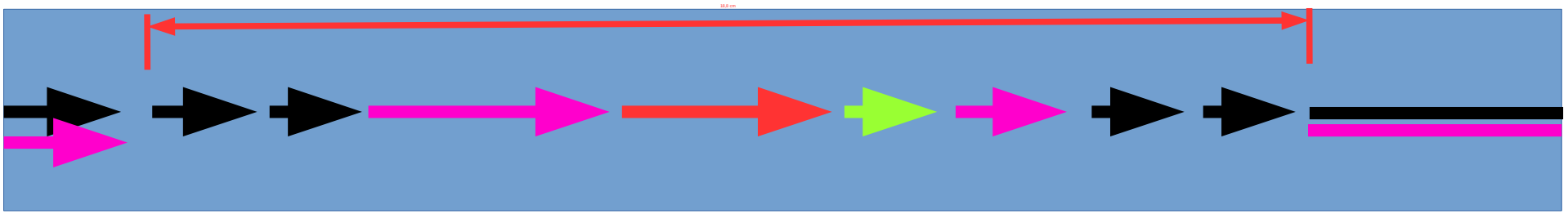


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

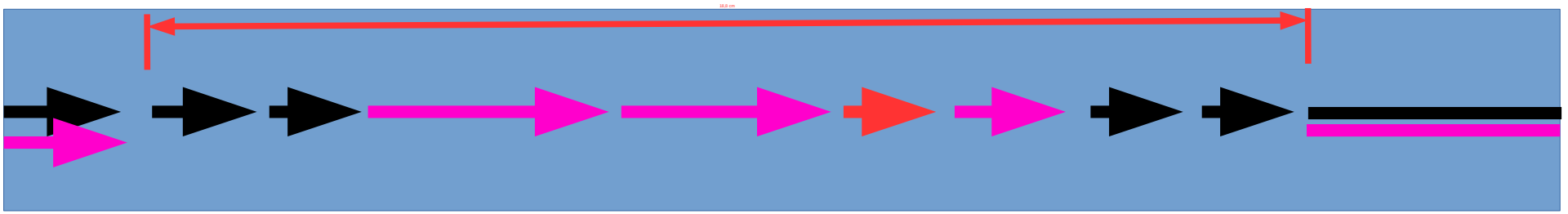


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

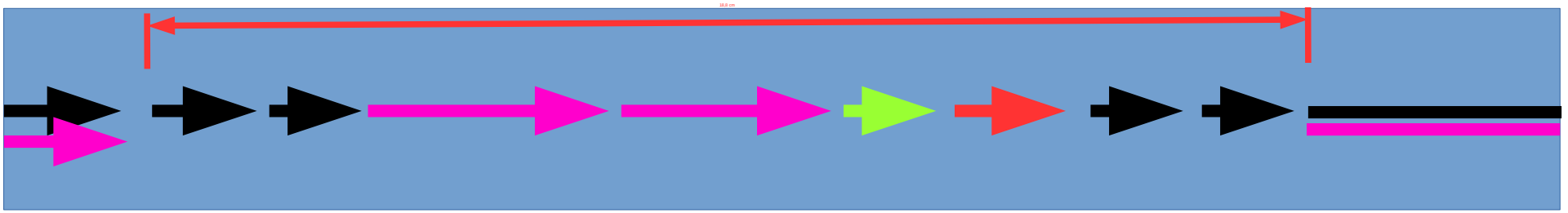


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

Prozessor 1

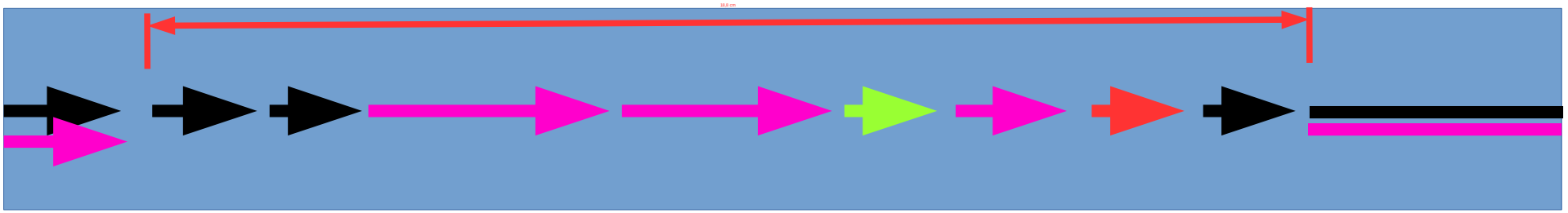


```
sub syncget {
    my $result = undef;
    get(shift, sub {
        $result = shift;
    });
    while (!defined($result)) {
        $poe_kernel->run_one_timeslice;
    };
    return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

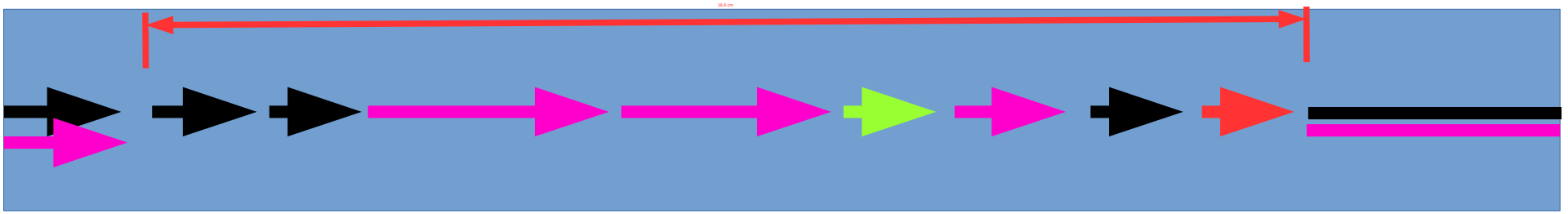
Prozessor 1



```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

POE - AsyncAwait

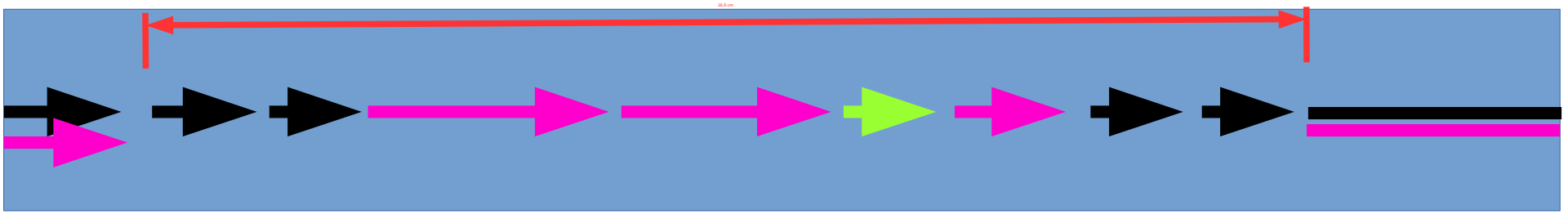
Prozessor 1



```
sub syncget {
    my $result = undef;
    get(shift, sub {
        $result = shift;
    });
    while (!defined($result)) {
        $poe_kernel->run_one_timeslice;
    };
    return $result;
});
```

POE - AsyncAwait

Prozessor 1

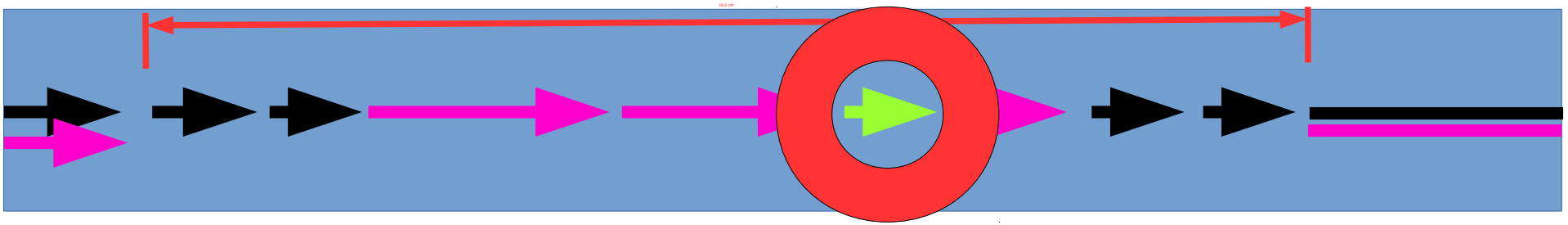


```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE - AsyncAwait

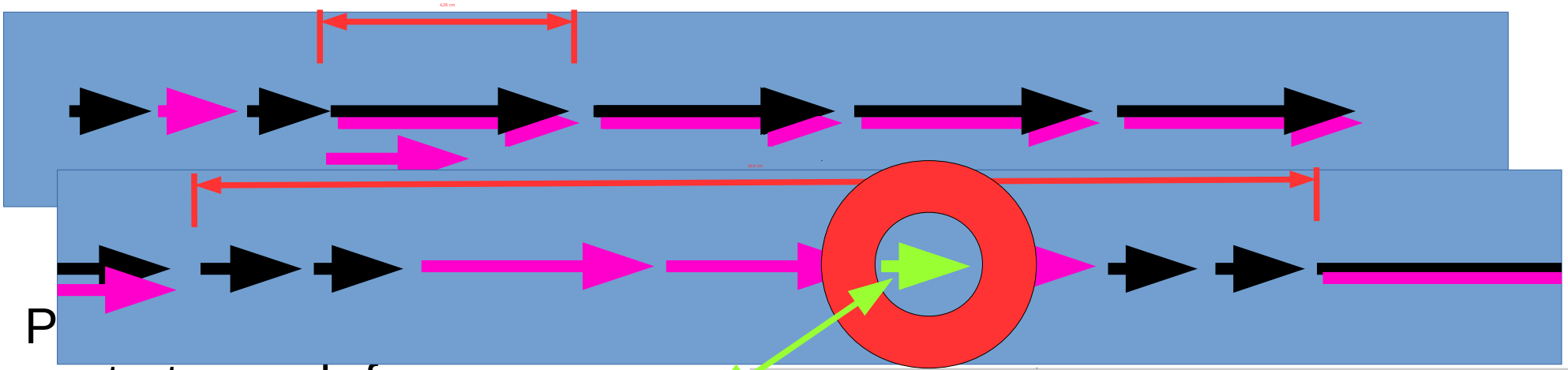
Prozessor 1



```
sub syncget {
   my $result = undef;
   get(shift, sub {
      $result = shift;
   });
   while (!defined($result)) {
      $poe_kernel->run_one_timeslice;
   };
   return $result;
});
```

POE - AsyncAwait

Prozessor 1



P

```
_start => sub {
    $poe_kernel->delay(sub {
        print „Hello World!\n“;
    } => 15);
    $request1 = syncget("https://cryptomagic.eu");
    $request2 = syncget("https://www.heise.de/");
    $request3 = syncget("https://blog.fefe.de/");
    $request4 = syncget("https://gesicherte.email/");
}, ...
```

```
while (!defined($result)) {
    $poe_kernel->run_one_timeslice;
};
```

CRYPTO MAGIC einfach. sicher. verbunden.

POE – Feeling: Nachrichten

HTTP Client – POE::Component::Client::HTTP

```
use HTTP::Request::Common qw(GET);
use POE qw(Component::Client::HTTP);

POE::Component::Client::HTTP->spawn(Alias => 'ua');

POE::Session->create(
   inline_states => {
      "_start" => sub {
          $poe_kernel->post("ua" => "request", "got_response",
               GET "http://www.cryptomagic.eu/");
      },
      "got_response" => sub {
          my ($request, $response) = @_[ARG0, ARG1];
          print $response->[0]->code()."\n";
      }
   }
);

POE::Kernel->run();
```

POE – Feeling: Sessions

```
for ( 1 .. 2 ) {
  POE::Session->create(
    inline_states => {
      _start => sub {
         print "Session ", $_[SESSION]->ID,
           " has started.\n";
         $_[HEAP]->{count} = 0;
         $_[KERNEL]->yield("count");
      },
      _stop => sub {
        print "Session ", $_[SESSION]->ID,
          " has stopped.\n";
      },
      count => sub {
        my ( $kernel, $heap ) = @_[ KERNEL, HEAP ];
        my $session_id = $_[SESSION]->ID;
        print "Session $session_id has counted to ".
          ++$heap->{count}."\n";
        $kernel->delay("count" => 1)
          if $heap->{count} < 10;
      }
    }
  );
}
```

```
Session 1 has started.
Session 2 has started.
Session 1 has counted to 1
Session 2 has counted to 1
Session 1 has counted to 2
Session 2 has counted to 2
Session 1 has counted to 3
Session 2 has counted to 3
Session 1 has counted to 4
Session 2 has counted to 4
Session 1 has counted to 5
Session 2 has counted to 5
Session 1 has counted to 6
Session 2 has counted to 6
Session 1 has counted to 7
Session 2 has counted to 7
Session 1 has counted to 8
Session 2 has counted to 8
Session 1 has counted to 9
Session 2 has counted to 9
Session 1 has counted to 10
Session 2 has counted to 10
Session 2 has stopped.
Session 1 has stopped.
```

POE – Feeling: Sessions

```
for ( 1 .. 2 ) {
  POE::Session->create(
    inline_states => {
      _start => sub {
         print "Session ", $_[SESSION]->ID,
           " has started.\n";
         $_[HEAP]->{count} = 0;
         $_[KERNEL]->yield("count");
      },
      _stop => sub {
        print "Session ", $_[SESSION]->ID,
          " has stopped.\n";
      },
      count => sub {
        my ( $kernel, $heap ) = @_[ KERNEL, HEAP ];
        my $session_id = $_[SESSION]->ID;
        print "Session $session_id has counted to ".
          ++$heap->{count}."\n";
        $kernel->delay("count" => 1)
          if $heap->{count} < 10;
      }
    }
  );
}
```

```
Session 1 has started.
Session 2 has started.
Session 1 has counted to 1
Session 2 has counted to 1
200
Session 1 has counted to 2
Session 2 has counted to 2
Session 1 has counted to 3
Session 2 has counted to 3
Session 1 has counted to 4
Session 2 has counted to 4
Session 1 has counted to 5
Session 2 has counted to 5
Session 1 has counted to 6
Session 2 has counted to 6
Session 1 has counted to 7
Session 2 has counted to 7
Session 1 has counted to 8
Session 2 has counted to 8
Session 1 has counted to 9
Session 2 has counted to 9
Session 1 has counted to 10
Session 2 has counted to 10
Session 2 has stopped.
Session 1 has stopped.
```

Übersicht

Synchron	Perl-Multithreading	Fork	Event-Loop
Einfache Wartbarkeit und Debugbarkeit	Multi-CPU Ausnutzung Vermeidung von Wartezeiten durch Parallelität	Multi-CPU Ausnutzung Vermeidung von Wartezeiten durch Parallelität	Vermeidung von Wartezeiten durch Asynchronität
	Speicher-Locking notwendig, Deadlocks möglich Bibliotheken müssen Thread-safe sein chdir/chroot/umask/… Synchronisation Kopieraufwand	Datenzusammen-führungsaufwand oder Shared-Memory Kein Windows	Kürzere Laufzeit wenn Nebenläufigkeit möglich ist und umgesetzt wurde Gut kombinierbar mit Fork und (Perl-)Multithreading

POE

Filedescriptorbasierte Schnittstelle:

→ $poe_kernel->select_read($socket, „event“);
Einwege-Sockets, UDP-Verbindungen

→ POE::Wheel::SocketFactory
Sockets, TCP-Verbindungen, Webserver, IRC-Server, SMTP-Server, ...
→ Filter, z.B. für SSL, HTTP, Mail, ZLIB, XML, SMTP, IRC, JSON, Freeswitch, DNS, u.v.m.
Zur Laufzeit austauschbar, z.B. für STARTTLS.

Beispiele:

→ http://poe.perl.org/?POE_Cookbook
- „tail -f“ → POE::Wheel::FollowTail
- Asynchronous MySQL → $poe_kernel->select_read

→ https://github.com/pRiVi/poe-vortrag

Sonstiges:

→ MouseX::POE

→ POE::Filter::SSL

CRYPTO MAGIC einfach. sicher. verbunden.