MTU и сокеты

```go
func main() {  Viacheslav Balashov
    a := &net.TCPAddr{
        IP:   net.IPv4zero,
        Port: 8880,
    }

    l := must.OK(net.ListenTCP( network: "tcp4", a))

    conn := must.OK(l.AcceptTCP())

    buffer := make([]byte, 10_000_000)

    read := must.OK(conn.Read(buffer))

    fmt.Println(read)
}
```

```java
public class Main {  Viacheslav Balashov *
    public static void main(String[] args) throws Exception {  Viacheslav Balashov *
        int targetPort = 8880;
        var host = "localhost";

        try (var socket = new Socket(host, targetPort); var out = socket.getOutputStream()) {
            var msg = "1".repeat( count: 10_000);
            var msgBytes = msg.getBytes();

            System.out.println(msgBytes.length);

            out.write(msgBytes);
        }
    }
}
```

```go
func main() {  Viacheslav Balashov
    a := &net.TCPAddr{
        IP:   net.IPv4zero,
        Port: 8880,
    }

    l := must.OK(net.ListenTCP( network: "tcp4", a))

    conn := must.OK(l.AcceptTCP())

    buffer := make([]byte, 10_000_000)

    read := must.OK(conn.Read(buffer))

    fmt.Println(read)
}
```

```
/Library/Java/JavaVirtualMachines/
10000

Process finished with exit code 0
```

```
<4 go setup calls>
10000

Process finished with the exit code 0
```

```java
public class Main {  Viacheslav Balashov *
    public static void main(String[] args) throws Exception {  Viacheslav Balashov *
        int targetPort = 8880;
        var host = "localhost";

        try (var socket = new Socket(host, targetPort); var out = socket.getOutputStream()) {
            var msg = "1".repeat( count: 10_000);
            var msgBytes = msg.getBytes();

            System.out.println(msgBytes.length);

            out.write(msgBytes);
        }
    }
}
```

```go
func main() {  Viacheslav Balashov
    a := &net.TCPAddr{
        IP:   net.IPv4zero,
        Port: 8880,
    }

    l := must.OK(net.ListenTCP( network: "tcp4", a))

    conn := must.OK(l.AcceptTCP())

    buffer := make([]byte, 10_000_000)

    read := must.OK(conn.Read(buffer))

    fmt.Println(read)
}
```

```java
public class Main {  Viacheslav Balashov +1 *
    public static void main(String[] args) throws Exception {  Viacheslav Balashov +1 *
        int targetPort = 8880;
        var host = "192.168.50.15";

        try (var socket = new Socket(host, targetPort); var out = socket.getOutputStream()) {
            var msg = "1".repeat( count: 10_000);
            var msgBytes = msg.getBytes();

            System.out.println(msgBytes.length);

            out.write(msgBytes);
        }
    }
}
```

```go
func main() {  Viacheslav Balashov
    a := &net.TCPAddr{
        IP:   net.IPv4zero,
        Port: 8880,
    }

    l := must.OK(net.ListenTCP( network: "tcp4", a))

    conn := must.OK(l.AcceptTCP())

    buffer := make([]byte, 10_000_000)

    read := must.OK(conn.Read(buffer))

    fmt.Println(read)
}
```

```
<4 go setup calls>
2896

Process finished with the exit code 0
```

```
<4 go setup calls>
5792

Process finished with the exit code 0
```

```
/Library/Java/JavaVirtualMachines/
10000

Process finished with exit code 0
```

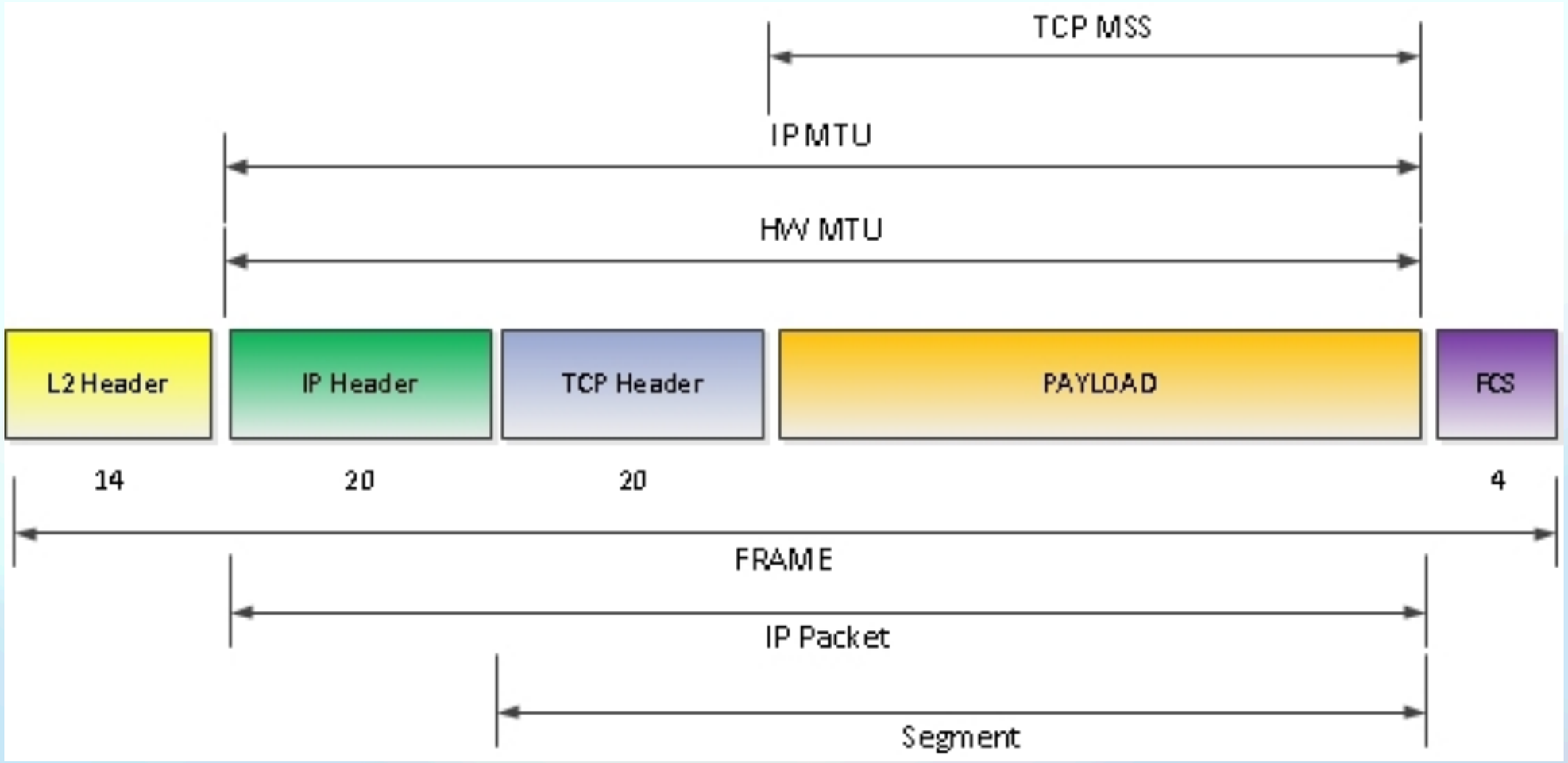
```java
public class Main {  Viacheslav Balashov +1 *
    public static void main(String[] args) throws Exception {  Viacheslav Balashov +1 *
        int targetPort = 8880;
        var host = "192.168.50.15";

        try (var socket = new Socket(host, targetPort); var out = socket.getOutputStream()) {
            var msg = "1".repeat( count: 10_000);
            var msgBytes = msg.getBytes();

            System.out.println(msgBytes.length);

            out.write(msgBytes);
        }
    }
}
```

Что такое MTU?

Maximum transmission unit (MTU) — **это максимальный объём данных**, **который может быть передан протоколом за одну итерацию**.

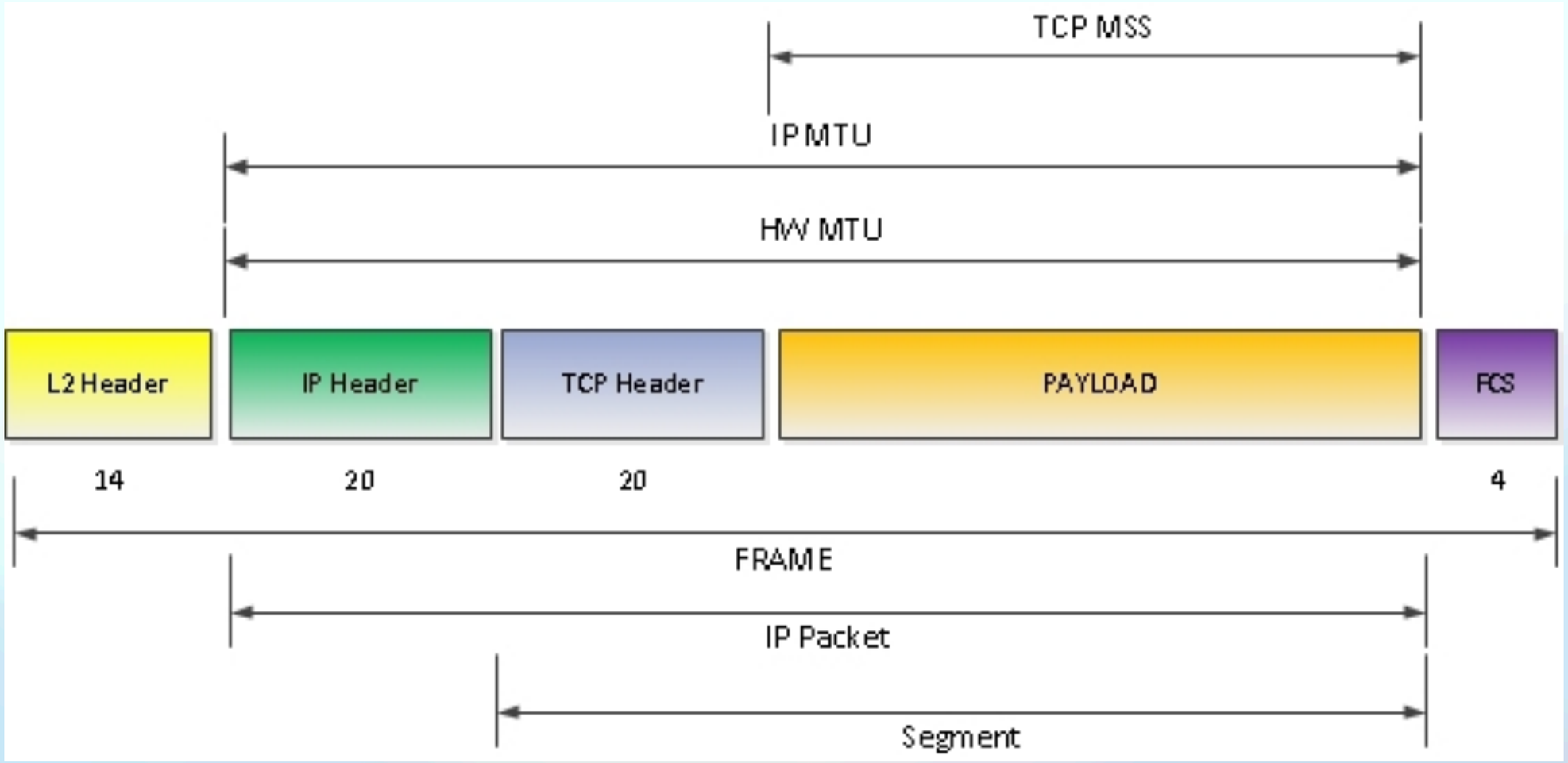
MTU?

максимальный размер поля данных IP-пакета



MTU?

максимальный размер поля данных IP-пакета



Какие цифры?

Посмотреть в Linux можно с помощью ip a

В среднем для localhost — максимальное значение (65536 байт)

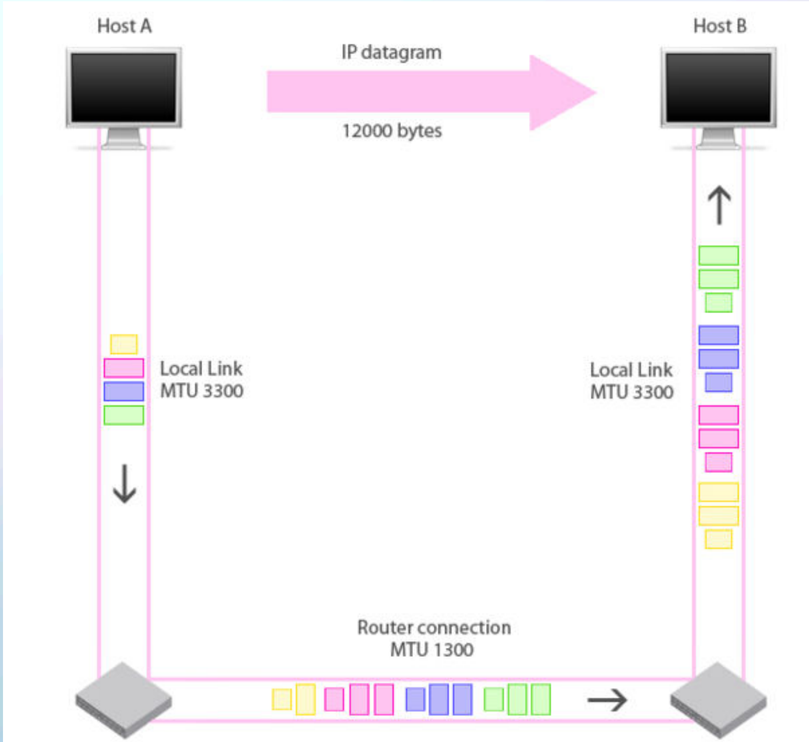
Для внешних сетей — 1500 байт

```
(deck@steamdeck mini-client)$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host proto kernel_lo
       valid_lft forever preferred_lft forever
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 14:80:cc:94:92:7b brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.157/24 brd 192.168.50.255 scope global dynamic noprefixroute wlan0
       valid_lft 83112sec preferred_lft 83112sec
    inet6 fe80::1680:ccff:fe94:927b/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
6: enp4s0f3u1u4: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 00:e0:4c:21:80:c8 brd ff:ff:ff:ff:ff:ff
    altname enx00e04c2180c8
```

MTU

Почему такой размер?

- Большие пакеты дольше проверяются
- При некоторых конфигурациях сетей может произойти коллизия данных
- Размер буфера роутеров ограничен
- Потеря или повреждение маленького пакета не критично



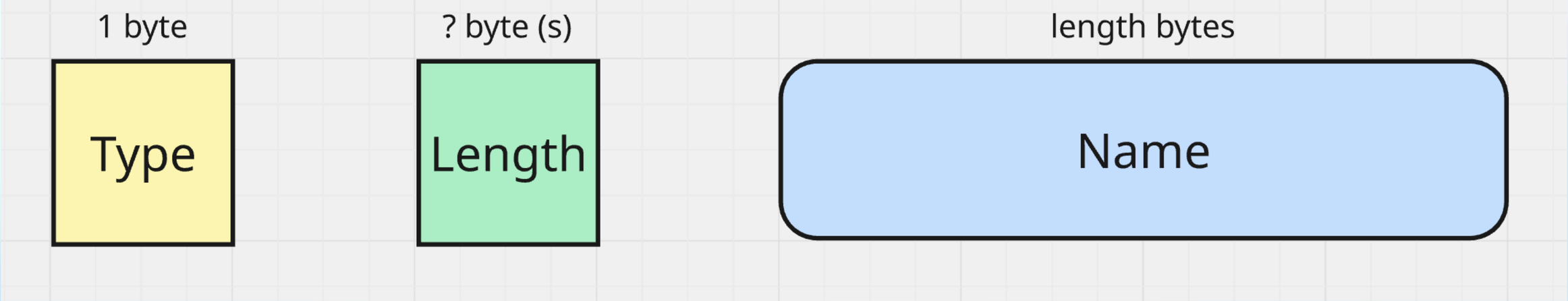
И что с МТУ делать?

Ничего)

Знать что он есть и что из-за него данные приходят по-чуть чуть

По протоколу 1 задачи

Какие проблемы?

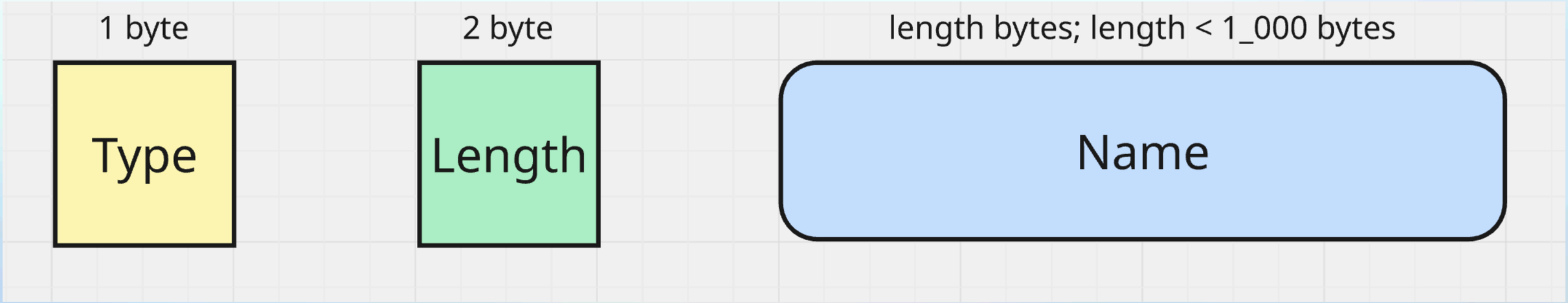


По протоколу 1 задачи

Уникальность имен?

Длина имен?

Таймауты?

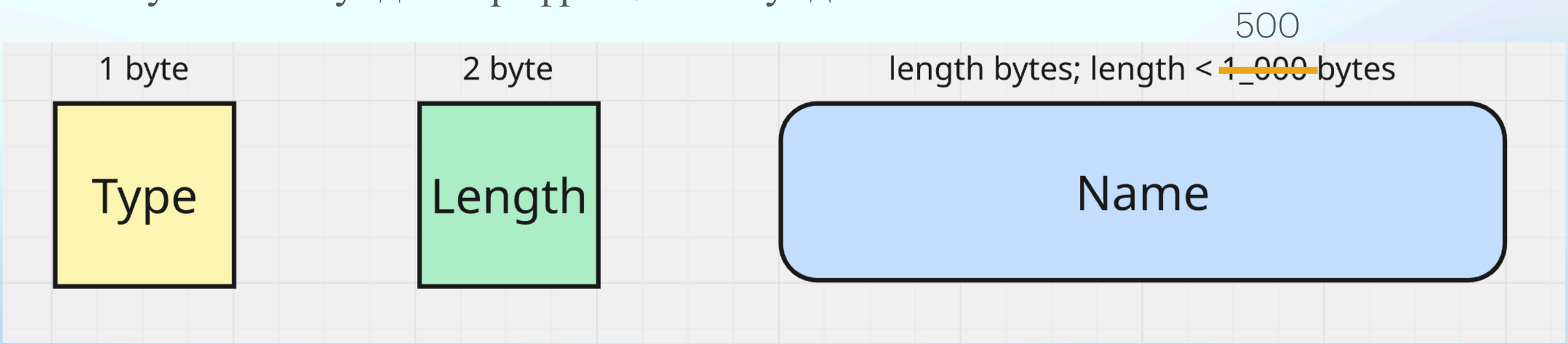


По протоколу 1 задачи

Уникальность имен — добавляем информация об адресе и порте

Длина имен — не больше 500 байт

Таймаут — 3 секунды на рефреш, 10 секунд на отвал



Type 0 - появился в сети

Type 1 - покидает сеть