DESCOBERTA DE SERVIÇOS VIA DNS

DNSDISCO

SERVIÇO

SERVIÇO 2

SERVIÇO 1

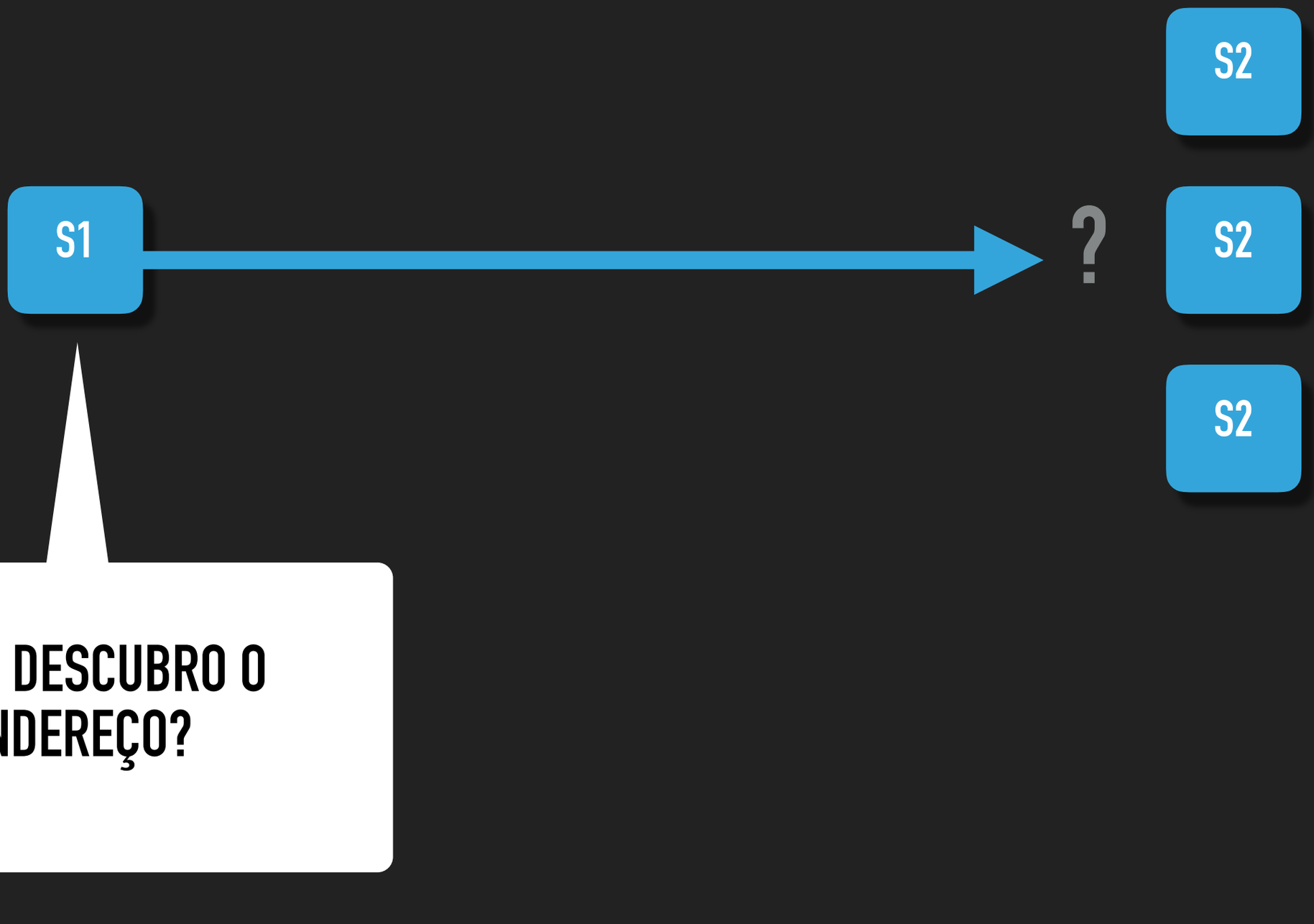
?

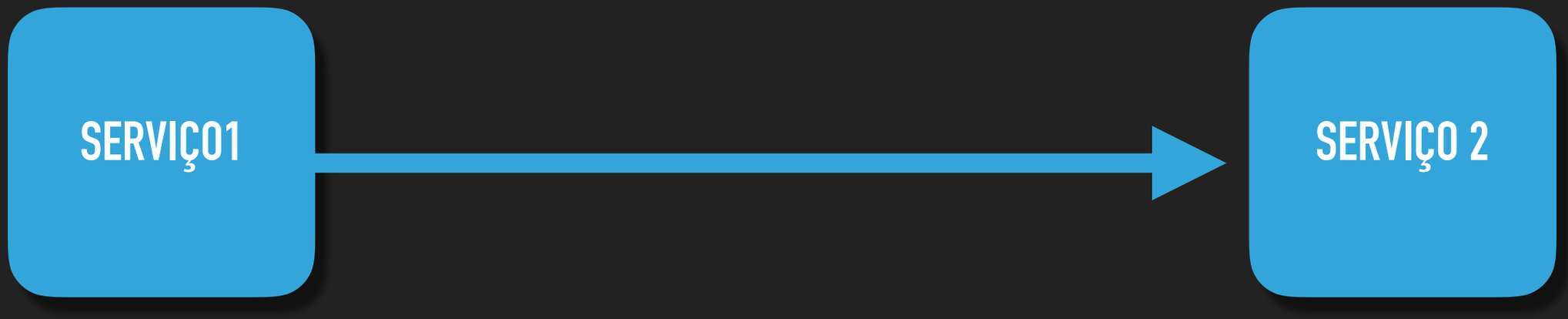
SERVIÇO 2

SERVIÇO 2

COMO DESCUBRO O ENDEREÇO?

(MICRO)?SERVIÇO





O QUE AS SOLUÇÕES DEVERIAM ME OFERECER?

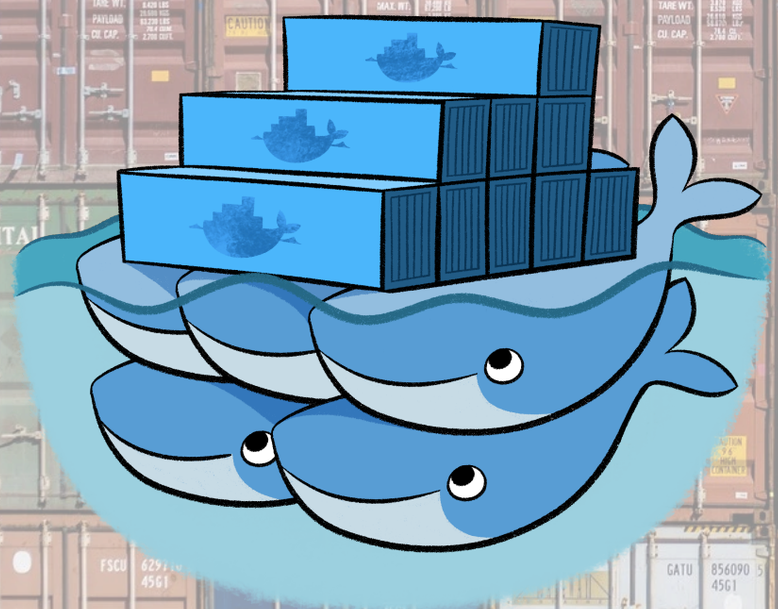
- Resiliência
- Desempenho
- Simplicidade
- Elasticidade

BALANCEADOR DE CARGA

CENTRALIZADORES DE CONFIGURAÇÃO

FERRAMENTAS DE AUTOMAÇÃO

ORQUESTRAÇÃO DE CONTAINERS



kubernetes

```
11
12      location /papers {
13          alias /Users/tyler/works
14
15          if ($request_filename ~*
16              expires 30d;
17          }
18      }
19
20      # if you don't like seeing a
21      location = /favicon.ico { ac
22
23      # if you don't like seeing c
24      location = /robots.txt { acc
25
26      #error_page  404
27
28      # redirect server error page
29      #
local/nginx/lab.tylergaw  1,1  Plain Text
```

O CLÁSSICO ARQUIVO DE CONFIGURAÇÃO

TEMOS MAIS UMA OPÇÃO!

O BOM E VELHO DNS

DNS

exemplo.com.br. 86400 IN SOA ...

exemplo.com.br. 86400 IN NS ns1.exemplo.com.br.

exemplo.com.br. 86400 IN A 192.0.2.1

exemplo.com.br. 86400 IN MX 1 m.exemplo.com.br

DNS

NSEC3PARAM
SPF
NULL
URI
TKEY
TALINK
NSAP
UID
TA
OPENPGPKEY
NIMLOC
OPT
NSAP-PTR
NS
SOA
TSIG
NAPTR
PTR
MINFO
MF
ISDN
MD
MAILA
L64
TXT
KX
PX
DHCID
SRV
CDS
GID
NSEC
CERT
NXT
EUI64
IXFR
UINFO
NSEC3
X25
RT
A
WKS
TLSA
SSHFP
DS
MR
A6
DLV
MB
HIP
RP
CDNSKEY
SIG
KEY
LOC
AFSDB
MX
CAA
AXFR
HINFO
CSYNC
EUI48
DNSKEY
EID
MG
UNSPEC
NINFO
AAAA
SINK
ATMA
AVC
SMIMEA
APL
DNAME
LP
GPOS
L32
IPSECKEY
RKEY
NID
CNAME
MAILB
RRSIG

DNS

SRV

NSEC3PARAM
SPF
NULL
URI
TKEY
TALINK
TA
OPENPGPKEY
NSAP
UID
NSAP-PTR
NIMLOC
OPT
NS
SOA
TSIG
NAPTR
PTR
MINFO
MD
MAILA
L64
TXT
PX
ME
ISDN
CERT
KX
DHCID
CDS
GID
NSEC
NXT
EUI64
IXFR
UINFO
NSEC3
X25
RT
A
WKS
TLSA
SSHFP
DS
MR
A6
HIP
DLV
MB
RP
CDNSKEY
SIG
KEY
AFSDB
LOC
MX
CAA
AXFR
HINFO
CSYNC
DNSKEY
UNSPEC
NINFO
EUI48
EID
MG
AAAA
AVC
SINK
SMIMEA
ATMA
APL
DNAME
LP
GPOS
CNAME
L32
IPSECKEY
NID
MAILB
RKEY
RRSIG

DNS

$ORIGIN exemplo.com.br.

_https._tcp 86400 IN SRV 0 5 443 s1.servico

DNS

```
$ORIGIN exemplo.com.br.

_https._tcp 86400 IN SRV 0 5 443 s1.servico
_https._tcp 86400 IN SRV 0 7 443 s2.servico
_https._tcp 86400 IN SRV 1 9 443 s3.servico
```

DNSDISCO

SERVIÇO

DNS

SERVIÇO 1

SERVIÇO 2

SERVIÇO 2

SERVIÇO 2

CARACTERÍSTICAS

- Busca de serviços no DNS
- Health check
- Algoritmo de escolha baseado na RFC 2782
- Flexibilidade

COMO USAR?

```go
package main

import (
  "fmt"
  "github.com/rafaeljusto/dnsdisco"
)

func main() {
  target, port, err := dnsdisco.Discover("jabber", "tcp", "registro.br")
  if err != nil {
    fmt.Println(err)
    return
  }

  fmt.Printf("Target: %s\nPort: %d\n", target, port)
}
```

COMO USAR?

```go
package main

import (
    "fmt"

    "github.com/rafaeljusto/dnsdisco"
)

var discovery dnsdisco.Discovery

func init() {
    discovery = dnsdisco.NewDiscovery("jabber", "tcp", "registro.br")
}

func main() {
    if err := discovery.Refresh(); err != nil {
        fmt.Println(err)
        return
    }

    target, port := discovery.Choose()
    fmt.Printf("Target: %s\nPort: %d\n", target, port)
}
```

COMO USAR?

```
func main() {
    if err := discovery.Refresh(); err != nil {
        fmt.Println(err)
        return
    }

    finish := discovery.RefreshAsync(5 * time.Second)
    defer close(finish)

    for {
        if errs := discovery.Errors(); len(errs) > 0 {
            fmt.Println(errs)
            return
        }

        target, port := discovery.Choose()
        fmt.Printf("Target: %s\nPort: %d\n", target, port)
        time.Sleep(time.Second)
    }
}
```

FLEXIBILIDADE - HEALTH CHECKER

```
func healthChecker(target string, port uint16,    string) (bool, error) {
    r, err := http.Get(fmt.Sprintf("http://%s:%d/teste", target, port))
    return err == nil && r.StatusCode == http.StatusNoContent, err
}

func main() {
    // ...
    discovery.SetHealthChecker(dnsdisco.HealthCheckerFunc(healthChecker))
    // ...
}
```

FLEXIBILIDADE - BALANCEADOR

```
type loadBalancer struct {
   servers []*net.SRV
}

func (l *loadBalancer) ChangeServers(servers []*net.SRV) {
   l.servers = servers
}
```

```
func (l loadBalancer) LoadBalance() (target string, port uint16) {
   if len(l.servers) > 0 {
      target = l.servers[0].Target
      port = l.servers[0].Port
   }
   return
}

func main() {
   // ...
   discovery.SetLoadBalancer(new(loadBalancer))
   // ...
}
```

FLEXIBILIDADE - OBTENTOR

```
func retriever(service, proto, name string) ([]*net.SRV, error) {
   name = strings.TrimRight(name, ".")
   z := fmt.Sprintf("_%s._%s.%s.", service, proto, name)

   var request dns.Msg
   request.SetQuestion(z, dns.TypeSRV)
   request.RecursionDesired = true

   response, err := dns.Exchange(&request, "8.8.8.8:53")
   if err != nil {
      return nil, err
   }

   var servers []*net.SRV
   // conversão de response.Answers para []*net.SRV
   return servers, nil
}

func main() {
   // ...
   discovery.SetRetriever(dnsdisco.RetrieverFunc(retriever))
   // ...
}
```

github.com/rafaeljusto/dnsdisco

EXEMPLO https://gist.github.com/rafaeljusto/ec64b408ec240df52f77