RNP

MCT

Hierarquia Nacional de Proxies

Implantação e Interoperação do Squid com o IBM Network Dispatcher

Novembro de 1998

Márcio Anthony Gonçalves Cesário
Wagner Meira Júnior
Laboratório de Configuração e Testes - RNP

Índice:

Parte A: Introdução à Hierarquia Nacional da RNP

- Cache W W W: Introdução e Motivação
- Evolução e Tecnologias utilizadas
- Proxies
- Estratégias de cooperação
- Hierarquia da RNP
- Equipamentos de Hardware (Squid Server)

Cache WWW: Introdução e Motivação

WWW: situação atual e tendência futura

Fatos:

- Maior e mais acessado recurso da Internet em apenas 5 anos
- Apresenta crescimento exponencial:
  - Tráfego WWW dobra a cada 100 dias (1998)
  - 70 % dos pacotes são HTTP (1997)
- Crescimento não planejado, motivado pelo baixo custo de acesso e comodidade.

Parte A: Introdução à Hierarquia Nacional da RNP

Tendência:

- Novas tecnologias para clientes (e.g., modem a cabo) apenas agravam a situação.
- Demanda cada vez maior por uma solução escalável, uma vez que a população cresce mais rápido que a banda passante disponível pelo baixo custo de acesso e comodidade.

Parte A: Introdução à Hierarquia Nacional da RNP

Características do Tráfego WWW:

- alta variabilidade nos tamanhos dos objetos e no tempo para acessá-los
- alta dinamicidade dos padrões de acesso (temporal e física)
- comportamento migratório
- interesse momentâneo
- documento é transferido mais de uma vez (13% - 1992)

Caches replicam dados acessados freqüentemente, explorando a localidade temporal dos documentos.

Não é um conceito novo na Internet (e.g., DNS).

Cache: Características desejáveis

- Transparência
- Escalabilidade
- Facilidade de instalação
- Tolerância à falhas
- Cooperação com servidores de conteúdo

Cache: Problemas

- Balanceamento de carga
- Coerência entre servidores e caches
- Computação e comunicação adicional
- Replicação de objetos

Parte A: Introdução à Hierarquia Nacional da RNP

Evolução e Tecnologias utilizadas

- 1a geração: baseados em proxies
  - CERN
  - Squid
- 2a geração: redirecionamento de requisições
  - Cisco Cache Engine
  - Cache Flow
- 3a geração: monitoração da rede
  - Dynacache

Proxies

Proxies como Cache Servers

Características:

- Necessidade de configuração dos clientes
- Utilização de hardware comum
- Escalabilidade demanda cooperação

Proxies

Web Proxy Cache Server

Funcionamento:

- Proxy recebe requisições HTTP de clientes
- Proxy repassa requisições aos servidores, de acordo com os protocolos suportados
- Servidor responde ao proxy
- Proxy responde ao cliente e, opcionalmente, armazena localmente uma cópia do documento (cache)

Estratégias de cooperação

Como compartilhar informações entre caches

Questões:

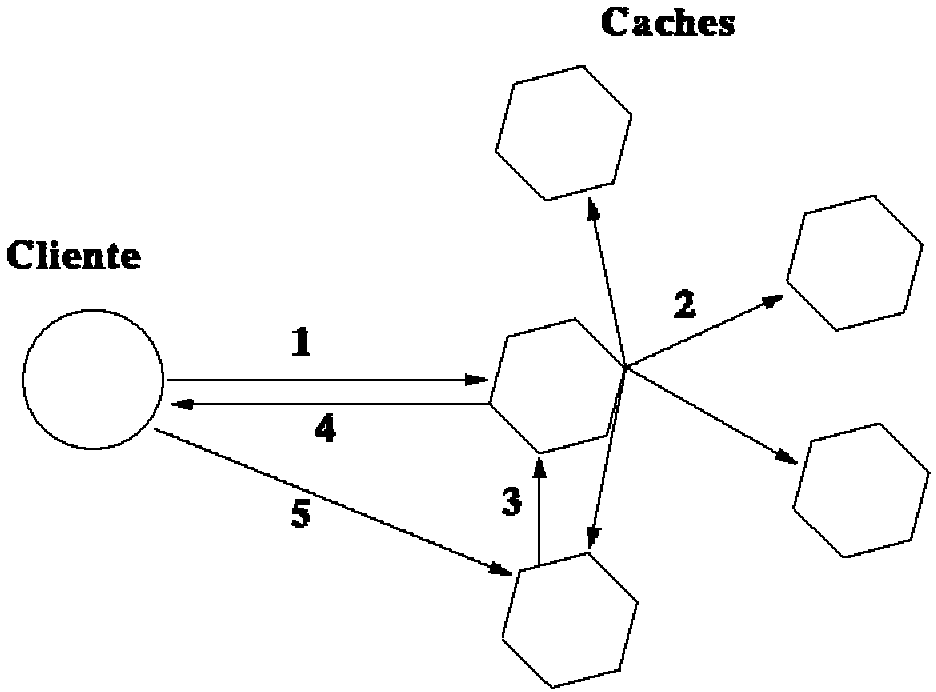
- Quanto custa cooperar?
- Qual a redução de latência?
- Qual a taxa de acerto?

Cache sob demanda:

- Cooperação informativa
- Cooperação hierárquica
- Cooperação por diretório
- Cooperação por resumos

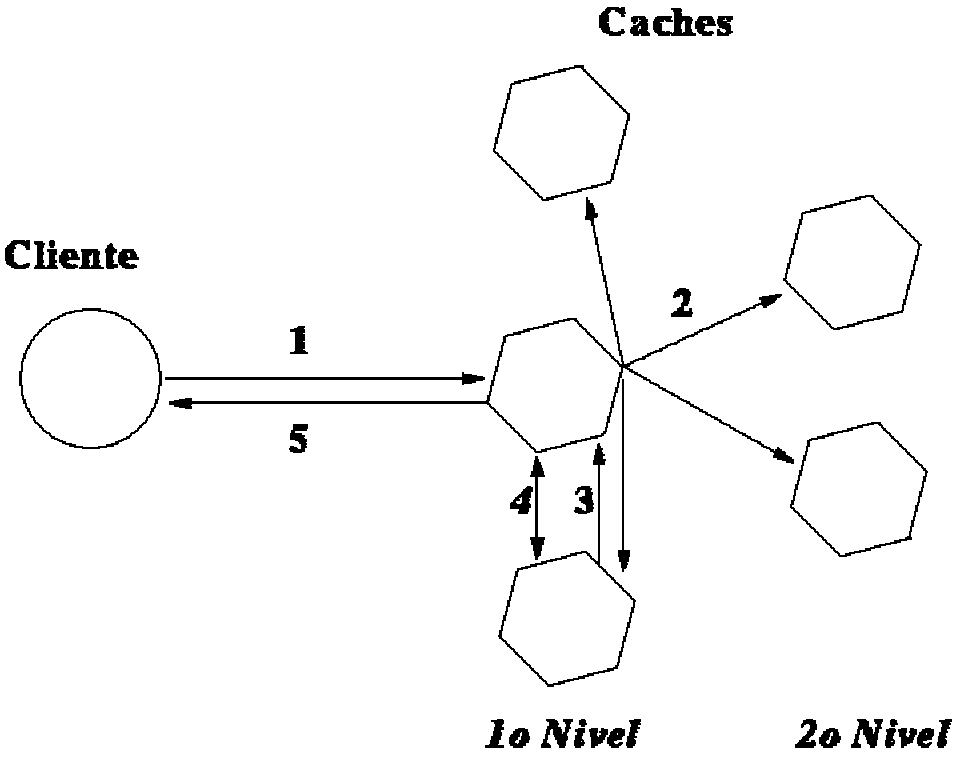
Parte A: Introdução à Hierarquia Nacional da RNP

Cooperação Informativa



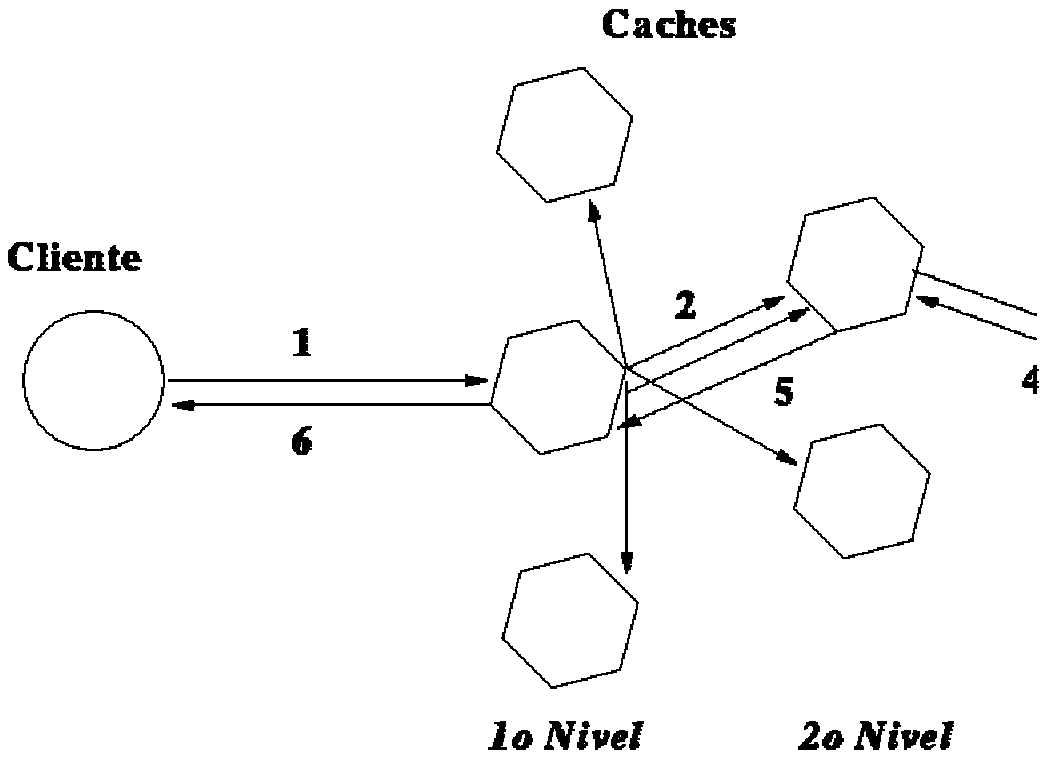
Parte A: Introdução à Hierarquia Nacional da RNP

Cooperação Hierárquica



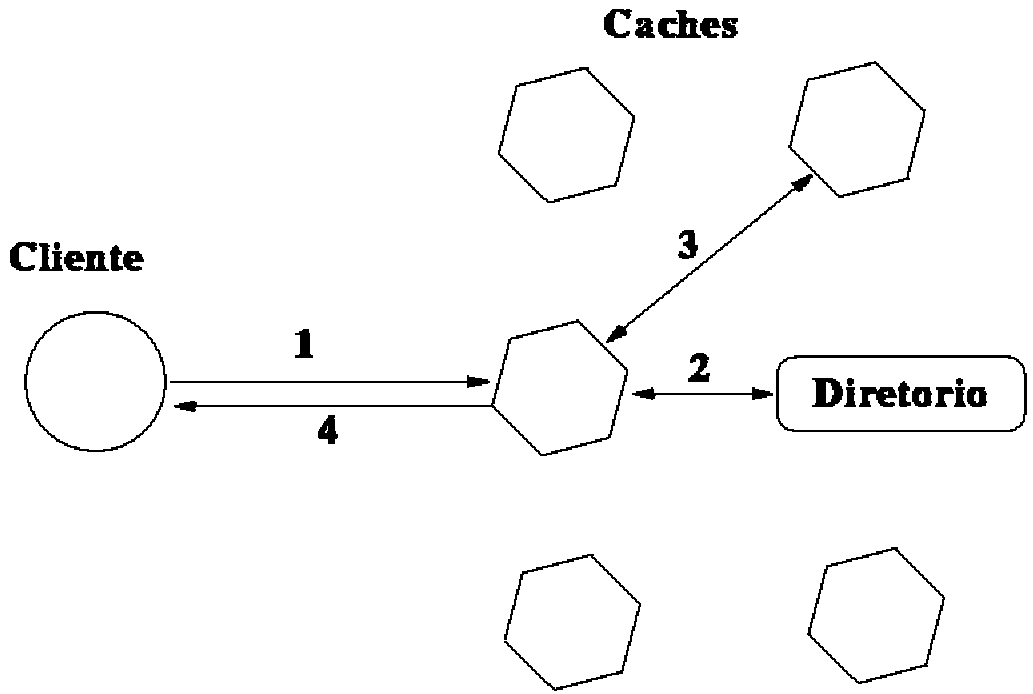
Parte A: Introdução à Hierarquia Nacional da RNP

Cooperação Hierárquica



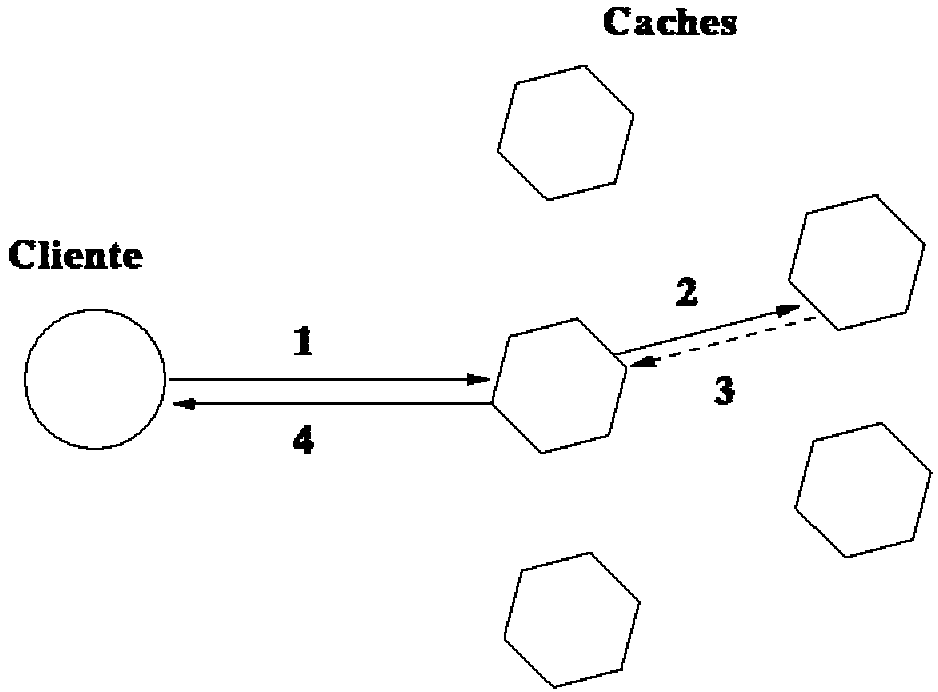
Parte A: Introdução à
Hierarquia Nacional da RNP

Cooperação por Diretório



Parte A: Introdução à Hierarquia Nacional da RNP

Cooperação por Resumos



Outras estratégias de cooperação

- Prefetching
- Replicação voluntária
- Server push
- Client pull
- Geographical push caching

Hierarquia da RNP

O que é hierarquia ?

- Servidores proxy cache se comunicam através de um protocolo de aplicação para que um deles possa buscar objetos armazenados em outro
- Relações Hierárquicas:
  - Sibling (irmandade): servidor proxy busca objeto de outro servidor somente se este possuir o objeto “cacheado” localmente
  - Parent (paternidade): servidor filho busca objeto do servidor pai mesmo no caso deste não possuir o objeto (neste caso, ele deverá conseguir o objeto de alguma forma e repassar ao filho)

Hierarquia da RNP

Formação da Hierarquia

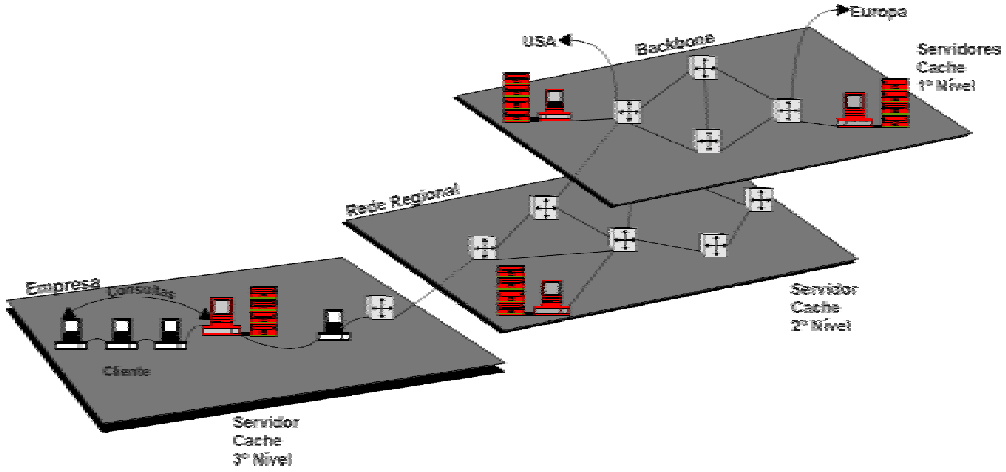
Divisão em três níveis

Nível 1: responsável apenas por responder requisições de outros servidores proxy. É formado pelas máquinas mais robustas, e estará localizado em dois pontos: RJ e DF.

Nível 2: responsável por atender requisições de usuários finais e de proxies nível 3. Localizado nos PoPs: DF, RJ, MG, SP, RS, PE, CE, PR, SC, BA e PB

Nível 3: formado por máquinas de menor porte, é responsável, nos outros PoPs (AL, RN, AM, PA, SE, MA, PI, ES, MT, MS, GO, TO, RO, RR, AP e AC), por atender apenas a requisições de usuários finais.

Parte A: Introdução à Hierarquia Nacional da RNP



- Relação de paternidade (parent): Requisições não resolvidas num nível são encaminhadas ao nível superior
- Relação de irmandade (sibling): Servidores proxy de mesmo nível se comunicam para trocar informações (objetos) entre si.

Equipamentos de Hardware (Squid Server)

Nível 1

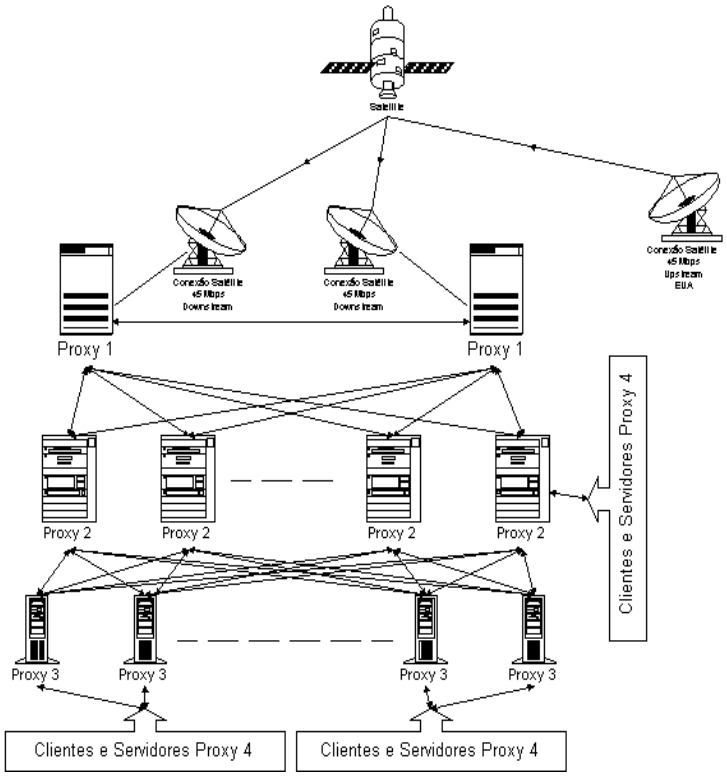
IBM SP2 (8 nós) com 300 Gb disco

Nível 2

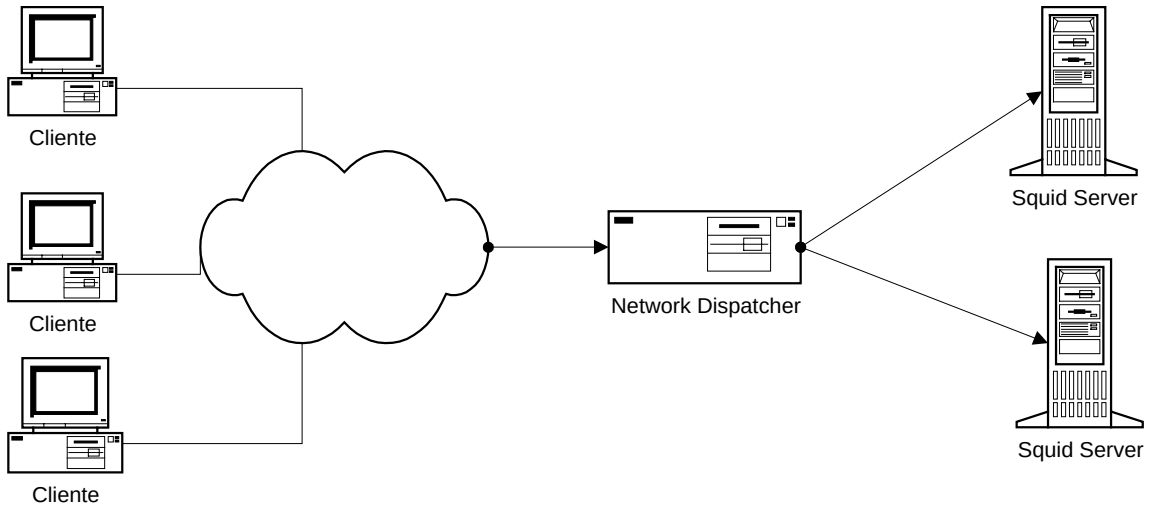
2 x IBM R50 com 100 Gb disco

Nível 3

1 x IBM R50 com 50 Gb disco



Parte A: Introdução à Hierarquia Nacional da RNP

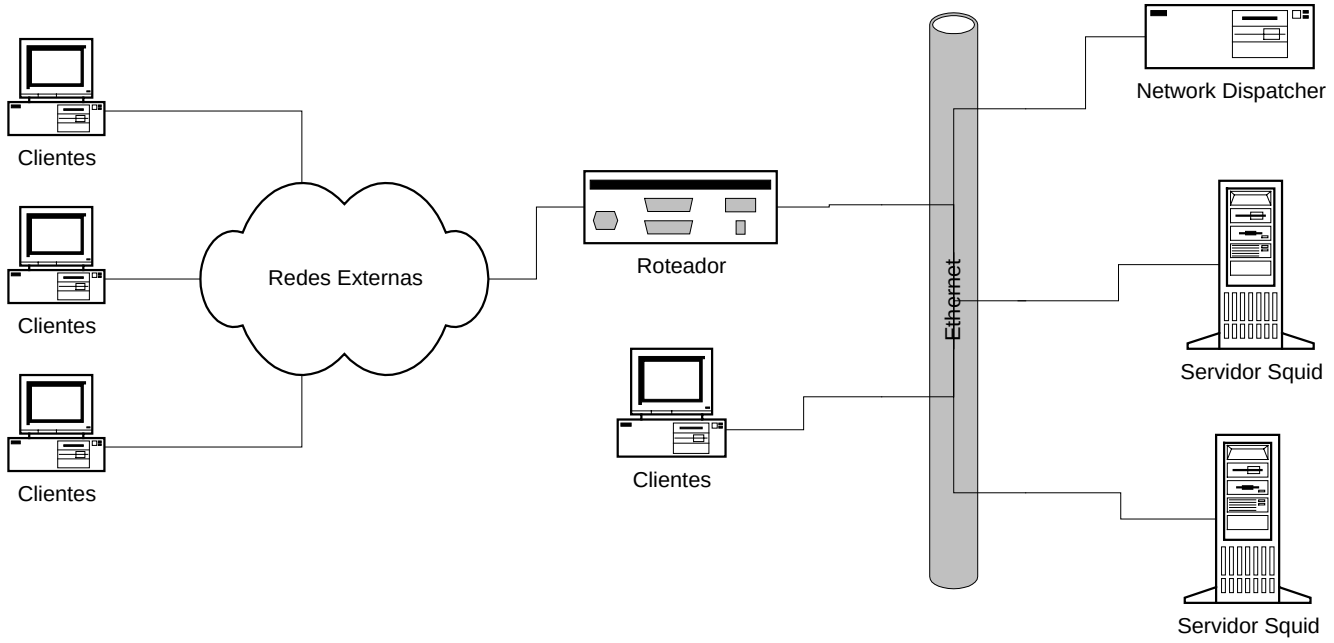


Organização lógica dos servidores

Nos PoPs onde existirem mais de um servidor (níveis 1 e 2), haverá uma outra máquina (dispatcher) que se encarregará de distribuir as requisições entre esses servidores, levando-se em consideração a disponibilidade e carga de cada um.

Parte A: Introdução à Hierarquia Nacional da RNP

Organização física do Network Dispatcher e dos servidores squid



Parte B: Instalação e Configuração do Squid

- Atuais versões
- Instalação do Squid
- Configuração do Squid
- Configuração de Hierarquias
- Referências

Atuais versões

Versões do Squid em distribuição atualmente

Squid 1.1.x

Utilizada há quase dois anos, é a segunda geração do software (a primeira é a 1.0.x). Provê alguns recursos avançados de configuração de hierarquias, mas sua tendência é ser inteiramente substituída pela versão 2.x devido aos grandes avanços em áreas como gerenciamento de memória, comunicação entre servidores, entre outros.

Squid 2.x

Primeiro *release* lançado em outubro/1998, apresenta avanços como: suporte a HTTP 1.1 (*persistent connections*), I/O de disco assíncrono (usando *pthreads*), melhor utilização de memória, uso de cache digest para comunicação entre servidores, gerenciamento via snmp, mensagens de erro customizáveis, entre outros.

Instalação do Squid

Preparação do Sistema Operacional

Criação de usuário

O squid deve ser executado por um usuário comum (ex: *user*: **nobody**, *group*: **nogroup**). O mais recomendado é a criação de um usuário específico para essa finalidade (ex: *user*: **squid**, *group*: **squid**)

Configuração do Sistema Operacional

Deve-se ficar atento para qualquer limitação do sistema operacional que possa impedir o usuário **squid** de realizar suas tarefas. Entre as limitações mais comumente encontradas estão: número máximo de conexões simultâneas, número de processos simultâneos, quantidade de memória alocada por processo, etc.

Instalação do Squid

Compilação

Geração dos *Makefiles*

Parâmetros passados ao *configure* (programa que gera os *Makefiles*)

--enable-async-io

--enable-cache-digests

--enable-icmp

--enable-snmp

--enable-err-language=Portuguese

--prefix=/squid_path

Resultado da instalação (após *make all; make install*)

$home/bin/RunAccel : script utilizado para iniciar o squid no modo *accelerator*

$home/bin/RunCache : script utilizado para iniciar o squid no modo *cache server*

$home/bin/cachemgr.cgi : cgi usado para coletar estatísticas geradas pelo squid (deve ser instalado em um web server)

$home/bin/client : cliente HTTP (*URL retriever*)

$home/bin/dnsserver : usado pelo squid resolver nomes (fqdn)

$home/bin/squid : squid propriamente dito

$home/bin/unlinkd : usado pelo para apagar arquivos sob demanda

Resultado da instalação

$home/etc/errors : diretório com os arquivos (html) de mensagens de erros

$home/etc/icons : ícones usados pelo squid para listagens de diretórios

$home/etc/mib.txt : MIB (snmp) usada para monitorar o squid

$home/etc/mime.conf : associa extensões de arquivos a tipos *mime*

$home/etc/squid.conf : arquivo de configuração do squid (controle de acesso, hierarquia, etc)

$home/logs/ : diretório onde serão armazenados os logs de acesso, erro, etc.

Configuração do Squid

squid.conf

Configuração do servidor é realizada através do *squid.conf*

- Quantidade de memória destinada ao armazenamento de objetos
- Tamanho do disco usado para cache
- Timeout de conexões
- Opções de segurança como: permissões de acesso, logs e ident
- Personalização das mensagens de erro
- Configuração do agente snmp (comunidade, permissões etc)
- Configuração de hierarquia (quem são os pais e irmãos)

squid.conf

Configuração de objetos que são sempre buscados diretamente da fonte

hierarchy_stoplist *URL_substring (...)*

URL_substring: substring que, quando encontrada nula URL, faz com ela seja buscada diretamente da origem, sem passar pelos vizinhos da hierarquia.

Exemplo: para evitar de requisitar para os vizinhos algumas páginas geradas dinamicamente, use:

hierarchy_stoplist *cgi-bin ?*

squid.conf

Configuração da quantidade de memória (RAM) usada para objetos

cache_mem *bytes*

bytes: quantidade de memória reservada para armazenar objetos em trânsito, *hot-cache* e *negative cache*

Memória para hot-cache = quanto mais melhor. Na prática, entretanto, deve-se respeitar os limites do sistema, evitando assim o uso de swap. Geralmente calcula-se:

cache_mem = Total_RAM - (squid_meta_data + outros_processos)

squid.conf

Tamanho máximo dos objetos gravados em disco

maximum_object_size *bytes*

bytes: tamanho máximo do objeto que será armazenado no cache

maximum_object_size grande = maior byte Hit Ratio

maximum_object_size pequeno = menor ganho em largura de banda, mas possível melhora na velocidade (tempo de resposta)

Na prática, recomenda-se valores de entre 4 e 16 Mb, dependendo do papel do proxy na hierarquia e da disponibilidade de disco.

squid.conf

Arquivos de log:

cache_access_log *file*
Registra todas as requisições recebidas (hora, tamanho, tempo, hit/miss etc)

cache_log *file*
Informações sobre o estado do servidor, mensagens de erro, etc.

cache_store_log *file*
Registra a entrada e saída de objetos no cache

cache_swap_log *file*
Usado pelo squid para gravar os meta-dados dos objetos gravados no disco. Esses dados são usados sempre que o squid é iniciado.

squid.conf

Diretório usado para armazenar os objetos do cache:

cache_dir *directory Mbytes level1 level2*
cache_dir *directory Mbytes level1 level2*
(...)

directory: diretório do disco onde ficarão os objetos do cache.
Mbytes: espaço máximo que pode ser ocupado por esse arquivos
level1: número de sub-diretórios que serão criados dentro de *directory*
level2: número de sub-diretórios dentro do *level1*. É no *level2* que os objetos são efetivamente gravados.

Se for especificada mais de uma cláusula **cache_dir**, o squid faz a distribuição do cache entre as várias partições.

squid.conf: segurança

Definindo uma lista de acesso:

acl *aclname acltype value*

aclname: nome qualquer para identificar a lista de acesso. Esse nome será usado quando se for referencia a lista.

acltype value: tipo da lista (cada tipo requer um valor num formato específico):

- **src** *client_ip_addr/netmask ...*
- **dst** *server_ip_addr/netmask ...*
- **srcdomain** *client_ip_name ...*
- **dstdomain** *urlserver_name ...*
- **srcdom_regex** *client_name ...*
- **dstdom_regex** *urlserver_name ...*
- **time** [{**S**|**M**|**T**|**W**|**H**|**F**|**A**}] [*hh*:*mm*-*hh*:*mm*]

acltype value:

- **url_regex** *urlregex ...*
- **urlpath_regex** *pathregex ...*
- **port** *portrange ...*
- **proto** *protocol ...*
- **method** *methodname ...*
- **browser** *regexp ...*
- **user** *username ...*
- **src_as** *number ...*
- **dst_as** *number ...*
- **proxy_auth** [*refresh*]

Exemplos:
acl localhost src 127.0.0.1/255.255.255.255
acl all src 0.0.0.0/0.0.0.0
acl ok_ports port 80-85 443 563

squid.conf: segurança

Permitindo e negando acessos:

http_access {**allow**|**deny**} *aclname*
icp_access {**allow**|**deny**} *aclname*
miss_access {**allow**|**deny**} *aclname*

aclname: nome referente a uma lista de acesso anteriormente criada

Configuração de Hierarquias

Comunicação entre servidores

Como é realizada:

1. Geração do Cache Digest

Cada servidor gera, de tempo em tempo, seu cache digest. O cache digest nada mais é que um vetor de bits gerado (por funções hash) a partir das URLs *cacheadas* servidor. Através dele é possível saber com certa precisão se, dada uma URL, ela será ou não encontrada no servidor que gerou o digest.

Parte B: Instalação e Configuração do Squid

2. Transmissão do digest

Cada servidor squid requisita a todos os seus vizinhos (pais ou irmãos) uma cópia do digest de cada um. Como o digest não é estático, ele é requisitado periodicamente.

3. Busca de objetos

Quando uma requisição chega a um proxy server e ela não pode ser resolvida localmente, o proxy verifica o digest de todos os seus vizinhos para saber se o objeto será encontrado em algum deles. Assim, o objeto será buscado do vizinho mais próximo (pai ou irmão, indistintamente) dentre aqueles que têm esse objeto. Caso todos os digests causem um *miss*, a requisição será direcionada ao pai mais próximo (se existir) ou ao servidor de origem.

Eficiência do Cache Digest

Problemas:

- False Miss: uma consulta ao digest “informa”, erroneamente, que o vizinho NÃO POSSUI determinado objeto.
- False Hit: uma consulta ao digest “informa”, erroneamente, que o vizinho POSSUI determinado objeto

O problemas mais grave são os *False Misses*, que faz com que o squid a busque os objetos da origem quando na verdade poderia buscá-lo de um dos vizinhos. Medições realizadas na hierarquia do NLANR mostraram que, na prática, o número de False Misses é muito pequeno (na faixa de 0,3%).

squid.conf: hierarquia

Definição de servidores vizinhos

cache_peer *name* *type* *tcp_port* *icp_port* *options*

name: Nome (ou endereço IP) do vizinho

type: parent ou sibling

tcp_port: porta usada pelo vizinho para atender requisições HTTP

icp_port: porta usada pelo vizinho em sessões icp

options: proxy-only, weight=n, ttl=n, no-query, default, round-robin, multicast-responder, closest-only, no-digest, no-netdb-exchange, no-delay

squid.conf: hierarquia

Configuração de vizinhos apenas para determinados domínios

cache_peer_domain *name* *[!] domain (...)*

name: Nome (ou endereço IP) do vizinho

domain: domínios para os quais esse vizinho responde, separados por espaço. Um ponto de exclamação é usado como negação

Exemplo: para mandar para o vizinho de ip 10.0.0.1 requisições de todos os subdomínios de .br exceto .rnp.br, use:

cache_peer_domain *10.0.0.1* *!.rnp.br .br*

Referências

- **Squid Cache Object - NLANR**

  http://squid.nlanr.net/

- **Configuring Hierarchical Squid Caches** - Duane Wessels, Aug 1997

  http://squid.nlanr.net/Squid/Hierarchy-Tutorial/

- **Cache Digests** - Alex Rousskov and Duane Wessels, Jun 1998

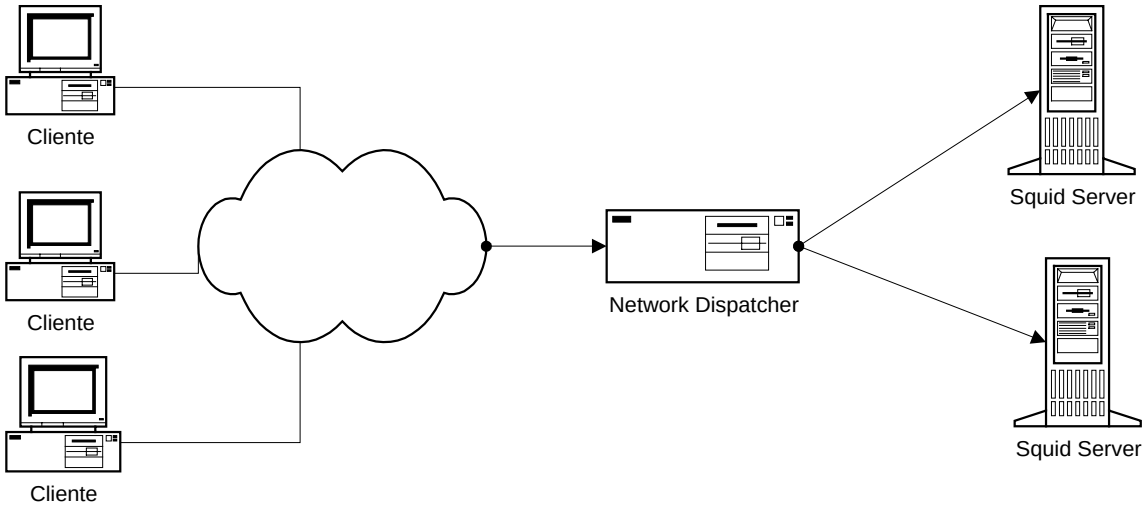
  http://wwwcache.ja.net/events/workshop/31/rousskov@nlanr_net.ps

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- O que é o Network Dispatcher
- Módulos do Dispatcher
- Instalação e Configuração
- Interactive Session Suport (ISS)
- Integração do Dispatcher com Squid
- Referências

O que é o Network Dispatcher

Software para rotear requisições TCP/IP entre um cluster de servidores, possibilitando a realização de balanceamento de carga



Módulos do Dispatcher

O Network Dispatcher é dividido em três tarefas:

Executor

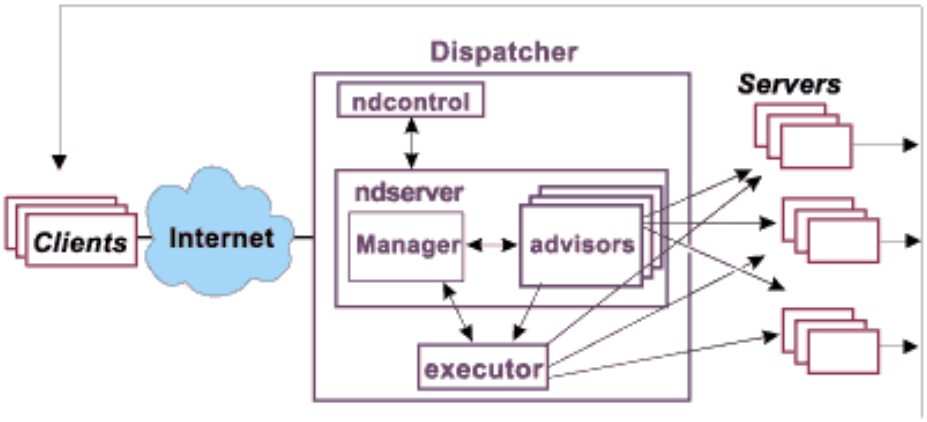
É o responsável por endereçar cada nova conexão a um dos servidores. O **executor** mantém uma tabela com pesos para cada um dos servidores, usando-a para determinar a quantidade de conexões que será enviada para eles (o número de conexões que vai para cada servidor é proporcional ao seu peso).

Manager

Coleta informações do **executor**, dos **advisors** e de outros programas de monitoração (ex: ISS) e as usa para calcular o peso que cada servidor do cluster passará a ter.

Advisor

Os **advisors** são específicos para cada protocolo de aplicação (no caso do proxy, usaremos o **advisor** http). Ele monitora uma determinada porta de cada servidor e coleta informações como tempo de resposta e disponibilidade, repassando essas informações para o **manager**.



Instalação e Configuração

O funcionamento do Network Dispatcher

O dispatcher é, na verdade, uma máquina que tem dois números ips associados à sua interface de rede:

- **IP do dispatcher (NFA):** é o número IP primário da interface de rede. Os pacotes endereçados a esse IP tem como destino final o próprio dispatcher.
- **IP do cluster:** é um número IP virtual associado à interface de rede. Os pacotes recebidos pelo dispatcher que são endereçados a esse IP são repassados para os servidores do cluster.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

Os servidores que fazem parte do cluster também utilizam dois números IP:

- **IP do servidor:** é o número IP da interface de rede. Cada servidor tem um número IP único que o identifica (endereçamento padrão de uma sub-rede).
- **IP do cluster:** esse IP (o mesmo que fora associado à interface de rede virtual do *dispatcher*) também deve existir no servidor, para garantir a consistência da comunicação IP. Esse número, entretanto, não pode ser associado à interface de rede de cada servidor, pois isso ocasionaria um conflito na rede (já que ele já está sendo usado no *dispatcher*). É criada então um interface virtual no loopback dos servidores e associada a ela o IP do cluster.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

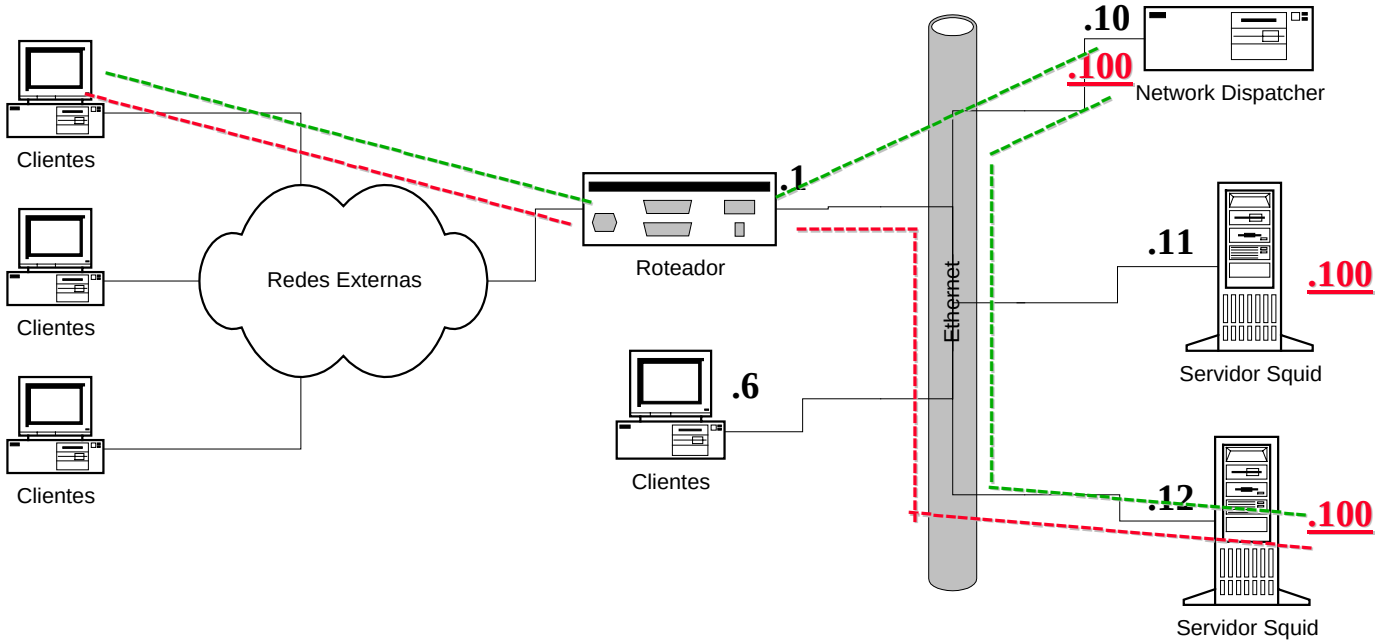
A comunicação passa pelo dispatcher apenas no sentido cliente -> servidor, e não no sentido servidor -> cliente:

Quando um cliente tenta se conectar no número IP do *cluster*, o pacote IP é roteado até chegar à interface (virtual) do *dispatcher*. O dispatcher, por sua vez, sabe quais são os endereços dos servidores, roteando o pacote para um deles (o servidor aceita esse pacote, pois o endereço final é igual ao endereço (virtual) de sua interface de loopback).

Como o servidor desconhece a presença do *dispatcher*, a resposta é enviada para o endereço que consta no campo *src_addr* do pacote IP, ou seja, a resposta vai diretamente para o cliente, sem passar de novo pelo *dispatcher*.

Novos pacotes enviados pelo cliente pertencentes à mesma sessão/conexão são roteados pelo dispatcher para o mesmo servidor.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...



Rotina de Configuração

- O número ip fixo para gerência (NFA) do dispatcher é o número ip primário da interface de rede. Escolha um número ip para o serviço a ser fornecido pelo dispatcher (um para cada cluster) e configure-o como ip *alias* do adaptador de rede:

ifconfig *adapter* **alias** *cluster_ip* **netmask** *netmask*

- Configure a interface de loopback de todos os servidores com o mesmo número ip (alias):

ifconfig lo0 alias *cluster_ip* **netmask** *netmask*

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- Execute o *ndserver*:
  ndserver start

- Execute o *executor*:
  ndcontrol executor start

- Configure o endereço ip NFA do dispatcher:
  ndcontrol executor set nfa *ip_address*

Configure o endereço ip cluster do dispatcher:
ndcontrol cluster add *cluster*

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- Defina as portas que serão respondidas pelo cluster:
  ndcontrol port add *cluster:port*

- Defina quais servidores farão parte desse cluster nas portas especificadas:
  ndcontrol server add *cluster:port:server1*
  ndcontrol server add *cluster:port:server2*
  ndcontrol server add *cluster:port:server3*
  (...)

Configuração do Manager e Advisors

- Execute o manager
  ndcontrol manager start

- Execute o advisor
  ndcontrol advisor start *protocol port*
  onde *protocol* = {ftp, telnet, smtp, http, pop3, nntp, ssl}

- Ajuste o peso máximo permitido para um servidor (utilização máxima desse servidor em relação ao menos utilizado)
  ndcontrol port set maxweight *peso*

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- Configure a periodicidade (segundos) da interrupção do executor pelo manager:
  ndcontrol manager interval *segundos*

- Configure periodicidade (em número de intervalos) em que o manager pedirá informações ao executor:
  ndcontrol manager refresh *intervalos*

- Configure a periodicidade do advisor para pedir aos servidores o status das portas monitoradas e reportará os dados ao manager:
  ndcontrol advisor internal *protocolo porta segundos*
  Recomendação da IBM: não mudar o default.

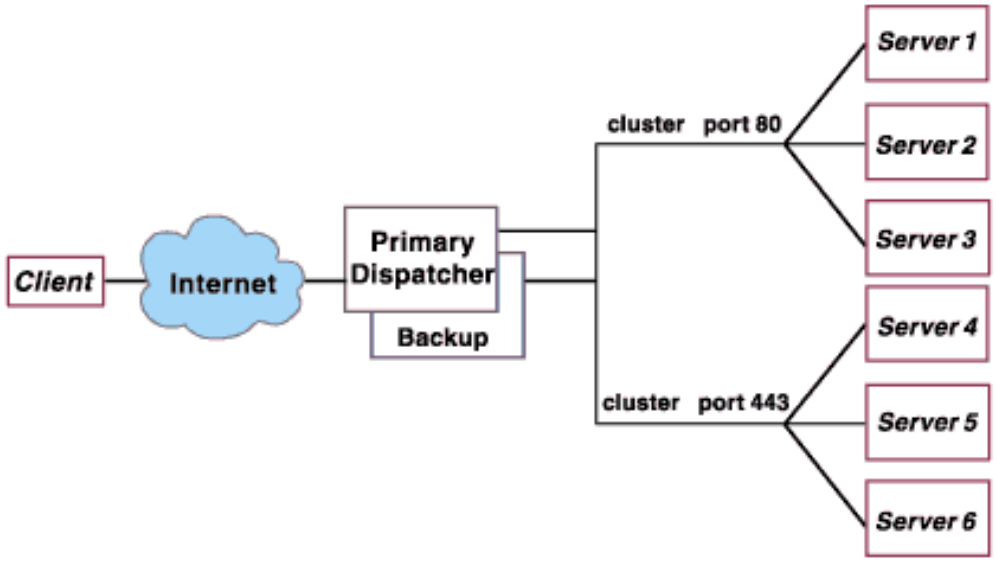
Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- Configure o *threshold* do manager, ou seja, qual a mudança mínima necessária em relação ao estado atual do servidor para que ele interrompa o executor e altere os pesos: (a sensitividade é dada em porcentagem do estado atual)
  ndcontrol manager sensitivity *sensitividade*

- Configure o *smoothing index*, ou seja, a constante que regula a alteração máxima que pode ser feitas nos pesos. Um índice X pode ser pensado como sendo alteração máxima de weight/X, ou seja, quanto maior o X mais suaves são as alterações.
  ndcontrol manager smoothing *índice*

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

High Availability



Com o High Availability, dispatcher backup assume as funções do dispatcher primário em caso de falhas.

High Availability

Para o uso de high availability, é necessário o uso do manager e dos advisors.

- Adicione a informação de “heartbeat” nos dois dispatchers:
  ndcontrol highavailability heartbeat add *ip_nfa_local ip_nfa_remoto*

- Adicione os hosts que o dispatcher deve alcançar para garantir seu funcionamento:
  ndcontrol highavailability reach add *ip*

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

- Configure, no dispatcher primário, a porta que deverá ser utilizada na comunicação com o servidor backup:
  ndcontrol highavailability backup add primary [**auto**|**manual**] *numporta*

- Configure, no dispatcher backup, a mesma porta:
  ndcontrol highavailability backup add backup [**auto**|**manual**] *numporta*

- Confira o status da configuração nas duas máquinas:
  ndcontrol highavailability status

- Copie os scripts **goActive** e **goStandby** do diretório **samples** para o diretório **bin**, e edite-os para que eles reconfigurem o cluster corretamente em casos de queda e volta do dispatcher primário

Interactive Session Suport (ISS)

Integração com o Network Dispatcher

O ISS é usado em conjunto com o Network Dispatcher para prover ao manager informações sobre a carga de cada servidor do cluster. É composto de dois módulos:

iss-agent

Agente que deve ser executado em todos os servidores do cluster. Esse agente, a pedido do ISS, executa um determinado comando externo e retorna como resposta a saída desse comando (um valor númerico).

Esse comando externo que será executado será informando pelo ISS, e pode ser qualquer script ou programa que retorne um valor numérico.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

ISS

É executado em uma máquina (por exemplo, o dispatcher) e, periodicamente, chama cada um dos agentes. Os valores obtidos como resposta são utilizados para se saber o quanto um servidor está melhor que outro.

O resultado dessa análise é passado para o módulo *manager* do *dispatcher* que, por sua vez, irá usá-lo quando for atribuir pesos para os servidores.

Exemplo de configuração do ISS:

```
METRIC CUSTOM netstat -tn | wc -l

POLICY MIN
```

indica que, quando menos (MIN) conexões TCP o servidor tiver, maior será o peso atribuído a ele pelo *dispatcher*. Podem ser criados scripts que capturem o tempo de CPU, uso de disco, rede, etc.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

Instalação do iss-agent

- Via SMIT, instale o iss-agent em cada servidor
- Altere o **/etc/services**, acrescentando o serviço *issagent porta/tcp*. A porta geralmente utilizada é a 10001
- Altere o **/etc/inetd.conf** para permitir que o issagent seja disparado pelo dispatcher.
  Será acrescentada a seguinte linha ao arquivo:
  issagent stream tcp nowait root /usr/lpp/issaix/iss/issagent issagent
- Reinicie o inetd dos servidores

Instalação do ISS

- Exemplo de arquivo de configuração (para integração com o dispatcher)

```
ISS_MODE IGNORE_NAMED
ISS_PORT 10001
NAME                    cachernp.anades.dcc.ufmg.br
DISPATCHER_PORT         10004
METRIC                  CUSTOM netstat -tn | wc -l
POLICY                  MIN
SERVERS                 rnp0 rnp1
DISPATCHERS             dispatcher
HEARTBEAT_INTERVAL      10
HEARTBEATS_PER_UPDATE   2
POOL_ALARM echo "cachernp ran out of servers at `date`" >>
    /tmp/issalarm.log
```

Integração do Dispatcher com Squid

Configuração de Hierarquias

Relacionamento entre os servidores de um PoP

Tipo de relacionamento

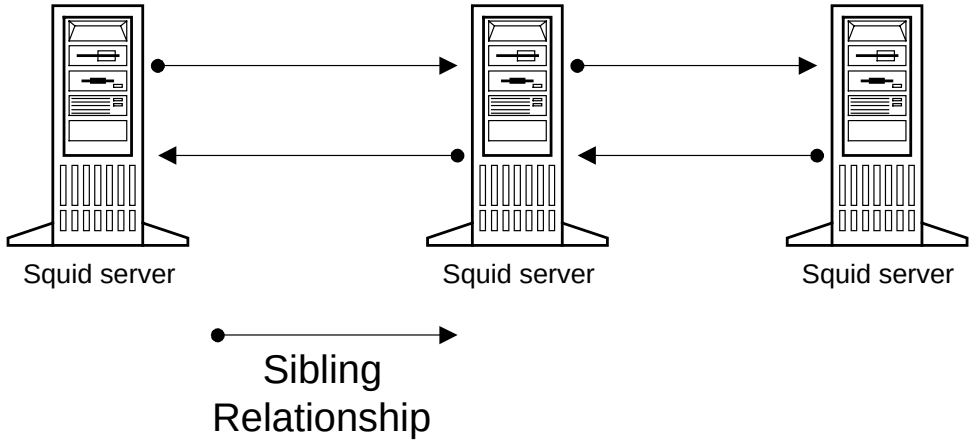
Os servidores de um mesmo cluster são configurados como sibling, ou seja, um nodo só faz requisição a outro em caso de Hit. A configuração é feita diretamente no squid, sendo independente do uso do dispatcher.

Replicação de objetos entre servidores do cluster

A teoria nos leva a crer que esse relacionamento deve ser do tipo *proxy-only.* Ainda nos resta testar, na prática, os prós e contras desse relacionamento.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...

Relacionamento entre os servidores de um PoP



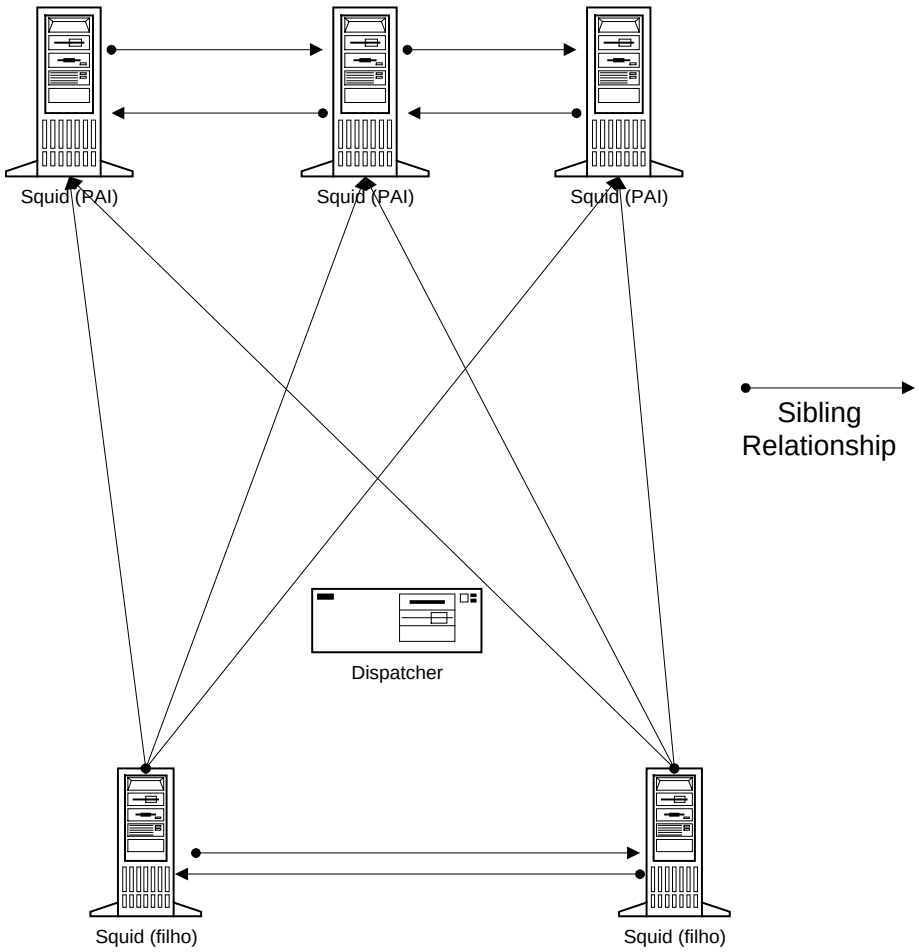
Parte C: Instalação e Configuração do Network Dispatcher/ISS...

Relacionamento entre PoPs

Sibling

No caso de dois PoPs serem irmãos (relação para PoPs no mesmo nível da hierarquia), o relacionamento é configurado fazendo-se todos os nodos do cluster de um PoP como *sibling* de todos os nodos do cluster do outro PoP. Essa configuração também não depende da existência do dispatcher. A disponibilidade é garantida pelo próprio squid, que só requisições para vizinhos que estejam operantes.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...



Relacionamento entre PoPs

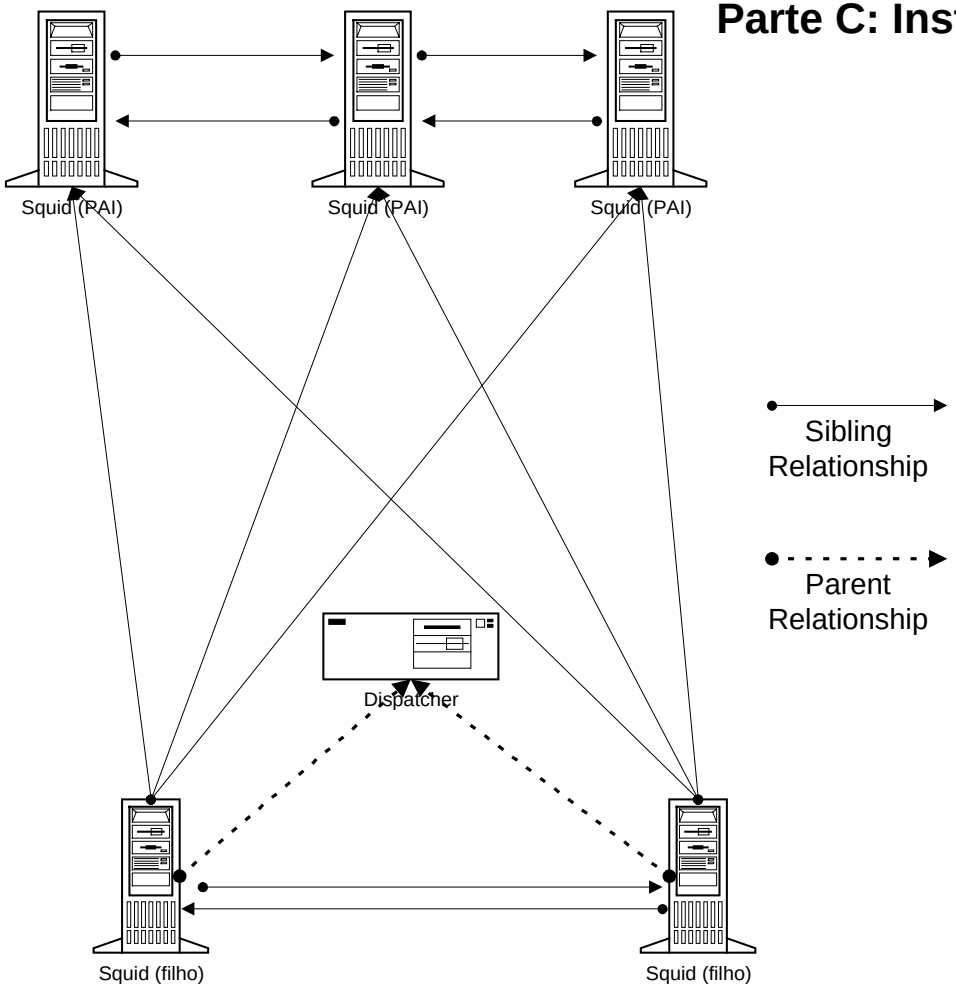
Parent

No caso do cluster de um PoP ser filho do cluster de um outro PoP (relação entre PoPs de diferentes níveis da hierarquia), o relacionamento é configurado da seguinte forma:

- Todos os nodos do cluster do PoP do nível inferior são configurados como *sibling* de todos os nodos do cluster do PoP de nível superior
- Todos os nodos do cluster do PoP do nível inferior são configurados como filhos do Network Dispatcher (endereço do cluster)

Essa configuração garante a melhor taxa de acerto. O balanceamento de carga continua sendo preservado pelo dispatcher, que roteará as requisições resultantes de miss (e as feitas de outros clientes) para os servidores menos carregados.

Parte C: Instalação e Configuração do Network Dispatcher/ISS...



Referências

- **Using and Administering Interactive Session Support for AIX**
- **IBM eNetwork Dispatcher**

  http://www.software.ibm.com/enetwork/dispatcher/
- **Network Dispatcher 2.0 User´s Guide**

  http://www.software.ibm.com/enetwork/dispatcher/library/publications/ug_nls.html

Parte D: Estatísticas e Tuning dos Servidores

- Squid: estatísticas (cache manager, snmp)
- Squid: tuning
- Dispatcher: estatísticas
- Dispatcher: tuning

Squid: estatísticas (cache manager, snmp)

Interface Web para busca de estatísticas providas pelo protocolo cache_object, entre elas:

- Informações sobre o servidor: uptime, uso de memória e disco, número de requisições
- Parâmetros configurados: capacidade máxima do disco, memória, timeout, etc
- Utilização do servidor: Kb/s, Hit Ratio, número de objetos, etc (por protocolo)
- Estatísticas de I/O de disco
- Utilização instantânea dos file descriptors providos pelo sistema
- Lista dos objetos que estão no cache
- Estatísticas (tempo de acesso, hit ratio, etc) de cada cliente e cada vizinho
- Utilização e conteúdo do cache DNS

Parte D: Estatísticas e Tuning dos Servidores

Instalação e uso do cache manager

O **cachemgr.cgi** é gerado na compilação do squid, sendo gravado no diretório **bin/**

Copie-o para um servidor web (com permissão de execução) e use-o a partir daí. O servidor squid deve ser configurado, através das listas de acesso, para permitir que o servidor Web busque informações via cache manager.

O cache manager era a principal ferramenta para monitoração do servidor. Apesar de ser bastante útil em determinadas ocasiões, a tendência é que ele perca espaço, uma vez que a grande maioria das informações por ele providas já podem ser obtidas via snmp.

Squid: tuning

Dentre os aspectos de configuração que costumam ser fontes de problema, estão:

- Tempo de vida do objeto (LRU expiration time) muito baixo: implica que a atual capacidade do disco está pequena para as necessidades. Soluções são o aumento do disco ou a diminuição do tamanho máximo do objeto que pode ser armazenado.
- Todos os Squid DNS Servers (resolvedores de nomes do squid) em utilização: um dos potenciais gargalos do squid, pois resolução de nomes é uma das únicas chamadas blocantes que o squid realiza.
- Uso constante de swap: configuração muito alta do cache_mem (pool de memória reservado para hot_cache e objetos em trânsito) - ou, obviamente, existência de pouca memória no sistema.
- entre outros.

Squid: tuning

Criação de uma linha de base (*baseline*)

Estatísticas de todo o sistema (squid e sistema operacional), bem como os detalhes da configuração, devem ser coletados para a criação de uma *baseline* quando o servidor estiver apresentado uma boa performance.

Esses valores devem ser coletados e armazenados, pois terão grande uso na detecção de problemas futuros. Através de dados a respeito do funcionamento perfeito do servidor é possível saber a fonte de problemas caso eles venham a acontecer, possibilitando uma recuperação do sistema de maneira bem mais simplificada.

Dispatcher: estatísticas

Monitoração

Uma grande deficiência da primeira versão do dispatcher é não disponibilidade de um agente snmp (deficiência já contornada na versão 2). Assim, a principal forma de monitoração remota é através do uso da MIB do AIX.

Estatísticas

Estatísticas de utilização e informações sobre o estado atual do dispatcher podem ser obtidas através da adição da cláusula “status” ou “report” em vários dos comandos do ndcontrol. Exemplo:

ndcontrol manager status

ndcontrol advisor report *cluster:port*

Dispatcher: tuning

A otimização do dispatcher diz respeito à forma de melhorar a distribuição de carga. Isso pode ser feito de duas maneiras:

- Corrigindo os pesos atribuídos a cada módulo do dispatcher
- Reformulando as métricas usadas pelo ISS: idealmente, para se encontrar o servidor menos carregado usando-se o ISS deve-se levar em consideração aspectos como:
  - Tempo de CPU
  - Utilização de disco
  - Utilização de rede
  - Utilização de memória

Esses parâmetros são específicos para as necessidade do servidor. No caso do squid, ainda faltam estudos para definir o real peso de cada um.

Parte E: Ferramentas de Monitoração

- MIB do AIX
- MIB do Squid
- **pnm**
- Referências

MIB do AIX

Informações do sistema: versão SO, nome, uptime, etc.

Interfaces: características (MTU, velocidade), endereços físicos, informações de tráfego (número de

IP: configurações (ip forwarding, TTL), tabela de rotas, estatísticas (número de pacotes enviados, recebidos OK, descartados, etc)

TCP: número de conexões (máximo e atual), estado de cada conexão, estatísticas (número de segmentos enviados, recebidos OK, descartados, etc)

UDP: número de endpoints (sockets) abertos, estatísticas (número de datagramas enviados, recebidos OK, descartados, etc)

MIB do Squid

cacheSystem: uptime, vmsize, capacidade do disco

cacheConfig: nome/versão do software, email do admistrador

cacheStorageConfig: configuração de memória e disco usadas pelo squid

cachePerf: número de falhas de página, tempo de CPU consumido, número de objetos armazenados, total de requests, hits, errors, tempos de resposta (dns, http, icp), etc...

cacheNetwork: estatísticas do ip, dns e fqdn Cache (requests, hits, negative hits, miss)

cacheMesh: configuração de hierarquia (estado operacional dos siblings e parents, tempos de resposta desses vizinhos, etc), estatísticas de requisições recebidas (listas dos clientes, número de requests por cliente, número de hits, etc).

pnm

Monitoração de variáveis SNMP

O **pnm** é formado por um conjunto de módulos que trabalham de maneira integrada no gerenciamento de monitoração. Sua arquitetura foi concebida e maneira a aliar flexibilidade à facilidade de uso, tornando-se uma ferramenta poderosa em redes de qualquer porte.

Log das variáveis coletadas (histórico)

Através do armazenamento em disco dos valores coletados, é possível ao administrador configurar a geração de eventos baseado nos estados passados de todas as variáveis coletadas.

Tomadas de decisão baseadas em expressões regulares

Todas as tomadas de decisão são baseadas na avaliação de expressões regulares (sintaxe=perl), que podem envolver uma ou mais das variáveis coletadas (no seu estado atual ou em qualquer estado passado).

Ações Personalizadas

Os alarmes gerados pelo **pnm** nada mais são que scripts externos. Isso permite gerarmos qualquer tipo de alarme, como envio de email, pager, etc.

Correlação de Eventos

Além dos estados das variáveis SNMP, as expressões regulares do **pnm** podem ter como entrada qualquer fonte externa, permitindo uma grande flexibilidade.

Gerenciamento pró-ativo

Através do uso dos estados passados das variáveis, é possível implementar o gerenciamento pró-ativo facilmente. Podemos, por exemplo, gerar eventos sempre que algum estado permanecer inalterado por determinado tempo, prevenindo assim um potencial *hazard*.

Geração de estatísticas

Os arquivos gerados pela coleta das variáveis representam um histórico dos estados, podendo ser usados posteriormente como entradas para scripts de geração de estatísticas.

pnm

Configuração

Definição de *alias* para a hierarquia SNMP

Para facilitar a leitura do pnm.conf, podem ser definidos *aliases* para partes da MIB do elemento que estará sendo monitorado.

```
// Defines aliases for SNMP variables
// Syntax:
// #define alias variable
// Aliases can be later used as prefixes or suffixes,
// for example: alias.1.3  3.alias 1.2.alias.4
#define oper interfaces.ifTable.ifEntry.ifOperStatus
#define admin interfaces.ifTable.ifEntry.ifAdminStatus
```

Definição das variáveis que serão coletadas

```
// Collects SNMP variables. Syntax:
// c:label:snmp_variable:community@machine
// labels can be used in regular expressions as $label or
label[0]
// to represent the current value of a SNMP variable. Past
states
// are accessed using $label[1], $label[2], $label[3]...

c:routerOperS0:oper.1:public@router.pop-mg.rnp.br
c:routerAdminS0:admin.1:public@router.pop-mg.rnp.br
```

Definição das ações que serão tomadas de acordo com os valores das variáveis

```
// Calls the module that pages network administrators reporting that
the
// link is down if the interface is administratively up, if it was
// operationally up (two pollings before), if it became operationally
// down on the last polling and if it remained that way.
// Syntax:
// a:regex:action
a:(("$routerAdminS0"=~/up/)&&("$routerOperS0"!~/up/)&&("$routerOperS0[1
]”!~/up/)&&("$routerOperS0[2]"=~/up/)):/opt/pnm/bin/beep router S0
routerOperS0

// If the interface was operationally down and came back up, report the
// network administrator
a:(("$routerAdminS0"=~/up/)&&("$routerOperS0"=~/up/)&&("$routerOperS0[1
]"!~/up/)):/opt/pnm/bin/beep router S0 $routerOperS0
```

Referências

- **Squid SNMP**

  http://squid.nlanr.net/Squid/FAQ/FAQ-18.html

- **pnm: a flexible, extensible and low-cost network management solution**

  http://www.dcc.ufmg.br/~magc/artigos/

Parte F: A Hierarquia Nacional da RNP

- Topologia Inicial
- Tuning da Hierarquia
- Interligação com Hierarquias Mundiais

Topologia Inicial

O Papel do PoPs na hierarquia

Inicialmente, os SP2 (que seriam nível 1) estarão inativos.

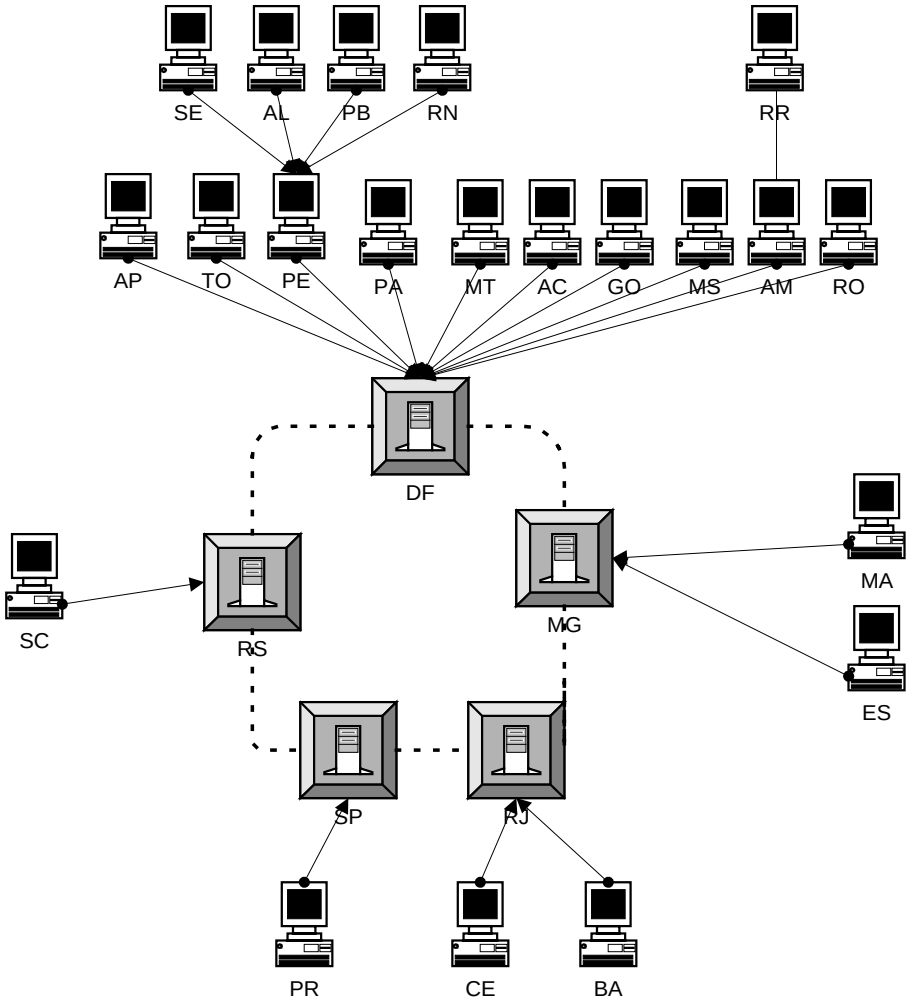
PoPs do Pentágono: Nível 1

- Todos estarão no nível 1 da Hierarquia e serão irmãos entre si
- Atuarão comos pais dos PoPs com cache nível 2, que são aqueles diretamente conectados com os PoPs do Pentágono

PoPs diretamente conectados ao Pentágono: Nível 2

- Serão filhos dos servidores dos PoPs nos quais estão conectados
- Atuarão comos pais dos PoPs com cache nível 3 (PoPs que não se incluem nas duas situações citadas)

Parte F: A Hierarquia Nacional da RNP



Sibling Relantionship

Parent Relantionship

Tuning da Hierarquia

Ainda é cedo para sabermos quão boa é a hierarquia inicial proposta. Durante o seu funcionamento, algumas métricas serão analisadas para sabermos onde e o que ajustar. Essas métricas representam a qualidade do serviço, sendo elas:

- Tempo de serviço (*Service Time*)
- Taxa de acerto (*Hit Ratio*)
- Estado (tráfego) das linhas do *backbone*

Questões:

- Quais são os resultados aceitáveis?
- Como descobrir a causa de resultados abaixo dos aceitáveis?
- Como corrigir o problema?

Tuning da Hierarquia

Service Time

A alteração no tempo de serviço pode ser ocasionada por:

- **Miss Ratio**
  - Variedade dos objetos
  - Thrashing
- **Overload de servidores**
  - Outstanding Connections
  - Overload de CPU
  - Overload de memória (page faults)
  - Overload de disco
- **Tráfego nas linhas (upstream / downstream)**
  - Problema conjunto: ajuste de hierarquia e de roteamento

Parte F: A Hierarquia Nacional da RNP

Tuning da Hierarquia

Hit Ratio

Muitas das vezes, apesar de o tempo de resposta estar dentro do esperado, uma baixa taxa de hit pode estar impedindo a melhoria desse tempo.

O Hit Ratio é influenciado por:

- Variedade dos objetos: um caso possível é que requisições destinadas a um servidor que esteja apresentando Hit local baixo poderiam ser resolvidas por um outro servidor.

Ex: ES poderia ter um cliente governamental (que acessa apenas dados em Brasília) e redicionar suas requisições diretamente pra lá, uma vez que o Hit em MG está um baixo.

- Thrashing
  - Ocasionado por limitação de disco (o que não é um problema nesse caso, dada a configuração dos servidores da RNP)

Tuning da Hierarquia

Estado das linhas do backbone

Outra métrica a ser observada é o estado dos links. Apesar de ser extremamente improvável, algum PoP (dentro do pentágono) pode se tornar um “atrator de requisições”. Isso certamente causará um desbalanceamento nas linhas. Esse problema deverá ser resolvido levando-se em consideração:

- Alterações na política de roteamento
- Alterações na topologia lógica da hierarquia

Interligação com Hierarquias Mundiais

O NLANR

Outra configuração que deverá ser realizada é o uso dos PoPs que tem saída internacional como filhos dos servidores da NLANR mais “próximos”.

Nesse caso, seria configurada o forwarding apenas das requisições destinadas aos *top-level domains* pertencentes aos EUA (ou, em um caso a ser estudado, a tudo o que não estiver no domínio “.br”)

Deve-se verificar posteriormente (seguindo as métricas anteriormente citadas) a viabilidade de nos mantermos como filhos, usá-los como irmãos ou simplesmente não nos integrarmos.

Web Site e Lista de Discussão

http://www.lct.rnp.br/proxies

Lista: **mailto:proxies@rnp.br**

Abertas apenas para a RNP e os PoPs

Inscrições: **mailto:listproc@rnp.br**

No corpo da mensagem:

subscribe proxies username@pop-xy.rnp.br Nome_Completo