Implementação de um balanceador de carga utilizando o Linux Virtual Server

Caciano Machado, Éverton Foscarini, Fernando Macedo

[1]Universidade Federal do Rio Grande do Sul
Centro de Processamento de Dados
Rua Ramiro Barcelos, 2574 – Portão K – Porto Alegre – RS

{caciano,foscarini,fmacedo}@cpd.ufrgs.br

Resumo. *Tendo em vista a necessidade de aumento dos níveis de disponibilidade e desempenho dos serviços de TI, a UFRGS busca permanentemente soluções que permitam um crescimento escalável na capacidade de processamento dos serviços. As soluções de balanceamento de carga utilizadas pela UFRGS nos últimos 9 anos não eram mais adequadas para atender às demandas atuais. Neste artigo será apresentada a nova arquitetura de balanceamento de serviços implementada na UFRGS, utilizando o Linux Virtual Server.*

1. Introdução

O balanceamento de carga é uma técnica empregada para distribuir a carga de trabalho entre múltiplos servidores e para permitir redundância, de forma a aumentar a disponibilidade de um serviço. Essa técnica é amplamente utilizada por provedores de serviços, sites e sistemas que necessitam de escalabilidade e tolerância a falhas.

Neste artigo, partimos do pressuposto que o leitor já tem conhecimento básico desta técnica, e buscamos apresentar as vantagens e melhorias encontradas ao utilizar o Linux Virtual Server em substituição aos balanceadores anteriores utilizados pelo CPD da UFRGS, assim como apresentamos especificidades de nossa implementação, principalmente nas configurações de redundância e *failover* automático.

1.1. Conceitos básicos

Abaixo enumeramos alguns conceitos básicos que não serão explicados neste artigo, e cujo entendimento é pré-requisito para a total compreensão do mesmo.

- *RealServer* - Servidor que provê o serviço.
- Farm - Conjunto de *RealServers* que faz parte do *cluster* de balanceamento.
- VIP - IP virtual (alocado para o serviço). É o endereço que o cliente acessa.
- RIP - IP do *RealServer*.
- Probe - Operação de monitoramento executada em serviço ou servidor para verificar se está operacional.

2. Motivações para uma nova solução de balanceamento de carga

De 2004 até 2012, a UFRGS utilizou serviços nativos do roteador de principal para fazer o balanceamento da carga entre os servidores. Dependendo de carga, essa tarefa podia ser bastante onerosa para o roteador pois não havia hardware dedicado para isso nesse

equipamento de rede. Além disso, a utilização do processador de propósito geral do equipamento comprometia não apenas o balanceamento, mas também as demais tarefas desempenhadas pelo roteador.

De 2004 a 2009 utilizou-se o Application Load Balancer do roteador Enterasys SSR 8600 e de 2009 a 2012 o SLB (Server Load Balancing) nativo do Cisco 6509-E. No caso do SLB da Cisco, ainda existia outra limitação que impactava na disponibilidade dos serviços: O SLB nativo não tinha suporte a VRF (funcionalidade que permite múltiplas tabelas de roteamento), utilizado pela UFRGS. Essa limitação fazia com que o tráfego dos probes do SLB tivesse que passar pelo firewall de borda da UFRGS. Dessa forma, se o firewall tivesse seu desempenho comprometido ou fosse desativado, todos serviços do SLB cairiam.

Na divulgação do resultado do vestibular da UFRGS de 2012, foi possível verificar que o SLB estava alcançando o seu limite. Conforme ilustrado nas figura 1, durante as primeiras duas horas da divulgação, quando o site da UFRGS chegou a ter 100.000 visitantes, houve um aumento expressivo do uso da CPU do roteador causado pelo pico de acessos. Os três servidores web (nginx [NGINX 2013]) do *cluster* receberam a carga de até 525 novas conexões por segundo (somados), que representa um aumento de 10 vezes na carga do serviço comparando com a média do dia.

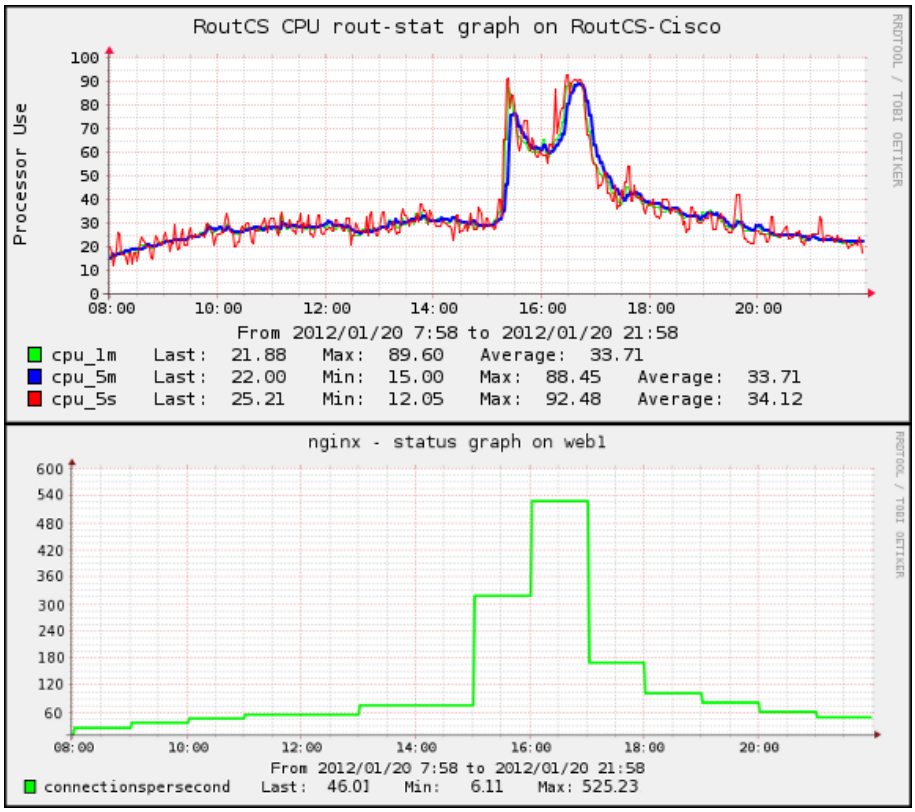

Figura 1. Divulgação do vestibular 2012

Outra limitação encontrada na solução de balanceamento até então utilizada era a impossibilidade de utilizar o SLB com endereços IPv6. Essa restrição impossibilitaria a transição de serviços para dual-stack (IPv4 + IPv6), processo que acompanha a implementação de endereçamento IPv6 para a Universidade.

As limitações impostas pelas soluções adotadas, assim como o problema de desempenho gerado no roteador, motivaram a busca por novas soluções que atendessem as necessidades atuais e futuras da UFRGS.

3. Definição de uma nova solução de balanceamento

As equipes de Engenharia de Rede e de Suporte de Software foram incumbidas de procurar uma nova ferramenta para substituir o balanceador de carga, e foram feitos estudos de duas opções:

3.1. Cisco Content Switching Module

O Cisco Content Switching Module (CSM) é um módulo de balanceamento de carga de serviços, e possui hardware dedicado para aceleração de suas funções. Esse módulo teria a vantagem de poder ser conectado diretamente em um dos slots do roteador 6509-E e ter configurações semelhantes às do serviço de SLB nativo do roteador que já estava em operação. Assim, a migração do serviço de SLB seria bastante simples.

3.2. Linux Virtual Server

O Linux Virtual Server [LVS 2013] é a de um balanceador de carga TCP/UDP no Kernel Linux. Essa implementação está integrada ao Kernel mainline desde a versão 2.4, e é suportada pelas principais distribuições. O suporte para IPv6 foi incluído na versão 2.6.28, em novembro de 2008 [LVS]. Sua configuração pode ser feita utilizando uma interface de linha de comando ou através de aplicações de gerência de *cluster*, como ldirectord e keepalived. O LVS já tem suporte nativo para sincronização de estado, permitindo a configuração de um servidor redundante, de forma a operar em modo de alta disponibilidade.

3.3. Decisão pelo uso do LVS

A solução mais cômoda a ser feita era a escolha do módulo Cisco CSM. Essa solução traria como vantagens o custo operacional menor e a possibilidade de manter um contrato de suporte com consultoria para as configurações. Além disso, o CSM é um módulo do roteador e utiliza a mesma interface CLI do 6509-E, inclusive com comandos praticamente iguais aos da solução de SLB que já era utilizada. Isso seria uma grande vantagem para a migração da solução. No entanto, também existem desvantagens que pesaram muito na avaliação, como o aprisionamento técnológico, as limitações nas possibilidades de configuração, a continuidade com um ponto único de falha e os próprios custos envolvidos na aquisição, contratos e atualizações/extensões da solução.

Além das vantagens mais óbvias do LVS, decorrentes do fato de ser software livre, como custo de aquisição zero e flexibilidade para customização, o sistema apresenta outras vantagens bastante interessantes. O LVS possui uma extensa base de usuários e soluções para os mais diversos tipos de cenários. Casos de sucesso, como o Google [Weiden and Frost 2010], pesaram bastante como critério de avaliação. A possibilidade de uso de servidores virtualizados, relatada pelo administrador do site github.com [Matt Palmer 2009] também é um elemento importante e considerado no ambiente da UFRGS, e nos indicou que não precisariamos adquirir hardware para este novo serviço. O suporte ao IPv6 era um requisito importante a ser cumprido, para permitir

a utilização da tecnologia a longo prazo. A possibilidade de implementar a redundância de balanceadores sem aumento de custo também foi um ponto importante, pois conseguimos eliminar um ponto único de falha.

Após o estudo de um cenário de testes do LVS, decidiu-se adotar essa solução em vez das outras opções comerciais. Considerou-se que as vantagens do sistema compensam o impacto no custo operacional da implementação da solução em software livre.

4. Arquitetura da Solução com LVS

A arquitetura para a solução de balanceador de carga com Linux Virtual Server foi definida em colaboração pelas equipes de Engenharia de Rede e de Suporte a Software, de forma a atender às demandas de ambas as equipes, e para atingir níveis de funcionalidade e gerenciamento superiores à solução anterior. A figura 2 apresenta uma representação dessa arquitetura, contendo as principais elementos. O detalhamento da topologia de rede criada para implementar a comunicação entre os elementos da solução de balanceamento de carga será descrito na seção 4.1.

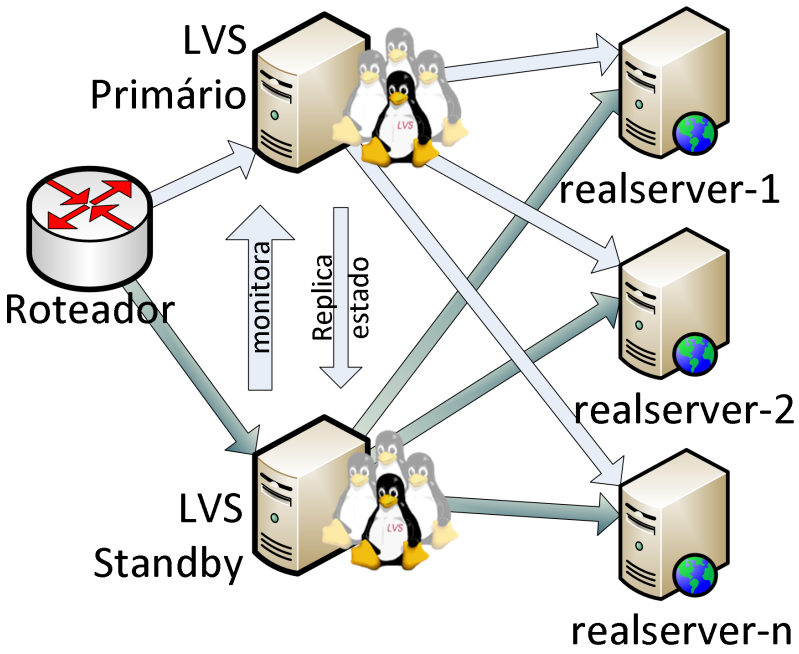

Figura 2. Arquitetura da solução LVS

Em nossa implementação, o Roteador tem a função de rotear pacotes IP destinados a um VIP através do LVS Primário, que se encarrega do balanceamento. O mecanismo de redundância entre servidores LVS, implementado pelo CPD, que se encarrega de ativar o LVS *Standby* em caso de falha do Primário será descrito na seção 4.2.

Cada servidor LVS é uma máquina virtual Xen dedicada especificamente para esta função, hospedada em um servidor XenServer, com 2 CPUs e 512MB de RAM. Utilizamos a distribuição CentOS em sua versão 6, com uma instalação mínima. Foram adicionadas apenas as ferramentas para gerência do serviço LVS: ipvsadm e ldirectord. O ldirectord é responsável por efetuar os probes nos serviços e gerenciar o estado LVS de cada servidor. A configuração do ldirectord foi efetuada de acordo

com a sua documentação e é amplamente explicada em diversos tutoriais disponíveis em [Joseph Mack 2012].

Os servidores do tipo *RealServers* são quaisquer servidores de aplicação cujo serviço seja compatível com balanceamento de carga. Atualmente no CPD da UFRGS, os seguintes serviços estão sendo servidos por balanceadores de carga:

- SMTP: postfix
- HTTP e HTTPS: apache, nginx e lighttpd
- LDAP: OpenLDAP
- HTTP-Proxy: squid
- Radius: FreeRadius

A configuração necessária para que um servidor forneça o serviço de *RealServer* é a alocação do VIP em sua interface de loopback (lo:0) com máscara /32, e configuração do serviço para que responda também através desse endereço.

4.1. Topologia da rede

A figura 3 apresenta um diagrama da topologia de rede utilizada para implementar a solução de LVS. Para o endereçamento dos VIPs foram reservados alguns IPs da rede da UFRGS, que são alocados sob demanda através de rotas específicas para os IPs (/32).

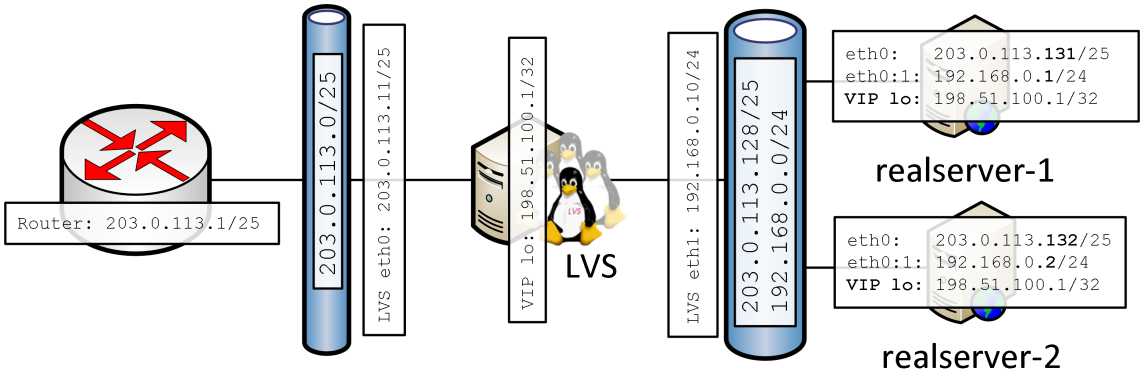

Figura 3. Topologia da rede

O servidor LVS acessa 2 segmentos de rede distintos, cada um conectado em uma interface de rede. Na primeira interface (eth0), o servidor LVS tem um endereço no segmento 203.0.113.0/25, IP pelo qual ele se comunica com o roteador. Na segunda interface (eth1), o único IP é de uma rede não-roteável (192.168.0.0/24), utilizada para comunicação com os *RealServers* para probes e encaminhamento do tráfego de clientes.

Os *RealServers*, por sua vez, têm apenas 1 interface de rede (eth0) conectada ao segmento 203.0.113.128/25. Essa mesma interface tem um IP alternativo configurado, de uma rede não-roteável (192.168.0.0/24), utilizada apenas para comunicação com o LVS.

O LVS e todos os *RealServers* têm configurados como IP alternativo de suas interfaces loopback (lo) o VIP. No LVS isso é necessário para que os pacotes destinados a um VIP sejam processados pelo servidor. Nos *RealServers* os serviços balanceados são configurados para responder também nesse endereço, de forma que a resposta é enviada

diretamente ao cliente, sem necessidade de efetuar uma tradução de endereços (NAT) no LVS ou no roteador.

Na terminologia do LVS, a topologia implementada em nossa solução é denominada LVS-DR (Direct Routing), e é a forma mais indicada de implementar um balanceador de carga com o LVS. As outras opções são o LVS-NAT e LVS-Tun. No LVS-NAT o servidor LVS intermedia o tráfego para os *RealServers*, fazendo a tradução dos endereços dos VIP para os endereços dos *RealServers*. Essa solução com NAT é semelhante à de SLB da Cisco que a UFRGS adotava e sofre dos mesmos problemas de desempenho, por isso foi descartada. O LVS-Tun requer a criação de interfaces de túnel IPIP em cada um dos *RealServers*, e só é compatível com Linux e FreeBSD. Como existe a necessidade de *RealServers* com Windows, essa última variante de LVS também foi descartada.

4.2. Failover automático para redundância de servidores LVS

A implementação de uma nova arquitetura de balanceamento de carga não poderia ser feita sem a preparação do ambiente para suportar falhas. Dessa forma, era imperativo que desde o primeiro momento já existisse a possibilidade de criar uma redundância para o servidor LVS, tanto para continuar operando se viesse a ocorrer uma falha de hardware quanto para possibilitar a manutenção do servidor LVS sem interrupção dos serviços balanceados.

O LVS implementa nativamente um *daemon* de sincronização de estado para possibilitar a configuração de um *cluster* de servidores de balanceamento. Esse *daemon* é configurado através da ferramenta de gerência ipvsadm, e a comunicação entre os servidores LVS é efetuada através de um grupo multicast local, em um segmento de rede compartilhado pelas máquinas. A sincronização do estado é feita periodicamente pelo próprio kernel Linux, de forma que o LVS *Standby* consegue manter praticamente todo o estado do LVS Primário, minimizando o número de conexões que precisarão ser reiniciadas em caso de ativação do LVS *Standby*.

Para detectar a falha do LVS Primário foi implementado um mecanismo de monitoramento no LVS *Standby*, que efetua probes a cada 2 segundos no LVS Primário. A falha de 10 probes no período de 1 minuto (20 segundos de indisponibilidade ou 30% de falhas no último minuto) é o gatilho para ativar um IP alternativo na interface de rede principal do LVS *Standby* (ifup eth0:0), que irá desencadear a alteração da rota do VIP através do mecanismo de SLA/Track.

O mecanismo SLA/Track é a utlização de uma combinação de funcionalidades do roteador Cisco [CISCO 2011]. O SLA faz o monitoramento da disponibilidade de um IP alternativo do LVS *Standby* através do protocolo ICMP. Quando o IP está disponível, o Track ativa uma rota que já estava configurada, mudando a tabela de roteamento imediatamente. A figura 4 apresenta todos os elementos dessa arquitetura de *failover* automático, e indica a sequência de eventos necessários para a ativação da rota do LVS *Standby*.

A configuração do roteador para trabalhar com o *failover* precisa ser efetuada apenas durante a alocação do VIP para um serviço. Utilizando o mecanismo do SLA/Track, o servidor LVS *Standby* consegue, apenas alocando um endereço IP

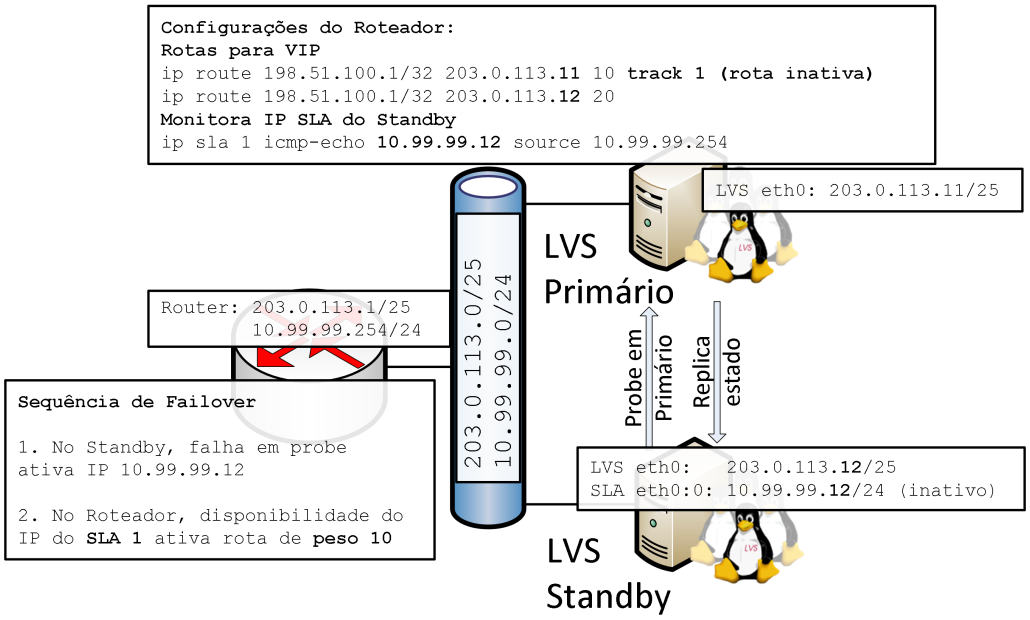

Figura 4. Failover automático do balanceador de carga

alternativo em sua interface eth0:0, reconfigurar a tabela de roteamento e executar o *failover* imediatamente.

Outros mecanismos de *failover* do servidor LVS estudados foram o Heartbeat (indicado na documentação em [Joseph Mack 2012]), e OSPF para anúncio de rota do VIP (adaptação de solução indicada em [Gardini 2007] para criação de DNS escalável), entretanto a solução adotada foi considerada mais simples de implementar e gerenciar.

5. Avaliação da solução

Para avaliar se a solução de balanceamento efetuamos testes de carga nos servidores web, buscando simular o tráfego do dia de divulgação do resultado do vestibular.

Foram criadas máquinas virtuais Linux que executavam baterias de testes através do Apache Benchmark durante 1 hora. Os testes consistiam em iniciar 50 conexões TCP concorrentes ao servidor web, e efetuar download de arquivos de tamanhos diversos via HTTP.

Nesses testes o LVS conseguiu atingir a taxa de de 2.280 conexões por segundo, e a limitação do teste foi a capacidade dos servidores web de tratar novas conexões. A CPU do roteador teve uma variação de carga que é normal para o horário do dia. A figura 5 apresenta o gráfico de desempenho de CPU e o número de conexões por segundo medidas no LVS.

Conforme apresentado na figura 6, servidor do LVS teve um aumento do load-average durante a execução do teste, porém não passou de 0,2. O uso de CPU foi quase nulo, e o tráfego de rede gerado é condizente com o número de conexões por segundo obtidas no teste.

Os resultados obtidos no teste indicam que o LVS nos permitiu atender pelo menos 4 vezes mais requisições por segundo do que o SLB do roteador Cisco gerando

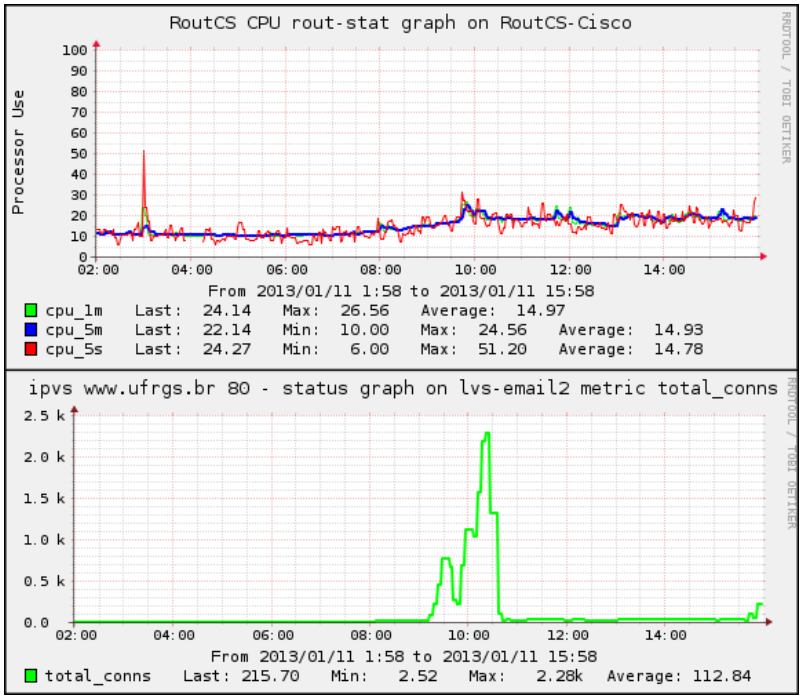

Figura 5. CPU do roteador e conexões por segundo no LVS

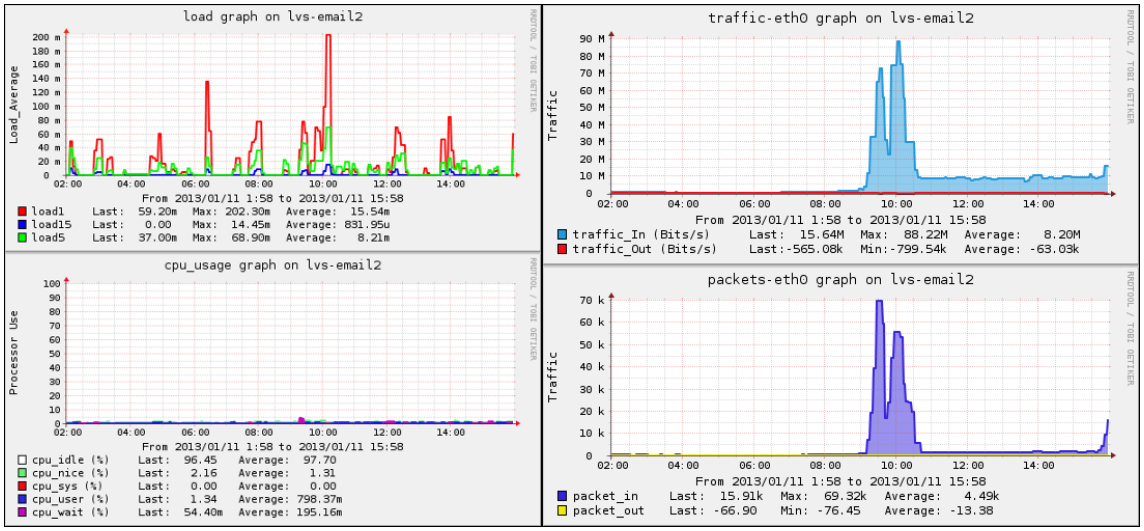

Figura 6. Uso de CPU e rede do servidor LVS

um impacto mínimo nos equipamentos de rede. Analisando o uso de recursos da máquina virtual, é possível inferir que os testes que atingiram o limite dos servidores web utilizaram menos de 20% da capacidade do balanceador, deixando uma boa margem para crescimento de demanda.

6. Conclusão

A utilização do LVS para balanceamento de carga nos permite continuar a oferecer os serviços balanceados sem a necessidade de aquisição de novos equipamentos, ao mesmo tempo que aumenta a capacidade dos serviços e desonera o roteador principal

da Universidade.

A replicação de estado do LVS nos permitiu criar um *cluster* de balanceamento redundante, permitindo a manutenção dos servidores sem a necessidade de agendamento de janelas de manutenção ou parada dos serviços, sem afetar a disponibilidade dos mesmos. Esse *cluster* também é mais resiliente, pois não há ponto único de falha.

A possibilidade de utilização de máquinas virtuais com recursos relativamente limitados como servidores LVS nos permite criar novos *clusters* caso o *cluster* inicial apresente problemas de desempenho, distribuindo a carga de acordo com o tipo de serviço.

Finalmente, o suporte nativo do LVS ao IPv6 nos permite utilizar os mesmos servidores e ferramentas de gerência, adicionando apenas os endereços IPv6 e replicando as configurações. Dessa forma, o mesmo serviço é oferecido para clientes que usem o ambos os protocolos de endereçamento, e não é preciso criar procedimentos ou utilizar ferramentas de gerência diferenciadas.

Referências

CISCO (2011). IP SLA Tracking.
http://docwiki.cisco.com/wiki/IP_SLA_Tracking_with_Configuration_Example.

Gardini, M. (2007). Gter23. In *DNS recursivo estável e escalável*.
ftp://ftp.registro.br/pub/gter/gter23/05-DNSrecEstavelEscalavel.pdf.

Joseph Mack (2012). Linux Virtual Server.
http://www.austintek.com/LVS/LVS-HOWTO/HOWTO/.

LVS (2013). Linux Virtual Server.
http://www.linuxvirtualserver.org/.

Matt Palmer (2009). Load balancing at Github: Why ldirectord.
http://www.anchor.com.au/blog/2009/10/load-balancing-at-github-why-ldirectord/.

NGINX (2013). Ngnix: open source web server and a reverse proxy server.
http://nginx.org/.

Weiden, F. and Frost, P. (2010). Lisa. In *Anycast as a Load Balancing feature*.
http://static.usenix.org/events/lisa10/tech/full_papers/Weiden.pdf.