NanoDataCenters – Uma alternativa para a distribuição de conteúdo

Aline Kaori Takechi, *Universidade Federal de São Carlos – Campus Sorocaba*

***Resumo*—Ao longo do tempo, o modelo cliente-servidor foi se consolidando como forma mais comum de distribuição de conteúdo na Internet. Entretanto, os servidores (Datacenters) utilizados para que esse modelo exista são conhecidos por terem um alto custo de manutenção e alto consumo de energia, além de reduzir a escalabilidade das aplicações devido à centralização de uma grande quantidade de dados num único local. Dessa forma, foram surgindo soluções alternativas para distribuir conteúdo, que evitassem as complicações geradas pelos Datacenters, e nesse contexto, o projeto NanoDataCenters foi criado. Esse projeto foi idealizado para possibilitar a utilização do considerável poder de processamento e armazenamento ocioso de pequenos dispositivos, tais como set-top boxes, para distribuir conteúdo entre pessoas, através da Internet.**

Palavras-chave—sistemas distribuídos, escalabilidade, distribuição de conteúdo.

I. Introdução

O projeto europeu NanoDataCenters, também conhecido pela abreviação NaDa, propõe basicamente a utilização de aparelhos como set-top boxes, que são comuns na Europa, para possibilitar uma nova solução para o armazenamento e distribuição de dados na Internet. Nesse projeto, pretende-se utilizar um grande número de nano datacenters geograficamente dispersos ao invés de pequenas quantidades dos convencionais datacenters. Além disso, em vez de utilizar esses grandes servidores conectados diretamente aos 'backbones' da Internet, o projeto propõe a utilização de uma comunicação baseada em *peer-to-peer* entre nano datacenters, procurando resolver a maioria dos inconvenientes do modelo cliente-servidor, utilizado atualmente em larga escala na Internet [1].

Num geral, esse projeto, definido pelo National ICT Australia (NICTA) com o apoio da comissão europeia 7th Framework Program (FP7), procura construir uma arquitetura para distribuir conteúdo a partir das 'bordas' da Internet, em vez de distribuir conteúdo de maneira centralizada através dos 'backbones' da Internet.

De acordo Max Ott, gerente do grupo de pesquisa de sistemas de rede do NICTA, a utilização do tradicional modelo baseado em grandes servidores faz com que exista um grande consumo de espaço físico e energia elétrica por parte desses equipamentos, além de apresentar um alto custo em termos de manutenção de hardware, redes e resfriamento [2], que são itens essenciais para manter grandes datacenters em funcionamento contínuo.

Ainda de acordo com Max Ott, atualmente o computador nem sempre é o dispositivo final em que os usuários utilizam a Internet. Por exemplo, na Europa, os serviços relacionados a entretenimento, como vídeos, música ou jogos MMOG (*Massively Multiplayer Online Games*) são comumente acessados a partir de set-top boxes ou de consoles de vídeo game. Assim, assumindo que há uma crescente evolução da capacidade de processamento, armazenamento e de eficiência energética desses pequenos equipamentos, o projeto propõe que eles sejam utilizados com a mesma funcionalidade de distribuir conteúdo dos atuais datacenters. Contudo, essa distribuição ocorreria através de centenas de milhares de set-top boxes se comunicando em conjunto, provendo diferentes serviços aos usuários finais [2].

Para que isso seja possível, foi necessário criar uma arquitetura muito bem definida para garantir segurança e privacidade dos dados pessoais, armazenados nas set top boxes [3]. Esse artigo descreve as características fundamentais dessa arquitetura e seu funcionamento na seção II.

A seção III é dedicada exclusivamente aos desafios que uma arquitetura distribuída deve superar, e em seguida, são apresentadas as soluções específicas propostas pelo projeto NanoDataCenters para cada caso.

Por fim, na seção IV, serão apresentados alguns casos que mais se adequam à arquitetura proposta pelos nano datacenters, como os serviços que oferecem vídeo sob demanda ou os games MMOG [4]. Além disso, nessa mesma seção, será abordada uma breve comparação da eficiência energética entre a utilização de conteúdos distribuídos (nano datacenters) e centralizados (atuais data centers) [5].

II. Arquitetura e Funcionamento

A. Princípios da Arquitetura NanoDataCenters

É possível afirmar que, atualmente, os modelos mais conhecidos que possibilitam a distribuição de conteúdo na Internet são bastante extremos: datacenters totalmente centralizados ou nós peer-to-peer (P2P) operando independentemente, de maneira distribuída. As duas abordagens possuem vantagens e desvantagens. Por exemplo, os datacenters são associados a alto consumo de energia, problemas de escalabilidade e manutenção, porém, devido à sua arquitetura centralizada, pode oferecer um nível de

segurança maior. Por outro lado, arquiteturas P2P podem lidar melhor com as questões de energia, escalabilidade e manutenção, porém não oferecem a mesma segurança que os datacenters.

Dessa forma, o projeto NanoDataCenters propõe uma arquitetura que apresenta características dos dois modelos citados anteriormente. Primeiramente, é uma arquitetura distribuída, semelhante ao P2P, formada por dispositivos que se encontram nas "bordas" da rede, que possuem capacidade para armazenar e distribuir conteúdo. Porém, uma única autoridade (como, por exemplo, empresas provedoras de Internet - ISP), se responsabilizaria por esses dispositivos, tornando-se mais fácil lidar com as questões de segurança da arquitetura em si.

A Figura 1 demonstra como seria o funcionamento básico da arquitetura. Os dispositivos que estão nas bordas da rede são identificados na figura por "DSL GW", e representam equipamentos conhecidos como "home gateways". Esses equipamentos podem ser, por exemplo, modems DSL ou a cabo ou set top boxes.

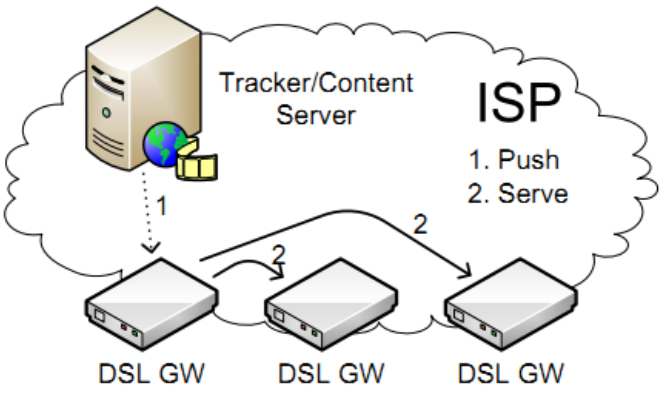

Figura 1 - Funcionamento simplificado da arquitetura NADA

Durante a criação da arquitetura dos nano datacenters, foram definidos quatro princípios que deveriam ser cumpridos: simplicidade, eficiência, segurança e robustez. Assim, para manter a simplicidade, a arquitetura possui como base o conceito de 'slices virtuais'. Esses slices, existentes em cada dispositivo dos nano datacenters, emulam quase todos os recursos de software que estão disponíveis nas set top boxes, desde o seu kernel. Dessa forma, qualquer desenvolvedor pode utilizar os recursos de um 'slice' em sua aplicação através de uma API simples, que oferece o mínimo necessário para controlar e configurar cada recurso virtualizado. Esse conceito é utilizado para manter a segurança dos dados pessoais existentes nas set top boxes, separando-os dos dados compartilhados com outros nano datacenters.

Além disso, procurou-se manter a eficiência ao virtualizar em vez de emular os dispositivos utilizados, pois dessa forma, garante-se que a aplicação também tenha acesso ao hardware do aparelho, otimizando a utilização da capacidade de processamento dos dispositivos. Pode-se dizer também que ao utilizar a virtualização, não se mantém apenas a eficiência, mas também a segurança dos nano datacenters, pois dessa forma, uma aplicação não possui acesso facilitado a nenhuma outra aplicação presente no mesmo dispositivo.

Por fim, a robustez vem como uma consequência direta da utilização da arquitetura P2P, por possibilitar replicações dos dados armazenados e tolerar falhas de alguns nós na rede.

B. Definição da Arquitetura

Baseando-se nos princípios citados na seção anterior, foram definidos três componentes principais: NADA Management Service, NADA Monitoring Service e NADA Caching Service. Foi definido também que esses componentes atuariam sobre uma plataforma formada por nós que hospedam e disponibilizam serviços.

1) NADA Management Service

Esse componente é responsável por alocar recursos de acordo com as requisições de cada aplicação, gerenciando o sistema como um todo. Pode recusar ou aceitar uma requisição, de acordo com o estado atual do sistema, procurando manter a plataforma segura e estável. Contudo, após conceder um recurso a uma determinada aplicação, essa aplicação torna-se responsável por gerenciar o que requisitou, ou seja, NADA Management Service não atua sobre recursos já alocados.

Nesse cenário, cabe ao NADA Management Service apenas decidir quais são os recursos que melhor atendem às requisições, o que não é trivial. Se as características requisitadas estão disponíveis em algum nó (ou conjunto deles), o NADA Management Service deve reservar esse recurso e transmitir à aplicação as credenciais corretas, que permitem o acesso ao recurso requisitado, sem nenhuma característica a mais ou a menos. Além disso, também é função do NADA Management Service controlar a migração de dados de um nó para outro, evitando que dados e aplicações tornem-se indisponíveis se um nó cair.

2) NADA Monitoring Service

O componente NADA Monitoring Service foi criado com o objetivo de verificar continuamente o status de cada nó. Os status monitorados são:

- Por slice: utilização de CPU, memória e armazenamento;
- Por nó: utilização da rede, consumo de energia e capacidade de transmissão de dados do disco;
- Por trajeto de rede entre pares de nós: acessibilidade, topologia do trajeto, capacidade, atrasos, perdas e banda disponível.

Esse componente auxilia o NADA Management Service a tomar decisões com relação à alocação de recursos, e também pode ser acessado pelas próprias aplicações a qualquer momento, possibilitando verificar o status dos nós em que seus recursos estão alocados.

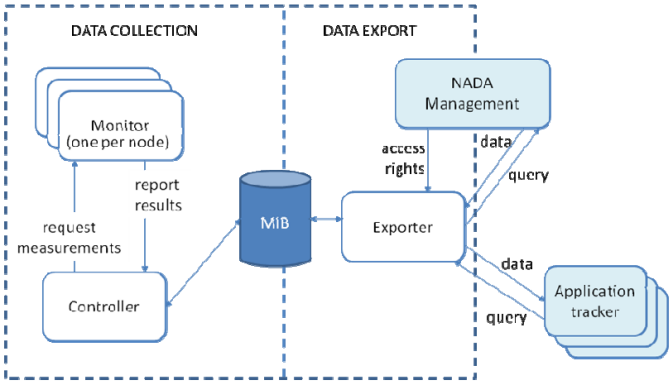

Figura 2 - NADA Monitoring Service

O NADA Monitoring Service é formado por dois subsistemas: Data Collection e Data Export, que podem ser vistos na Figura 2. O primeiro é responsável por monitorar efetivamente os nós e armazenar seus status num banco de dados, chamado MIB (Management Information Base). Para que isso seja possível, esse subsistema é composto por um conjunto de módulos, representados como Monitor na Figura 2, onde cada módulo monitora um nó diferente. Eles são responsáveis por verificar mudanças no status do nó em si e dos slices ativos nesse nó. Além disso, existe o módulo Controller, que se comunica com todos os módulos Monitor, requisitando periodicamente o status de cada nó, para que a MIB possa estar sempre atualizada.

O segundo, Data Export, existe para controlar o acesso ao MIB, de forma que apenas esse subsistema possa consultar diretamente o banco de dados, retornando apenas as informações necessárias para o NADA Management Service e para as aplicações autorizadas.

3) NADA Caching Service

O último componente, NADA Caching Service, é responsável por aperfeiçoar o desempenho do serviço de armazenamento oferecido pela plataforma. Esse componente armazena conteúdos que possuam muita demanda em servidores dedicados para cache, distribuídos pela rede. Esses servidores são vistos na rede como nós comuns, possibilitando a comunicação do servidor com qualquer outro dispositivo. Além disso, o NADA Caching Service oferece uma API para melhor utilização do serviço de cache, onde é possível requisitar o armazenamento e a remoção de arquivos do cache.

Esse componente também auxilia o NADA Management Service a tomar decisões, enviando informações sobre utilização dos recursos de cada servidor cache, de cada aplicação e também de cada arquivo armazenado.

Os componentes detalhados nos itens acima são representados na Figura 3, que ilustra como eles se comunicam entre si, entre a aplicação (APP TRACKER) e entre um nó. Percebe-se que a aplicação não se comunica diretamente com os componentes básicos do sistema, e sim por meio das interfaces NADA ID PROV e COMM/API. A primeira tem como objetivo abstrair a requisição dos recursos, ou seja, a aplicação só envia para o NADA ID PROV as características desejadas, e ele aceita ou rejeita essa requisição de acordo com estado atual do sistema. Entretanto, quem realmente toma essa decisão são os componentes NADA Management Service e NADA Monitoring Service.

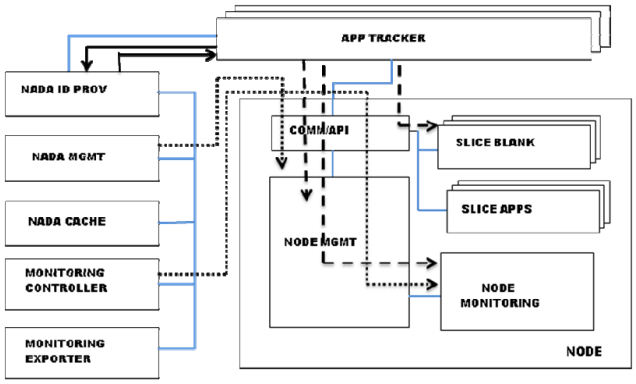

Figura 3 - Arquitetura

Ao aceitar a requisição, a interface NADA ID PROV devolve para a aplicação algumas credenciais que autorizam o acesso ao recurso alocado. Porém, a aplicação só pode utilizar esse recurso por meio de outra interface, representada por COMM/API na Figura 3. Por essa API, as aplicações podem acessar e controlar todos os recursos concedidos a ela, utilizando as credenciais obtidas do NADA ID PROV.

Contudo, a interface COMM/API deve lidar com dois tipos de instruções, pois, além da comunicação entre a aplicação e o nó, ela também realiza o intermédio na comunicação entre o NADA Management Service e o nó. Dessa forma, as instruções recebidas pelas aplicações devem ter uma prioridade menor do que as recebidas pelo NADA Management Service, para que esse componente possa manter toda a plataforma funcionando de maneira correta. Além disso, a aplicação não tem acesso aos mesmos recursos que o NADA Management Service, o que implica em funcionalidades e restrições diferentes para cada caso.

É possível verificar na Figura 3 que, além da instância local do NADA Management Service e do NADA Monitoring Service em cada nó, existem também duas representações diferentes de slices: SLICE BLANK e SLICE APPS. Os slice blanks representam recursos do nó que estão disponíveis para serem alocados, e apenas o NADA Management Service pode obter informações sobre esses recursos. Já os slice apps pertencem a aplicações específicas, de forma que as aplicações podem consultar apenas os status dos seus slice apps.

Entretanto, após conseguir alocar um determinado recurso, a aplicação precisa criar toda a estrutura de software para utilizá-lo, tornando a plataforma flexível, já que é possível determinar qual o sistema mais adequado para o funcionamento das aplicações, e instalá-lo em seus slices.

C. *Funcionamento*

Utilizando a arquitetura descrita anteriormente como base, a

plataforma em si utiliza a comunicação P2P (peer-to-peer) para distribuir conteúdo entre seus nós. Dessa forma, é possível distribuir o mesmo conteúdo para vários slices, que podem estar separados em diversos nós.

Existem vários protocolos P2P, onde cada um possui suas próprias vantagens e desvantagens. Entretanto, um único protocolo deve possuir, pelo menos, as seguintes características:

- Mínima infraestrutura centralizada;
- Operação controlável;
- Eficiência;
- Segurança.

A Figura 4 ilustra o tráfego de dados utilizando a arquitetura P2P entre o servidor cache e os nós. Para que isso seja possível, existem três principais componentes, que são P2P Controller, P2P Agent e o Cache.

O P2P Controller, identificado na ilustração como parte de Application Controller, é responsável por gerenciar toda a comunicação entre os slices de uma aplicação por meio do protocolo P2P, servindo como um tracker. Dessa forma, qualquer slice pode solicitar ao P2P Controller uma lista de outros nós que estão distribuindo determinado conteúdo, e assim conectar-se diretamente a ele. Há uma comunicação frequente entre os slices e o P2P Controller, procurando manter a lista de referências de cada nó sempre atualizada e também indicando ao P2P Controller que o nó continua ativo.

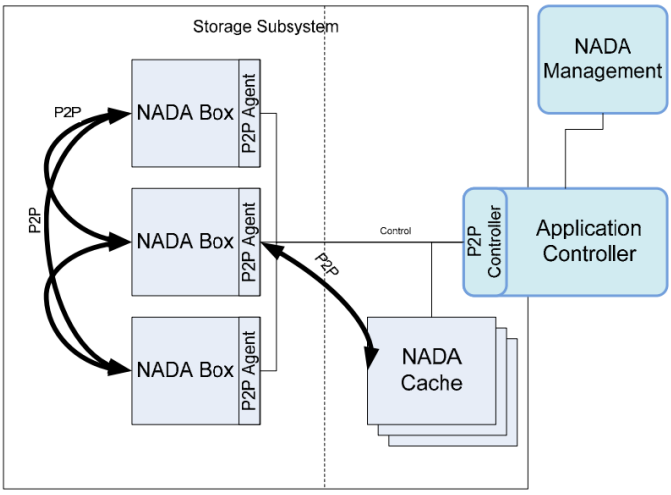

Figura 4 - Distribuição de Conteúdo entre nós

Já o P2P Agent, que se encontra instalado em cada nó pertencente ao sistema, possibilita o compartilhamento de conteúdo entre os nós de forma transparente. Ou seja, quando um nó precisa baixar ou compartilhar um arquivo, o P2P Agent se comunica com o P2P Controller correto, para iniciar uma sessão de compartilhamento.

Por fim, o servidor Cache funciona como qualquer outro nó, porém é dedicado a armazenar arquivos que possuem muita demanda de qualquer aplicação do sistema, provendo esse conteúdo quando necessário, de forma que a plataforma não se torne lenta.

III. Desafios

Como todas as arquiteturas desenvolvidas, esse projeto apresenta pontos positivos, que devem ser preservados, e negativos, que devem ser mitigados na medida do possível, e alguns desses pontos negativos acabam se tornando verdadeiros desafios, que podem impactar na utilização ou recusa da arquitetura final desenvolvida.

Um dos problemas se deve à simplicidade da arquitetura, baseada em slices, onde todo o controle dos recursos deve ser realizado pelas aplicações. Dessa forma, uma aplicação mal planejada pode acabar requisitando mais recursos que o necessário, podendo criar problemas de escalabilidade para o sistema como um todo. De acordo com os responsável pelo projeto NanoDataCenters, esse é um problema que ainda deve ser pesquisado. Contudo, um controle automatizado dos recursos alocados, por parte do sistema, pode torná-lo muito complexo e lento, contradizendo os princípios definidos para a criação da plataforma.

Outro desafio se deve ao NADA Monitoring Service, responsável por coletar o status atual dos nós. Dentre os status monitorados, alguns apresentam grandes variações em curtos períodos de tempo, como quantidade de banda disponível, por exemplo, e é necessário criar métricas para esses status, para que façam sentido para as aplicações que os utilizarão. Além disso, as aplicações precisam dos dados constantemente atualizados, entretanto, todas essas ações podem ocasionar uma sobrecarga no sistema como um todo. Para evitar esse cenário, deve-se encontrar um ponto ideal, onde o sistema não se sobrecarregue demais e onde os status não fiquem desatualizados por muito tempo. Além disso, a grande quantidade de dispositivos e a sincronização de tempo entre os nós também podem se tornar um grande problema para o NADA Monitoring Service.

Entretanto, o principal desafio se encontra ao tentar manter a segurança e privacidade no transporte dos dados e também no armazenamento dos mesmos, garantindo que aplicações que coexistem num mesmo hardware não possam ter acesso a dados que não lhe pertencem. Deve-se garantir também que aplicações externas não acessem dados privados de cada nó.

Para que a comunicação entre os nós se torne segura, foram idealizados dois mecanismos básicos, que são as Identities e o Ticket System. As Identities são basicamente “tokens” que podem ser associados unicamente a um nó, e esses tokens não podem ser duplicados ou ter sua informação facilmente extraída. Dessa forma, as mensagens enviadas por um nó são identificadas por esse token, que contém suas informações, garantindo a autenticidade da mensagem.

Já o Ticket System foi criado como sendo uma maneira de estabelecer uma conexão com um novo nó. Basicamente, quando um nó A pretende iniciar uma conexão com um novo nó B, esse nó B precisa requisitar ao sistema um ticket específico para o nó A, provando que B é confiável e possui status considerados como sendo válidos pelo sistema. Assim, pode ser estabelecida uma conexão segura entre os nós.

Por fim, para garantir que uma aplicação específica não

possa acessar informações de aplicações instaladas no mesmo nó, ou de dados privados que se encontram no mesmo local, utiliza-se a virtualização do hardware. A virtualização permite alocar recursos específicos para cada máquina virtual [6], e cada um desses sistemas virtualizados não possui conhecimento sobre outros sistemas sendo executados no mesmo local. Cada slice alocado para uma aplicação é uma nova máquina virtual, o que garante o acesso a apenas esse recurso dentro do nó.

IV. Possíveis Aplicações

Apesar da arquitetura oferecida pelos nano datacenters ser bastante flexível, existem algumas aplicações que podem obter melhor proveito das características dessa plataforma. Exemplos dessas aplicações são: vídeo sob demanda, distribuição de conteúdo gerado pelo usuário ou MMOG (*Massively Multiplayer Online Games*). Nesse artigo, será abordada a utilização da arquitetura em aplicações de vídeo sob demanda e dos games MMO.

A. Vídeo sob demanda

É possível criar um serviço de vídeo sob demanda sobre a plataforma NADA, utilizando suas características a seu favor. Nas próximas linhas, será descrito um exemplo sobre como pode ser criado um serviço nessa plataforma, e quais são as características favorecidas pela plataforma.

Inicialmente, pode-se definir um SLA (Service Level Agreement) onde o serviço de vídeo pode determinar, por exemplo, a quantidade de armazenamento desejada em determinado local geográfico, com condições restritas de atraso de envio do vídeo.

Assim, é preciso formular a requisição equivalente ao SLA e enviá-la à plataforma. Caso essas condições estejam disponíveis, é possível alocar esse recurso específico para a aplicação de vídeo sob demanda. Assim que o recurso for alocado, é necessário criar as credenciais corretas para que os desenvolvedores da aplicação acessem os recursos contratados. Além disso, os desenvolvedores também devem possuir a API necessária para manipular e controlar os recursos em si, e a comunicação entre os recursos.

Dessa forma, com as devidas APIs e credenciais, torna-se possível executar qualquer código, através do NADA Management Service. Entretanto, a partir desse momento, é responsabilidade dos desenvolvedores lidar com replicações, cache etc, assim como a comunicação entre os slices adquiridos.

Depois de ter o serviço sendo executado, é possível verificar métricas que são disponibilizadas pelo NADA Monitoring Service. Essas métricas podem ajudar a aplicação a tomar decisões sobre alocar ou liberar recursos, pois podem exibir informações sobre quantidade de acessos ao dia ou pontos geográficos que mais utilizam o serviço, por exemplo.

Além disso, existem dois pontos específicos das aplicações de vídeo sob demanda, que são favorecidos quando se utiliza a plataforma NanoDataCenters: a distribuição da popularidade de conteúdos e a dinâmica dessa popularidade ao longo do tempo.

À medida que a popularidade para determinados conteúdos não se mantêm constante, ou seja, alguns pontos geográficos costumam ter uma maior popularidade para conteúdos específicos, torna-se possível utilizar servidores cache próximos dessa localidade para distribuir esses vídeos com maior popularidade de forma muito eficiente. Além disso, torna-se simples adaptar o sistema à velocidade de mudanças de popularidade, pois basta mover os novos vídeos populares para servidores cache mais adequados.

B. Massively Multiplayer Online Games

Tendo como ponto de vista a utilização dos recursos oferecidos pelos nano datacenters, a grande diferença entre aplicações de vídeo sob demanda e jogos MMO tem a ver com a importância que cada aplicação dá esses recursos. No caso dos jogos, o estado das conexões entre os nós é uma informação de extrema relevância, pois é necessário manter uma conexão veloz, estável e constante entre os jogadores online. Já nas aplicações de vídeo sob demanda, os principais dados estão relacionados à quantidade de acesso aos vídeos disponibilizados. Entretanto, apesar dessas abordagens serem bastante diferentes, é possível verificar com um exemplo simples que a plataforma também pode ser bastante adequada ao cenário dos MMOGs.

Inicialmente, a aquisição de recursos e instalação da aplicação em si são tarefas bastante padronizadas pela plataforma, ou seja, esses passos são similares tanto no cenário de aplicações de vídeo sob demanda quanto no cenário dos jogos MMO.

Entretanto, assim que um jogador for autenticado num MMOG, ele será redirecionado para a instância mais próxima da mesma, procurando atingir a melhor velocidade de transmissão de dados entre os dois ou mais nós. Percebe-se que o objetivo nesse cenário não é lidar com a transmissão de conteúdo para uma grande demanda, e sim manter conexões rápidas e estáveis entre todos os nós envolvidos na aplicação.

V. Eficiência Energética

Essa sessão possui como objetivo comparar o aproveitamento de energia dos nano datacenters com os datacenters convencionais. Pretende-se demonstrar como a utilização de nano datacenters pode reduzir o consumo de energia gasto apenas com o compartilhamento de conteúdo.

As pesquisas foram conduzidas medindo-se a energia utilizada pelos servidores e por gateways. Assim, pôde se perceber que o consumo de energia dos dois tipos de equipamentos varia de acordo com a quantidade de dados transmitidos ao longo do tempo. Os gráficos abaixo indicam essa variação:

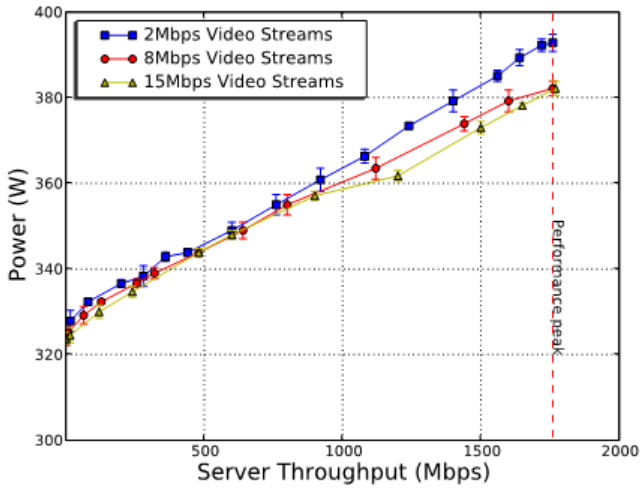

Gráfico 1 - Consumo de energia de servidores convencionais

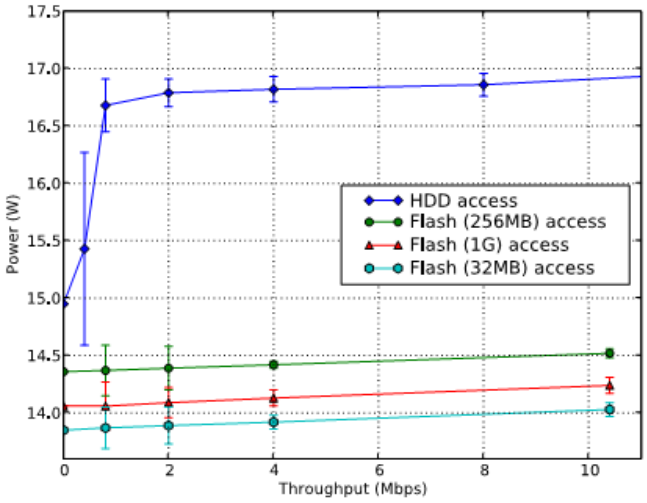

Gráfico 2 - Consumo de energia de um modem DSL

Apesar de serem necessários muitos modems para atingir a capacidade de distribuição de um único servidor, a utilização de modems como nano datacenters não afetaria nos gastos de energia totais, pois esses dispositivos costumam se manter ligados durante grandes períodos de tempo sem utilizar todo o seu processamento. Portanto, utilizá-los como nós de distribuição de conteúdo apenas otimiza o consumo de energia, pois como pode ser visto no Gráfico 2, uma maior utilização desse dispositivo não acarreta em um aumento significante do consumo de energia.

VI. Conclusão

O fato dos nano datacenters serem baseados em dispositivos com capacidade de processamento e armazenamento que se encontram nas "bordas" da rede, como por exemplo, set top boxes, não os tornam muito viáveis de serem implantados em países como o Brasil, onde essa tecnologia não está presente em todos os locais. Sua implantação não é viável porque, em regiões como essa, muitas das características positivas e interessantes oferecidas pela plataforma podem não ser aproveitadas, como por exemplo, a tolerância a falhas de nós.

Entretanto, nos locais onde esses dispositivos são mais numerosos, pode-se afirmar que a plataforma cumpre bem sua proposta de 'descentralizar' a distribuição do conteúdo, apresentando todas as vantagens da comunicação peer-to-peer em conjunto com a segurança dos datacenters comuns, em relação ao nível de serviço oferecido (SLA). Além disso, a energia consumida pelos nano datacenters acaba sendo muito baixa, em comparação com os datacenters convencionais. Devido à simplicidade da plataforma, onde o desenvolvedor tem controle de praticamente todo o hardware virtualizado, torna-se possível executar quase todo o tipo de aplicação, mantendo a plataforma muito flexível.

Referências

[1] FP7, 2008. "Projeto NanoDataCenters". Disponível em: ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/fire/projects-nanodatacenters_en.pdf.

[2] A. Hendry , 2008. "Set top boxes to revolutionise Internet architecture". Disponível em: http://www.computerworld.com.au/article/253324/set_top_boxes_revolutionise_internet_architecture/?fp=16&fpid=1.

[3] NanoDataCenters, 2010. "Combined Deliverable: D1.1 (System Design and Decomposition) and D3.1 (Draft Architecture Specification of security, privacy, and incentive mechanisms)" . Disponível em: http://www.nanodatacenters.eu/index.php?option=com_phocadownload&view=category&id=1:architecture&Itemid=66

[4] NanoDataCenters, 2010. "Deliverable 2.1 Measurement-based characterisation of application and user behaviour". Disponível em: http://www.nanodatacenters.eu/index.php?option=com_phocadownload&view=category&id=3:measurement&Itemid=66

[5] NanoDataCenters, 2010. "Deliverable D1.2: Evaluation of the energy efficiency of distributed vs. centralised content distribution". Disponível em:http://www.nanodatacenters.eu/index.php?option=com_phocadownload&view=category&id=1:architecture&Itemid=66#

[6] L. Cherkasova, 2005. "Measuring CPU overhead for I/O processing in the Xen Virtual Machine Monitor".