Abordagem em CDN

Lucas Begnini Costa[1]

[1]LARSIS - Departamento de Informatica – Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brazil

lucasbegnini@gmail.com, maziero@inf.ufpr.br

Resumo. *Este artigo tem como objetivo apresentar uma introdução a redes CDNs e sua segurança. Ao logo do artigo iremos entender a composição de uma CDN, seus mecanismos de busca e acesso e o que é necessário para discernir uma rede normal de uma CDN. Também iremos tratar de alguns aspectos de segurança da rede apresentando soluções reais e utilizadas.*

Abstract. *This article aims to present an introduction to the CDNs and their safety. At the end of the article we will understand the composition of a CDN, its search and access mechanisms and what is necessary to discern a normal network of a CDN. We will also address some aspects of network security by presenting real and used solutions.*

Sumário

Lista de Tabelas

Lista de Figuras

1. Introdução

Vivemos em um mundo rodeado de tecnologia, onde a cada dia somos surpreendidos com uma coisa totalmente inovadora, disruptiva. Uma era onde a internet foi responsável por atravessar mares, superar distâncias e até mesmo idiomas. Hoje se pode comunicar em tempo real com pessoas que estão em lados completamente opostos ao seu.
Hoje é possível que empresas estrangeiras muito distantes fisicamente, como China, India, EUA e etc, forneçam serviços para regiões mais remotas do mundo. Isso inclui servicos

de multimídia como armazenamento de fotos, de vídeos e até conteúdos de consumo instantâneo.
Esse encurtamento de distância pode parecer simples, mas vem de um sistema complexo que visa fornecer ao usuário final uma experiência agradável com conteúdos entregues de maneira satifatória mas sem, necessariamente, replicá-lo por todo o globo. O que nos leva a dizer que uma CDN (*Content Delivery Network*) é uma rede de distribuição de conteúdo que tem como objetivo fornecer ao usuário de aplicações globais uma experiência satisfatória na utilização de serviços, principalmente sob demanda. Podemos ver alguns desses serviços na figura 1.

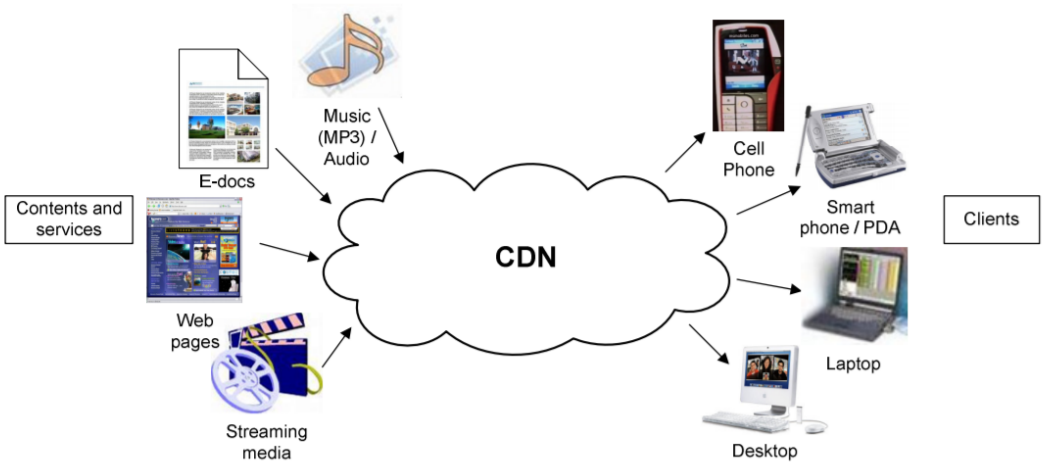


Figura 1. Fornecedores x Consumidores

Há vários serviços que se utiliza no dia a dia onde essa noção de CDN é completamente abstrata ao usuário final, como serviços de Video-On-Demand de empresas de TV, spotify, Amazon Prime Video e até Netflix, como é mostrado no artigo do [Adhikari et al. 2012].
Existem hoje diversas empresas que fornecem esse serviço ao redor do globo. Como Akamai, Limelight, Level 3 entre outras. Dessas redes a mais conhecida é a Akamai, que foi criada dentro do MIT (*Massachusetts Institute of Technology*), e tem como clientes empresas como Adobe, Airbnb, American Idol, Audi e muitas outras.
Essas três redes são hoje as principais fornecedoras de serviço de CDN da Netflix. Cada uma com uma característica e voltada para um público.
Nesse artigo vamos tentar entender um pouco mais a respeito de CDNs e também um pouco do seu sistema de segurança. Sendo assim, poderemos dividir o restante desse artigo em duas partes: A primeira será relacionada somente a composição da CDN, ou seja, tudo aquilo que é necessário para se constituir uma CDN. Na segunda etapa falaremos primeiro um pouco sobre segurança em um âmbito geral e depois aprofundaremos em seus conceitos voltados para CDN.

2. Composição de uma CDN

Para entendermos melhor uma CDN precisamos primeiro destrichá-la em vários pequenos pedaços para assim compreende-la em uma maneira global.

Uma CDN apesar de abstratamente ser vista como um mecanismo único, pode ser vista também como a soma de vários tipos de elementos, várias características que somadas e configuradas formará um mecanismo único e transparente aos usuários. Características essas que são organização, tipos de servidores, protocolos e tipos de conteúdo.

Organização - Quanto a organização uma CDN pode ser uma rede unicamente CDN ou uma rede *overlay*, que nada mais é que uma rede onde ela tenta abstrai as camadas de redes já existentes(como transporte, redes entre outras) e transforma-lá em uma rede puramente CDN.

Tipos de conteúdos - Os tipos de conteúdo que irão ser transportados dentro da rede são fundamentais para definir diversos aspectos de configurações que serão utilizadas dentro da rede. Como por exemplo, a forma de Cache que serão feitas os arquivos ou até mesmo a forma como vão ser distribuídos esses mesmos conteúdos, se serão distribuídos em conjunto ou em partes, como o caso de uma pagina HTML que possui um video para cada região do mundo.
Todos os outros pontos levam em conta primeiro o tipo de conteúdo para definir quais serão suas escolhas.
Os demais itens serão tratados nos próximos pontos. Tipos de servidores em 2.1 e protocolos de interações em 2.2

2.1. Tipos de servidores

Guardadas as devidas configurações físicas, tipos de memória, cpu ,entre outros, há dois possíveis tipos de servidores: Servidor de origem e servidor de ponta.

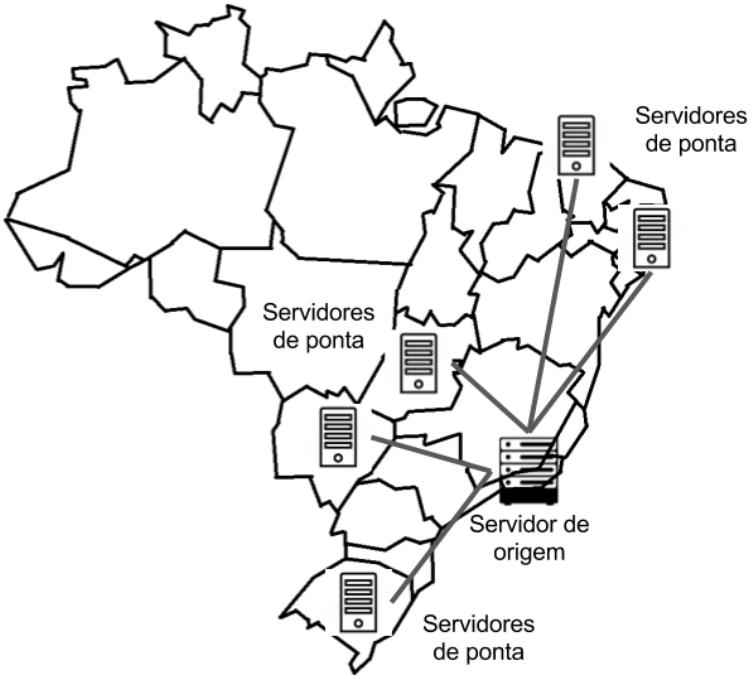
Servidores de origem são servidores onde toda a informação vai ficar baseada. É o lugar onde vão ser armazenados os conteúdos e o ponto principal do sistema.

É ele o responsável por, na ausência da informação perto do cliente, fornecer tudo o que o cliente necessita de informação. E em caso de um primeiro acesso é através dele que será feito o interfaceamento entre usuário e conteúdo.

Há uma necessidade intrínseca de que esse servidor tenha excelentes configurações físicas e ótimas regras de seguranças, *firewall* incluso, que possam garantir a integridade do sistema mesmo em caso de muitos acessos.

Servidores de ponta são servidores que mais próximos aos clientes. Parte do conteúdo do servidor de origem estará replicado nele. Dizemos em parte, pois não sabemos quais políticas de *cache* que será adotada pelos criadores da rede.
Podemos ilustrá-los conforme a figura 2.

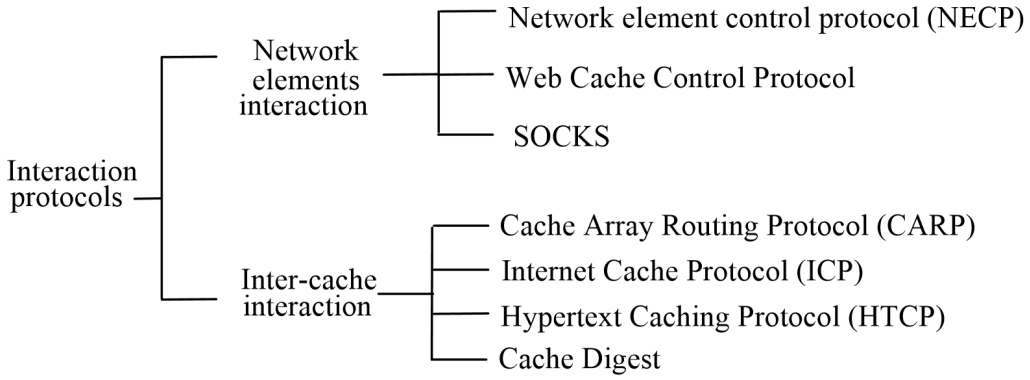
Figura 2. Tipos de servidores



Vale salientar que não necessariamente esses *status* de ser de ponta ou ser de origem são imutáveis. Em ambientes reais e comerciais um servidor de origem é também um servidor de ponta para um outro conteúdo. Isso é o responsável por tornar as grandes cdns completamente transparentes geograficamente perante aos seus clientes.

2.2. Protocolos de interações

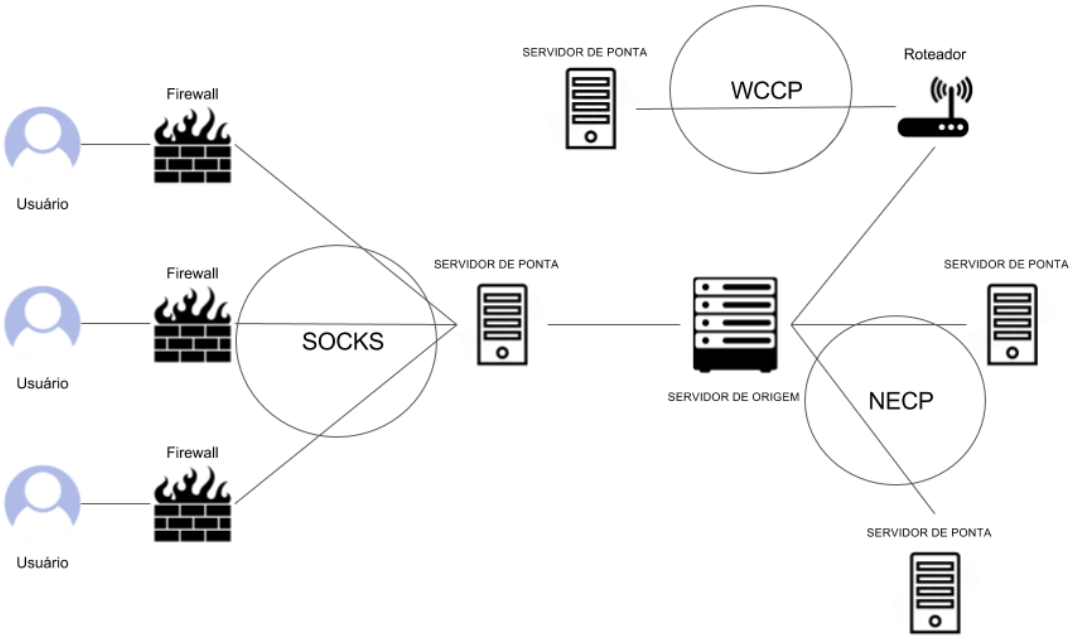
Figura 3. Tipos de relacionamentos



Os protocolos de interações podem ser divididos em duas partes: Protocolos de interações de elementos da rede e Protocolos de interações entre os servidores de cache da CDN.

2.2.1. Interações dos elementos da rede

Figura 4. Tipos de protocolos de iterações



Dentro dos protocolos de interações dos elementos de rede podemos verificar que cada um possui sua especificidade e funcionalidade bem definida, como podemos ver na figura 4 , tentando proteger o não só a rede mas também o usuário, o servidor, os roteadores e a comunicação entre os mesmos.

Socks é um protocolo desenhado para um *framework* para aplicações cliente-servidor que rodem em TCP ou UDP e utilizem os servicos do *firewall*. Sua principal funcionalidade é gerar um túnel de comunicação seguro entre cliente e servidor.

NECP é um protocolo leve de comunicação que serve para sinalizar o tráfico entre os elementos da rede.

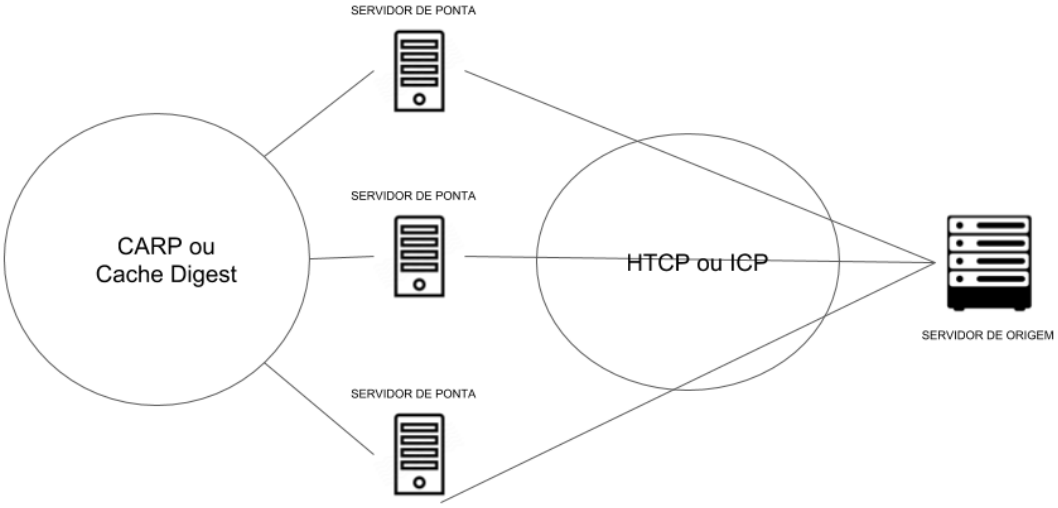
WCCP serve para iteraç oes entre roteadores e *web-caches* e também para transporte entre os mesmos.

2.2.2. Interações de cache

Os protocolos de interações de cache são protocolos que organizam as trocas de informações entre os servidores, ou seja, é ele que dita como irá funcionar a distribuição da informação dentro da rede.

Conforme vimos na figura 3, e segundo [Pathan and Buyya 2007], existem 4 tipos de protocolos aplicados nessa circunstância como podemos ver na figura 5

Figura 5. Tipos de protocolos de iterações



2 deles são: HTCP - Hypertext Caching Protocol e ICP - Internet Cache Protocol que são concorrentes entre sí e tem como funcionalidade controlar o fluxo de informação entre os caches. Sendo através deles que se controla o que irá para um determinado servidor de ponta, por exemplo. Falaremos mais sobre ambos em 2.2.3 e 2.2.4 respectivamente.
Existe também os protocolos CARP - Cache Array Routing Protocol e Cache Digest, também concorrentes, servem para controlar o conteúdo existente dentro de cada servidor e saber onde estão os outros conteúdos. Falaremos mais sobre ambos em 2.2.6 e 2.2.7 respectivamente.

2.2.3. HTCP

Como dito anteriormente o HTCP, Hypertext Caching Protocol, é um protocolo de interação entre os caches, suas principais caracteristícas são:

- Protocolo para descobrir Caches HTTP;
- Suporte ao HTTP 1.0;
- Permite incluir cabeçalhos nas respostas;
- Podem ser enviados via TCP/UDP;
- Devem ser resilientes a falhas.

2.2.4. ICP

Já o ICP, Internet Cache Protocol, é um protocolo muito mais leve que possui as seguintes caracteristícas:

- Protocolo de mensagem leve;
- Utilizado para comunicação de Caches;
- Utiliza consultas para determinar localização mais apropriada;
- Suporte ao HTTP 0.9;
- Comunica-se com caches vizinhos;
- recebe MISS ou HIT como resposta;
- Enviado via UDP;
- Falha por timeout indica caminho quebrado;
- Fornece informações para balanceamento através das medidas de perda.

2.2.5. HTCP x ICP

Analisando os dois protocolos HTCP e ICP, podemos fazer um quadro comparativo entre eles e colocá-los da seguinte maneira(figura 6):

Figura 6. HTCP x ICP

Serviços	HTCP	ICP
Envio TCP	✓	✓
Envio UDP	✓	
Suporte HTTP 1.0	✓	
Permite enviar apenas cabeçalho	✓	
Monitora caches remotos	✓	
Permite monitoramento de falhas		✓

2.2.6. CARP

CARP - Cache Array Routing Protocol
Protocolo de armazenamento distribuído baseado em uma lista conhecida de proxies fracamente acoplada e uma função hash para dividir o espaço URL entre esses proxies.

- Cliente HTTP pode enviar requisição à qualquer proxy da lista.

2.2.7. Cache Digest

Protocolo de intercâmbio e formato de dados entre caches.

- Fornece um resumo dos conteúdos na resposta;
- Soluciona os problemas de congestionamento e timeout;
- Torna possível determinar se um servidor possui em cache um conteúdo;
- Executado via HTTP ou FTP;
- Contém tempo de expiração na resposta;
- Pode ser utilizados para eliminar redundância.

2.3. Seleção e entrega de conteúdo

Dentro de uma CDN temos que nos preocupar com a forma como esse conteúdo vai catalogado, armazenado e distribuído dentro da rede, o que vimos no item 2.2, como também temos que nos preocupar como esse conteúdo vai chegar até o cliente (usuário) da forma mais otimizada possível, ou seja, o servidor o qual vai fornecer as informações para ele será o mais perto ou mais rápido.

Temos que destacar também a importância da otimização do fluxo de informação pela rede. Visto que quanto maior o tráfico de informação pela rede significa que a informação está mais distante do usuário e também que vai ter um custo maior pela troca intensa de informação.

Segundo [Krishnamurthy et al. 2001], na tentativa de otimizar o redirecionamentos de URL para o usuário se sacramentou dois tipos de técnicas de redirecionamentos:

- Full - site
- Partial - site

2.3.1. Full - site

Na técnica de Full-site todo o conteúdo é entregado ao usuário de um servidor ponta único. Ou seja, o usuário faz uma requisição ao servidor principal, onde o mesmo processa um algoritmo de roteamento para encontrar o servidor ponta que melhor se enquadra como resposta, e então retorna ao usuário o endereço onde então será consumido por fim todas as informações requisitadas.

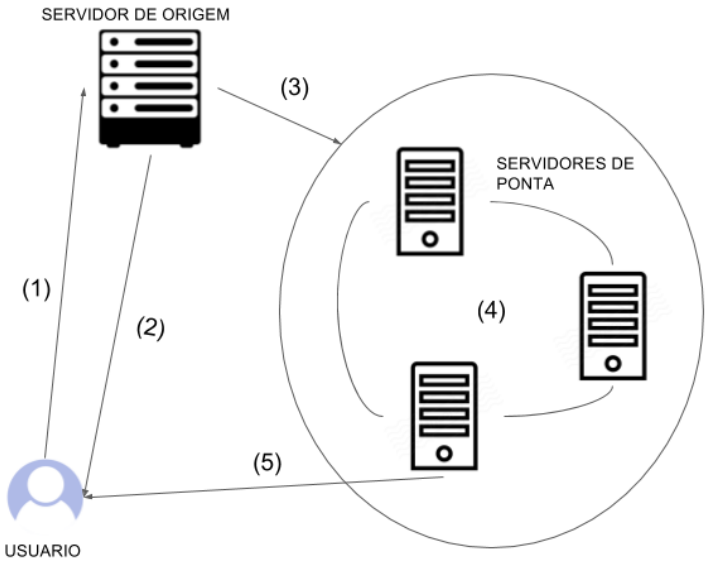
É importante salientar que essa técnica é amplamente utilizada por serviços que fazem pouco uso de dados da rede. Uma página estática da web, por exemplo, se encaixaria perfeitamente nesse contexto. Visto que possui baixo grau de modificações e seu tamanho é pequeno perto de outros tipos de mídias que circulam na web.

2.3.2. Partial - site

Já redirecionamentos do tipo Partial-sites os servideores principais retornam para o usuário uma parte do conteúdo e disparam, automaticamente, um algoritmo de roteamento para encontrar o restante da informação e retornar ao usuário. Conforme podemos ver na figura 7

Figura 7. Entrega de conteúdo

(1)	Requista rota
(2)	Retorna um index.html
(3)	redireciona a requisição para o provedor de CDN
(4)	Acha o melhor servidor de ponta
(5)	Envia o conteúdo ao usuário



Nela podemos ver que todo o processo acontece em, basicamente, 5 etapas. (1) o usuário faz uma requisição ao servidor principal, depois, em (2) o servidor principal retornar um html com as principais informações e dispara automaticamente (3) um processo de roteamento (4) para buscar o melhor servidor e retornar (5) para o usuário os conteúdos.

Entretanto há em (4) diversas formas de fazer esse roteamento quanto a distribuição do conteúdo pela rede e quanto a aglomeração desse conteúdo dentro do servidores de cache.

Tipo de distribuição nada mais é do que a forma como o conteúdo vai ser disperso na rede, como esse conteúdo vai se aproximar do nó que está mais perto do usuário.

Os tipos de distribuição mais frequentemente utilizados são:

- Empírico
- Popularidade

Empírico trata, como o próprio nome diz, de uma forma de distribuição sem nenhum conhecimento específico a respeito, utilizando-se apenas um conhecimento experimental de onde seria melhor posicionado o conteúdo.

Em um esquema baseado em popularidade a distribuição é feita conforme a notoriedade. Ou seja, quanto mais a requisição do conteúdo mais ele vai ficar nos servidores de ponta perto do usuário.

Ambos esquemas não são necessariamente excludentes, pode-se inicialmente aplicar a forma empiríca para gerar dados a respeito da distribuição e depois utilizar esses dados para aplicar o modo de popularidade.

Também existe as formas de aglomerações de conteúdo. Isso existe porque os conteúdos podem ser conjuntos de objetos ou objetos independentes. As formas de aglomerações são:

- Objeto
- Conjunto de objetos

Aglomeração por objeto vai juntar os objetos mais selecionados e distribuí-los individualmente entre os servidores. Já por conjunto de objetos ele vai separá-los em grupos e distribui-los em conjuntos.

Podemos exemplicar da seguinte forma: Em um site temos vários elementos, temos o HTML, temos o CSS e temos a mídia de um vídeo qualquer. Podemos separar da seguinte forma: temos 3 elementos onde 2 deles são altamente dependentes(HTML e CSS) e temos u que pode ser diferente em cada região do país, que é o vídeo.

Então podemos misturar as formas de aglomerações. Para o HTML e para o CSS agrupamos em conjuntos de objetos; e para o vídeo fazemos a aglomeração por objeto,já que será distribuído de maneira independente nos servidores.

Analisando os tipos,perceber-se que nenhuma das opções são excludentes entre sí, pode-se combinar quaisquer tipo de distribuição e aglomeração e as escolhas vão depender das necessidades de cada aplicação.É necessário uma análise minuciosa de cada aplicação para chegar em um veredito da melhor abordagem.

2.3.3. VOD

VOD - Video on Demand - Segundo [Garfinkle 1996], é um sistema de que proporciona uma interface de comunicação com o usuário de produtos disponíveis de uma estação central remota.

É um sistema muito utilizado por operadoras de conteúdo onde é o cliente quem decide o horário que irá consumir determinado conteúdo.

Isso permite às pessoas que tinham que esperar o horário certo para consumir determinado conteúdo, poderem a qualquer momento e em qualquer lugar fazer uso do mesmo.

Microsoft Smooth Streaming É um sistema de consumo de vídeo onde a qualidade do vídeo transmitida vai ser definida conforme a qualidade da banda disponível. Clientes que possuem alta disponibilidade de banda terão maior qualidade do vídeo.

Para conseguir tocar um vídeo nesse formato o player do usuário tem que ser capaz de interpretar um manifesto que contém dentro de outras coisas, o caminho de onde está localizado a mídia desejada.

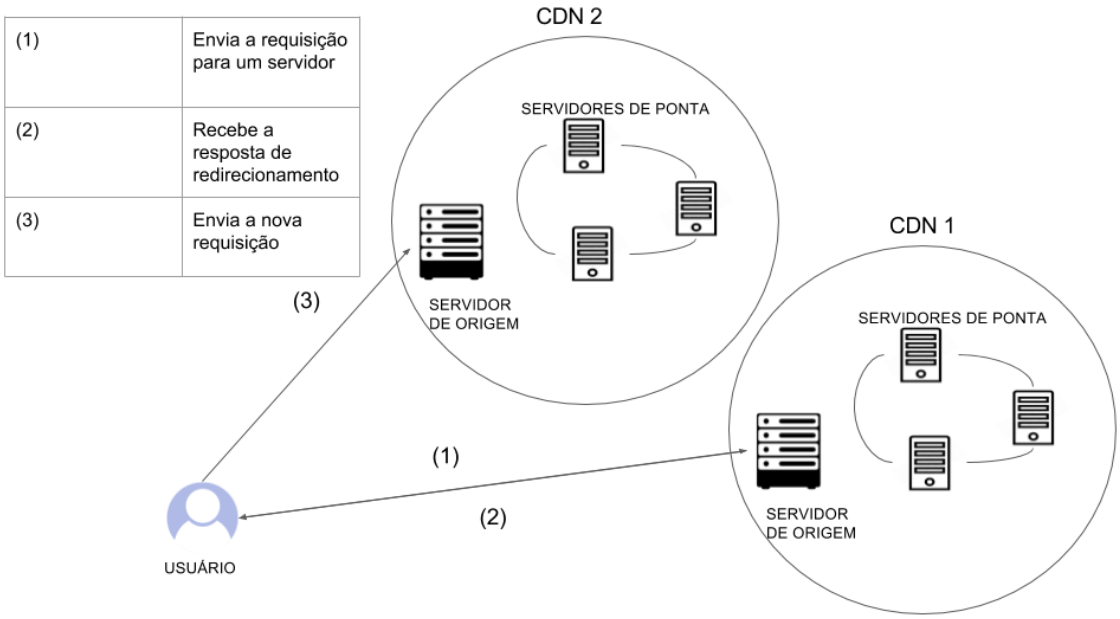
Já para consegui fazer a transição de qualidade o vídeo é quebrado em pequenos pedaços chamados de ”chuncks”. Então, conforme é verificada um aumento ou decremento da banda disponível é só o player começar q consumir chuncks de qualidade diferente da atual e assim o usuário já passará vê a diferença significante na tela.

No artigo [Zambelli 2009] podemos vê todas as aplicações e implicações dessa forma de consumo de mídia.

2.3.4. Exemplo

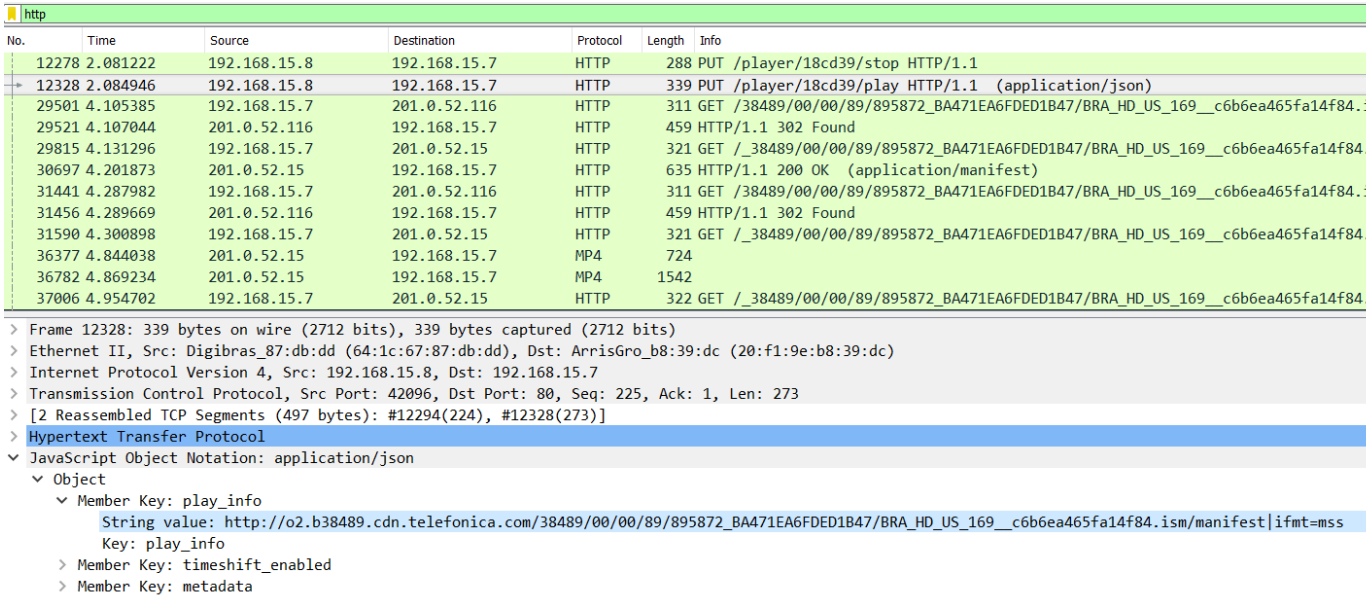
Agora vamos ilustrar com um exemplo mais prático. Utilizaremos uma requisição de uma URL de um VOD(Video On Demand) onde a parte de roteamento é feita no usuário. A aplicação do usuário faz uma requisição à CDN e enquanto o cabeçalho da resposta não for 200 ele vai ler um campo dentro do cabeçalho de resposta e fazer uma nova requisição. A figura 8 ilustra a situação.

Figura 8. Redirecionamento de CDN



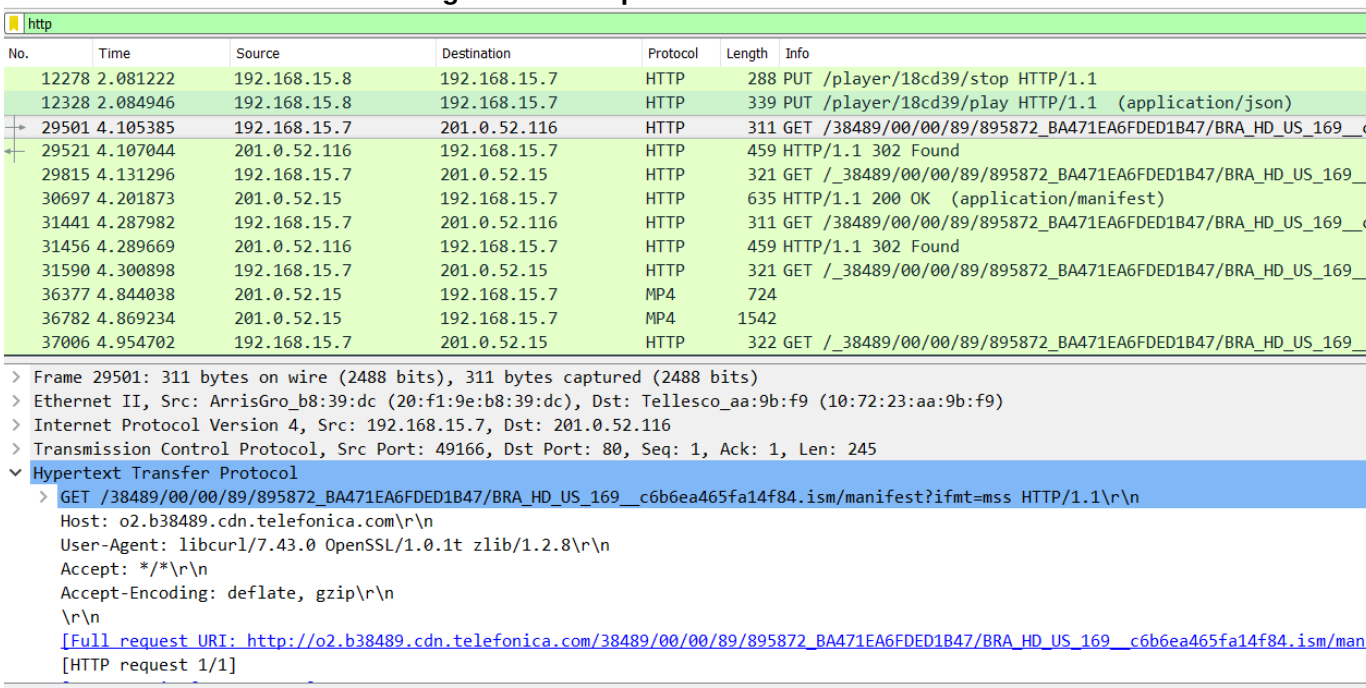
Como podemos observar na figura 9 a aplicação manda para o player a url associada ao VOD que é responsável por responder com o caminho oficial, ou intermediário, do manifesto. Na figura 9 ele está localizado no campo ”play_info”da requisição.

Figura 9. exemplo VOD



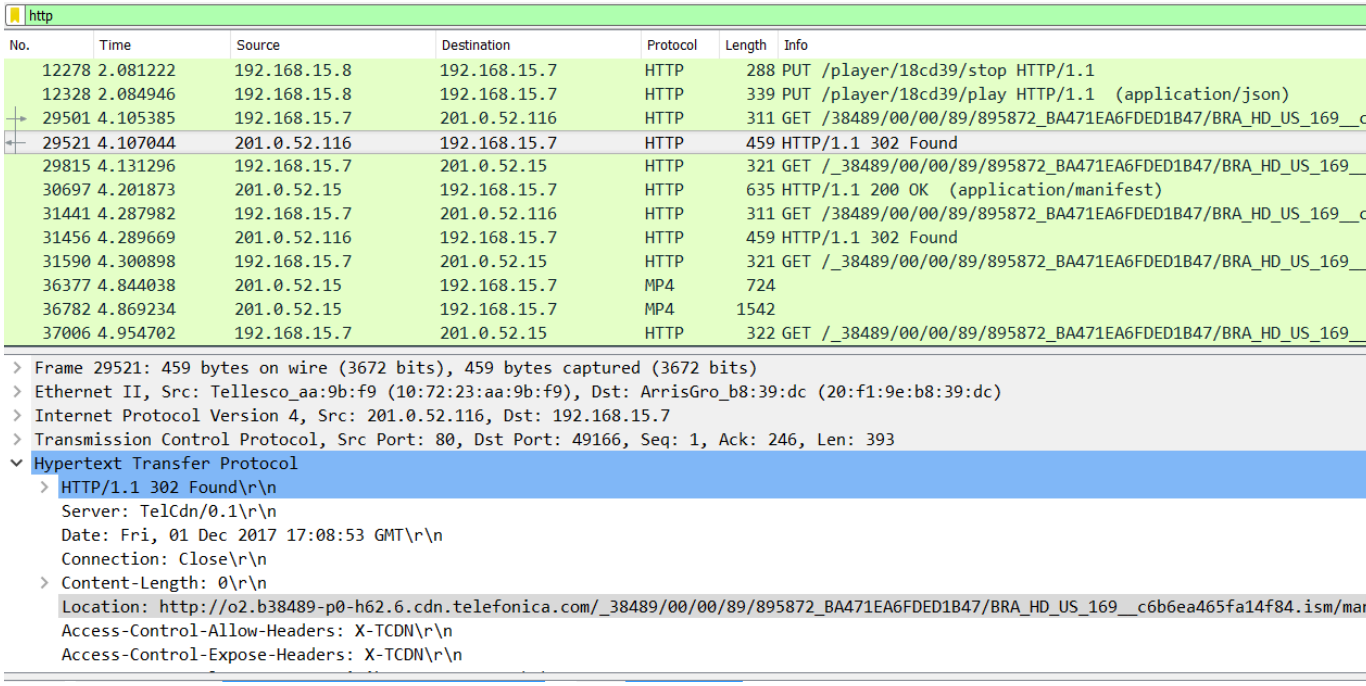
O player, por sua vez, como responsável por enviar as requisições, envia ao servidor principal a requisição do manifest do VOD que ele quer tocar. Figura 10.

Figura 10. exemplo VOD



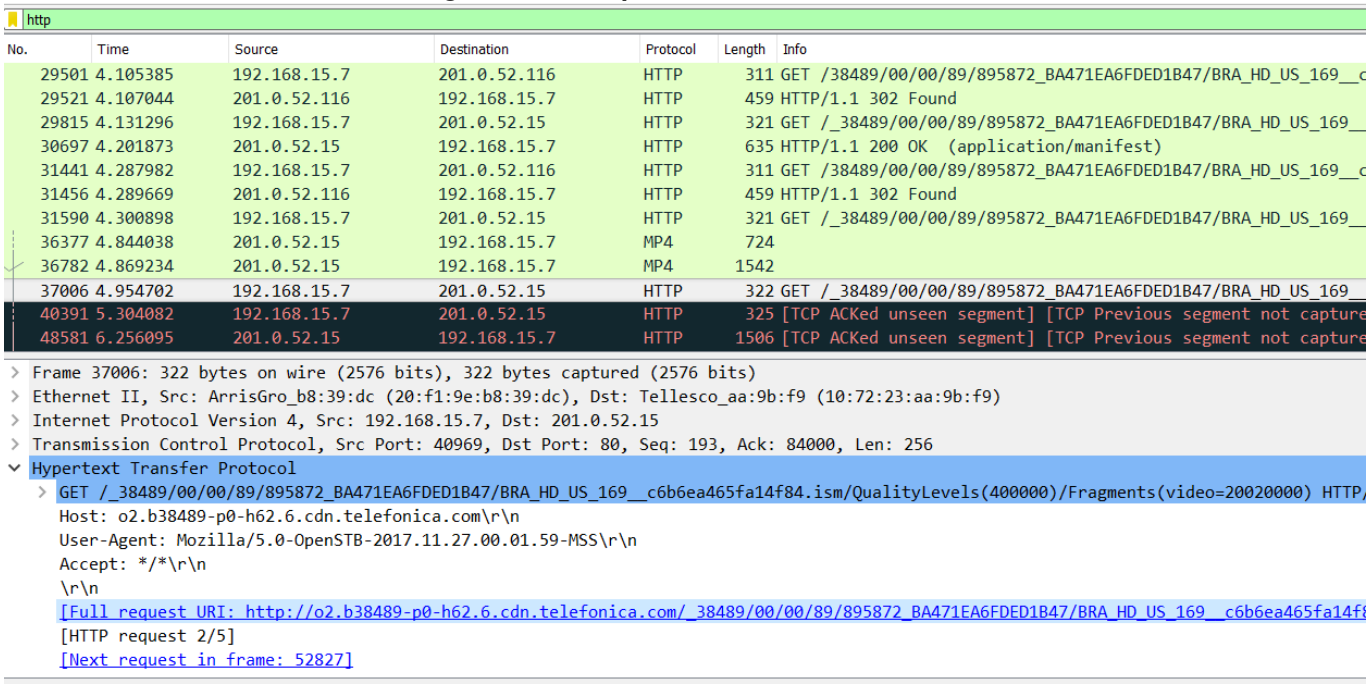
Na figura 11 vemos a CDN, que já fez um roteamento para uma outra rede(ou servidor), que retornou com um código 302 indicando que o retorno da requisição não é a resposta final. Dentro do cabeçalho da resposta tem um campo chamado ”location”o qual é responsável por retornar o endereço final (ou intermediário) para nova requisição. Esse processo será repetido até o código de retorno da requisição for 200.

Figura 11. exemplo VOD



Ainda na figura 11 podemos observar que ele recebe um 200 OK da requisição em seguida do 302 Found. O que simboliza que aquela URL é a URL final e o player pode utiliza-la como base do manifesto.

Figura 12. exemplo VOD



Em 12 vemos o player começando a consumir os ”chuncks”e por consequência tocando o vídeo.

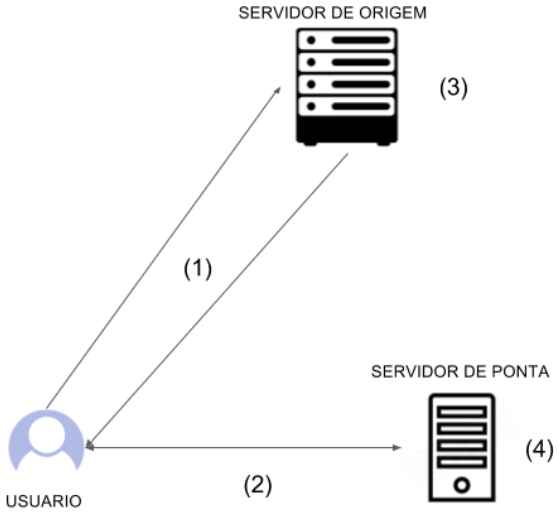
3. Segurança de uma CDN

Segundo [Ferreira 2004] segurança é definido como um conjunto de ações que e dos recursos utilizados para proteger algo ou alguém, ou, o que serve para diminuir os riscos ou perigo. Portanto, segurança é um tema extramente abrangente e complexo. Podemos falar de tópicos como segurança física, social, de um sistema ou até mesmo de um grupo de servidores interconectados, que é o caso da CDN, e ainda coloca-los todos dentro do mesmo grupo. Ou seja, podemos tratá-los de forma separada ou analisando o conjunto todo.

Ainda dentro de Segurança de CDN podemos tratar de pontos principais, onde o vazamento é ponto crucial e mais lógico de ser atacado.

Figura 13. Segurança

(1)	Autenticação de usuário
(2)	Autenticação de conteúdo
(3)	Segurança do Servidor de origem
(4)	Segurança do Servidor de Ponta



Como vemos na figura 13 podemos observar se tem quatro principais pontos a serem garantidos em uma rede. O primeiro trata-se da autenticação de usuário que é capacidade do sistema de garantir que aquele usuário que está acessando o conteúdo tem mesmo os requisitos para acessá-lo. Ou seja, é mesmo um usuário do sistema. Por ser um tema bem abrangente será tratado com mais profundidade em 3.3.

Em segundo lugar temos a autenticação de conteúdo. Que é a garantia que o conteúdo acessado é o mesmo buscado pelo usuário, que o usuário tem acesso a ele e

também é um conteúdo que faz parte da rede de disponibilidade da CDN(que não é um conteúdo inserido por terceiro sem autorização prévia). Tudo isso tem que ser minuciosamente tratado e averiguado antes de retornar ao usuário. Tudo isso será abordado em 3.4.

Nos dois outros, (3) e (4), trata-se da segurança dos servidores. Aí podemos falar de segurança física e lógica. Tem-se que levar em conta a forma de acesso de cada um na hora de mensurar os aspectos de segurança de cada um deles.

No servidor de origem se tem um acesso via DNS, com um endereço ”legível”. Esse acesso se dará muitas vezes por várias parte do globo, tendo que deixá-lo disponível , portanto, vulnerável à diversos tipos de ataque, o quê o torna extremamente complexo na hora de definir suas regras de segurança, não basta apenas subir um *firewall* bloqueando múltiplos acessos, é preciso saber exatamente os tipos de aplicações que serão tratadas para assim começar a desenhar as regras de *firewall* que serão aplicadas, e também, na maioria dos casos, será necessário a implementação de protocolos de proteção dentro da própria aplicação que rodará no servidor.

Já no servidor de ponta a preocupação com o acesso via DNS já é uma coisa a menos, visto que muitas vezes o acesso funcionará via redirecionamento do servidor de origem, sendo o controle de endereço feito pelo o mesmo. Mas há diversos outros aspectos que tem que serem levados em conta, como parte da autenticação de conteúdo para o usuário, que nada mais é que garantir que o usuário para o qual está sendo enviado o conteúdo tem acesso ao mesmo. Outra grande preocupação é quanto a sua disponibilidade, pois é necessário que o mesmo esteja ”de pé”quando for requisitado conteúdo e que durante o processo de transferência, caso haja alguma interrupção a mesma seja informada ao servidor de origem e o usuário transferido para outro servidor sem que haja muitos danos à experiência. Isso sem contar nas enumeras preocupações que se deve ter com servidores. No livro [Stallings 1995] se pode aprofundar um pouco mais nessas questões de segurança na *web*.

Mas para se aprofundar um pouco mais dentro dos conceitos de segurança de usuário e conteúdo de uma CDN é necessário primeiro conhecer um pouco sobre protocolos AAA e ter as definições de segurança muito bem esclarecidas, pois assim esses conteúdos o serão mais digeríveis. Ambos temas serão tratado em 3.1 e 3.2 respectivamente.

3.1. protocolos AAA - Authentication, Authorization and Accounting

A associação dos protocolos de autenticação, autorização e auditoria em um termo, protocolos AAA, se deu porque na maioria dos sistemas seguros esses três protocolos são extremamente necessários e amplamente utilizados. em [Metz 1999] se pode aprofundar melhor. Mas vamos há uma rápida síntese dos mesmos.

Autenticação - Verifica a identidade digital do usuário de um sistema, que segundo [Metz 1999], envolve a validação da identidade do usuário final afim de permitir seu acesso à rede.

Esse procedimento é baseado na apresentação de uma identidade junto com uma ou mais credenciais, que podem ser senhas, certificados digitais e etc.

Autorização - É a concessão de uso para determinados serviços. Essa etapa só é acionada mediante a autenticação prévia de um usuário. Ela leva em conta a identidade, o serviço requerido e o estado atual do sistema. Muitas vezes ela trabalha com a utilização de filtros para retornar que tipo de protocolos ou aplicações são suportadas.

Na maioria dos casos os protocolos de autenticação e autorização trabalham juntos. Onde um depende diretamente do funcionamento do primeiro. Em ambientes de VOD, como se foi apresentado em 2.3.4, essa etapa normalmente é verificada nos servidores de ponta, os quais são responsáveis por fornecer o conteúdo ao usuário.

Auditoria - A útima etapa do processo é relacionada a coleta de informações de tráfego utilizado pelo usuário.

Há dois tipos de coletas: coletas em tempo real e coleta *batch*. A primeira se refere a informações captadas em tempo de uso, ou seja, enquanto o usuário faz uso da rede essas informações são enviadas. Já a segunda se refere a uma coleta que é primeiro armazenada e enviadas posteriormente.

São essas informações que são utilizadas pelas operadoras para cobrança e taxação do serviços de VOD.

3.2. Definições de segurança

Como [O'Gorman 2003] fala em seu artigo as definições relativas aos sistemas e método em segurança são definidas como forte ou fraca. Isso acontece pois quando usamos termos relativos, segundo o mesmo, somos claros na mensagem que queremos passar a respeito da informação. Por exemplo, quando dizemos que uma porta com uma trava é mais segura, mais forte, que uma porta que não possui trava alguma, relativamos a segurança da trava tornando claro a mensagem que por mais que seja mais segura não é impossível de ser quebrada.

Por mais que a relativização muitas vezes seja vista com maus olhos no campo da informática a garantia de um sistema inviolável é quase impossível e quando possível altamente custoso.

Como podemos ver no livro do [Stuttard and Pinto 2011] há diversas maneiras de se burlar um sistema web por exemplo, mesmo sendo um sistema altamente visado e onde pessoas se preocupam o tempo todo em protegê-lo. Mas um ponto comum de quase todo ataque é o quê conhecemos como engenharia social. A entrada sempre acontece por meio de pessoas, ou fragilidade dessas pessoas.

É muito difícil mensurar segurança em termos absolutos. Uma forma de mensurar é através da força e da fraqueza de um sistema. Um sistema forte é aquele que o custo de atacar é muito maior que o ganho o atacando. Ou seja, o trabalho para consegui a informação vai ser tão grande que não compensará. Um exemplo é em [Biryukov et al. 2010] a alta complexidade e tempo para se quebrar uma chave AES-256.

Paralelamente à isso temos a fraqueza de um sistema. Essa é o oposto da força. Ou seja, o custo de ataque é menor do o ganho obtido com os resultados. E de que custo podemos entender não só dinheiro como também tempo, potencial para punição criminal e etc.

Mas focando em aplicações que utilizem CDN como distribuição podemos elencar os principais pontos de sua segurança conforme analisado por [Pomelo 2009].

Precisa-se primeiro listar as principais premissas para se garantir que um sistema de distribuição de multimídia, como o netflix, possa ser considerado fortemente seguro.

- Apenas os usuários com o devido acesso podem acessar determinados conteúdos;
- Os usuários não podem compartilhar suas informações com outros;
- Os conteúdos só podem ser acessados para regiões específicas. Usuários americanos só podem acessar conteúdos americanos.
- O conteúdo não pode ser redistribuído e reutilizado sem verificar novamente as credenciais de acesso.

Quanto a esses itens podem ser tomadas diversas precauções. Mas partindo da ideia do esforço mínimo com o máximo de ganho, podemos destacar algumas atividades que podem cobrir esses casos de usos e deixar o sistema seguro. Precauções essas que são:

- Fazer a requisição de autenticação de usuário momentos antes de tocar o vídeo;
- Limitar o número de acessos simultâneos ao serviço por usuário;
- Bloquear acessos de IPs de fora do *range* da região delimitada;
- Encriptar o conteúdo;
- Promovendo um único par de chaves entre conteúdo e *device*.

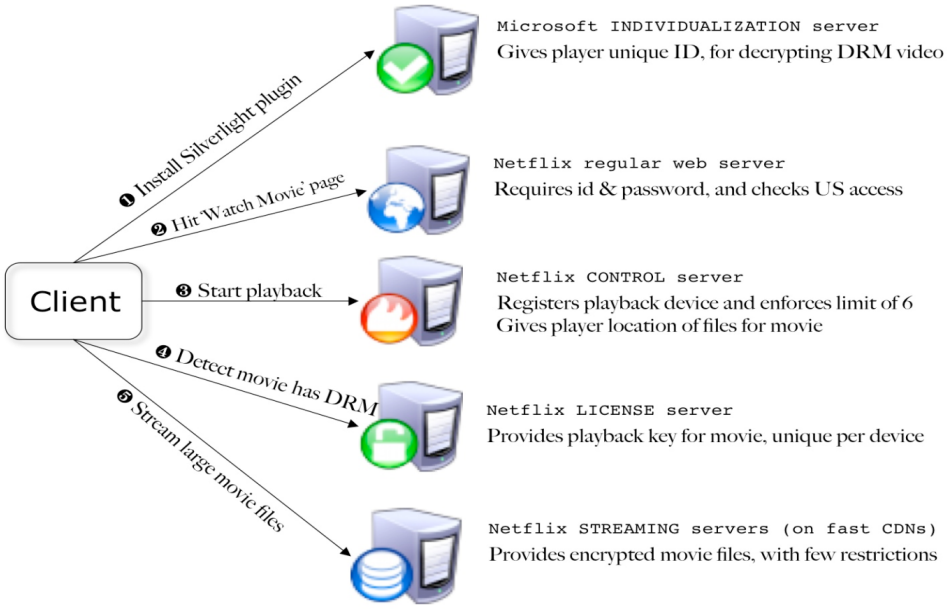
Essas são apenas algumas ações que se pode tomar para evitar vazamento de informação desejada em uma CDN voltada para a distribuição de conteúdo.

Para garantir que esses pontos sejam de fato cobertos a netflix utiliza de um esquema, explicado em [Pomelo 2009], que conta com quatro servidores distintos que possuem objetivos bem especificados.

- *Individualization server*;
- *Web services*;
- servidor de controle;
- e servidor de *streaming*.

Os objetivos desses servidores podem ser verificados na figura 14:

Figura 14. Rede de servidores netflix



3.3. Autenticação de usuário

A autenticação de usuário envolve a a certeza que o usuário que está consumindo o conteúdo ou tentando acessar a rede é o mesmo usuário que tem acesso à aplicação. E o que acontece quando o usuário faz login somente uma vez(em uma página)? Como garantir que é o mesmo usuário?

Bom, a forma de armazenamento vai depender muito de onde está sendo acessada a aplicação. Se for de um desktop, dentro de um navegador, normalmente vai ser armazenada dentro do *cookies* do navegador. Se for de uma aplicação mobile, ou até mesmo de uma aplicação desktop (sem ser um navegador), normalmente fica em alguma pasta temporária que é guardada pela memória *flash*. Então significa que uma vez feito login ele fica pra sempre armazenado?

Não. Essas autenticação são temporárias, muitas vezes com tempo de expiração definido pelas próprias aplicações, e requerem renovação da licença ou reautenticação do usuário.

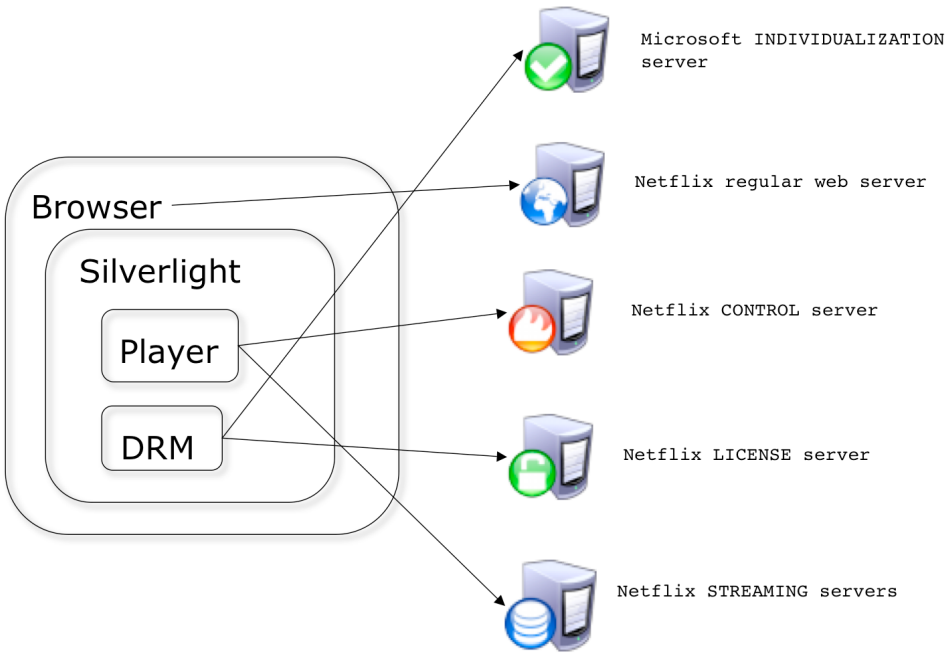
Em todos os casos, há diversas formas de autenticação dentro de sistemas computacionais. Há autenticação por login(normalmente email) e senha, por biometria, *token* entre outras. Mas em aplicações *web* login/senha são as mais utilizadas.

As etapas de verificação de quantidade de aparelhos e a origem dos IPs de requisição normalmente são feitas aqui.

Nesse processo é importante salientar também a autenticação dos *devices*. Nesse processo é muito importante que os *devices* sejam limitados para que não haja vazamento de usuários utilizando múltiplas sessões em mais de um dispositivo, quebrando assim a regra de que os acessos dos usuários não podem serem compartilhados.

Na figura 15 podemos ve como funciona um pouco do ciclo de autenticação dentro daqueles servidores citados anteriormente.

Figura 15. Ciclo de autenticação



3.4. Autenticação de conteúdo

Dentro de autenticação de conteúdo há uma gama de validações que são ou que podem ser utilizadas, [Wein et al. 2012] e [Leighton et al. 2007] são duas delas.

Aqui iremos focar em uma chamada *control DRM-encoded*, que é o método mais utilizado hoje em dia. Utilizado inclusive por sistemas como Netfilx e VOD de operadoras de TV.

Esse processo está intrinsecamente ligado ao processo de autenticação de usuário explicado anteriormente. Nele o conteúdo passa por uma série de validações antes de tocar.

A primeira etapa do processo para tocar o conteúdo é a etapa de ***License Acquisition***. Nela, apresentado por [Pomelo 2009], antes de tocar é feito:

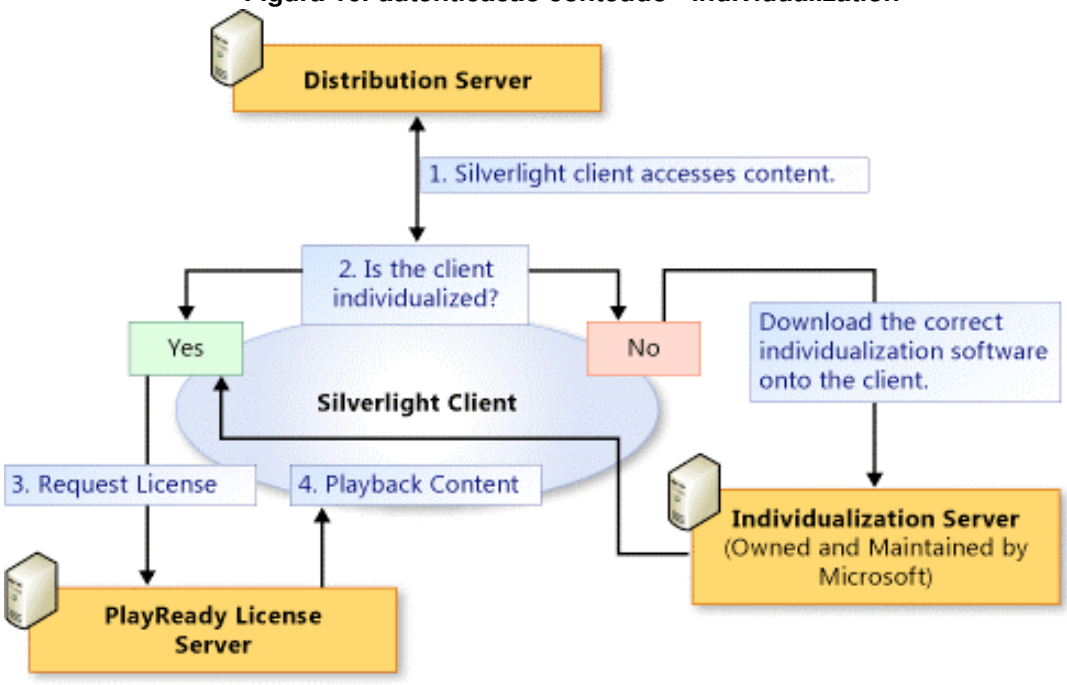
- Passo 1 - A aplicação requisita do servidor um pedaço do conteúdo. Este o retorna com o conteúdo criptografado.
- Passo 2 - O cliente lê o cabeçalho do conteúdo criptografado e determina que aquele conteúdo é criptografado. Para descriptografar o conteúdo é necessário que este tenha recebido a chave do servidor de licença. Caso não tenha recebido ele envia uma requisição para adquiri-la. Se for a primeira vez que é feita a validação da chave ele passa por um processo de *individualization*. Que será explicado logo mais a frente.(figura 16).
- Passo 3 - O cliente recebe a licença do servidor, que antes de enviar a chave faz todo o processo de autencidade necessário.
- Passo 4 - O cliente após receber a chave pode tocar o conteúdo. Isso se ele estiver licença para tal, é claro.

No passo 1 fica claro entender quando em 2.3.3 explicamos como é composto um VOD e qual o seu fluxo de comportamento.

No passo 2 o servidor de licença é o servidor que faz parte do rol de servidores citados em 3.2. O disparo do fluxo do *individualization* é feito pela própia aplicação do cliente, que verifica caso o cliente não possua nenhuma licença ele aponta para um terceiro servidor, onde esse vai retornar com uma licença individual que será utilizada na hora de validar e tocar conteúdo. O fluxo do processo é apresentado na figura 16.

Todo esse processo de troca de informações é feito utilizando trocas de chaves assimétricas públicas garantindo assim a segurança e procedência da informação.

Figura 16. autenticacao conteúdo - *individualization*



Após ***License Acquisition*** o *player* está apto a tocar o conteúdo. A comunicação agora é feita somente com o servidor de conteúdo. Recebendo o conteúdo criptografado e tocando através do processo de ***License Acquisition***.

4. Conclusão

Com todas as idéias apresentadas aqui fica claro que CDN, por mais que para o usuário final pareça simples, requer uma série de cuidados e atenção por toda a sua extensão. Tendo que se importar com sua extensão física e também lógica.

A segurança da informação ainda é a sua principal fonte de pesquisa. Principalmente pelo aumento de serviços que fazem o uso de serviços *On-Demand*. É necessário garantir não só a privacidade do usuário como também que conteúdo só chegue aos usuários que fazem mesmo uso do serviço. Evitando o vazamento de informações indevidas e indesejadas.

5. References

Referências

Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., and Zhang, Z.-L. (2012). Unreeling netflix: Understanding and improving multi-cdn movie delivery. In *INFOCOM, 2012 Proceedings IEEE*, pages 1620–1628. IEEE.

Biryukov, A., Dunkelman, O., Keller, N., Khovratovich, D., and Shamir, A. (2010). Key recovery attacks of practical complexity on aes-256 variants with up to 10 rounds. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 299–319. Springer.

Ferreira, A. B. d. H. (2004). Novo dicionário aurélio da língua portuguesa. In *Novo dicionário Aurélio da língua portuguesa*.

Garfinkle, N. (1996). Video on demand. US Patent 5,530,754.

Hsiang, H.-C. and Shih, W.-K. (2009). Improvement of the secure dynamic id based remote user authentication scheme for multi-server environment. *Computer Standards & Interfaces*, 31(6):1118–1123.

Krishnamurthy, B., Wills, C., and Zhang, Y. (2001). On the use and performance of content distribution networks. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 169–182. ACM.

Leighton, F. T., Lewin, D. M., et al. (2007). Html delivery from edge-of-network servers in a content delivery network (cdn). US Patent 7,293,093.

Metz, C. (1999). Aaa protocols: authentication, authorization, and accounting for the internet. *IEEE Internet Computing*, 3(6):75–79.

O'Gorman, L. (2003). Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91(12):2021–2040.

Pathan, A.-M. K. and Buyya, R. (2007). A taxonomy and survey of content delivery networks. *Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report*, 4.

Pomelo, L. (2009). Analysis of netflix's security framework for'watch instantly service.

Stallings, W. (1995). *Network and internetwork security: principles and practice*, volume 1. Prentice Hall Englewood Cliffs.

Stuttard, D. and Pinto, M. (2011). *The web application hacker's handbook: Finding and exploiting security flaws*. John Wiley & Sons.

Wein, J. M., Kloninger, J. J., Nottingham, M. C., Karger, D. R., and Lisiecki, P. A. (2012). Content delivery network (cdn) content server request handling mechanism. US Patent 8,122,102.

Zambelli, A. (2009). Iis smooth streaming technical overview. *Microsoft Corporation*, 3:40.