PTC3101 - Engenho e Arte do Controle Automático

Professor: Felipe Miguel Pait

Gabriel Takeshi Medeiros Yamasaki - 9837538

Preâmbulo: A inspiração para este ensaio veio da aula da disciplina em que o Professor abordou a compressão de dados para vídeo e áudio necessários para o correto funcionamento de diversas aplicações. Apesar de apreensivo que um ensaio abordando diversos protocolos de telecomunicações e formas de compressão de dados fugisse muito do tema da disciplina, o Professor encorajou que eu seguisse a ideia uma vez que eu parecia interessado. Agradeço a compreensão e o incentivo, eles realmente significam muito.

Protocolos e funcionalidades de interesse em telecomunicações

Nos dias de hoje, as telecomunicações estão cada vez mais presentes no dia-a-dia das pessoas. Quanto mais tempo passa, mais comum se tornam telefones celulares, computadores pessoais, tablets, etc. A cada ano que passa, os processos de telecomunicações ganham mais importância na vida da população mundial, e se torna cada vez mais interessante buscar compreender seu funcionamento.

Uma forma pela qual pode-se abstrair o funcionamento destas redes que interligam as mais diversas partes do mundo é através do Modelo OSI, ou *Open Systems Interconnection* (literalmente, Interconexão de Sistemas Abertos). O Modelo OSI oferece uma forma de compreender os diversos sistemas que regem algo tão importante e complexo como a atual rede de telecomunicações mundial abstraindo seus diversos processos em *camadas* que buscam resumir as funções de cada protocolo e cada peça de forma a explicitar sua importância e encapsular sua funcionalidade, de tal forma que os demais componentes desse complexo sistema possam ser compreendidos mesmo separados, sem necessidade de inter-relacionar funcionalidades que não ocorrem na mesma camada.

O Modelo OSI, portanto, divide as aplicações de rede em 7 camadas distintas: A camada de **aplicação**, que envolve o contato da aplicação com o usuário final; A camada de **apresentação**, que envolve manipulação de dados para a passagem entre redes de comunicação e aplicação, bem como funcionalidades tais como a compressão de dados e encriptação; a camada de **sessão**, que controla o diálogo entre as máquinas, estabelecendo, terminando e em geral gerenciando este canal; a camada de **transporte**, que se preocupa na garantia de transmissão da fonte dos dados até seu destino, enquanto também assegura o cumprimento de certos serviços que garantem a qualidade da conexão, controlando coisas como erros na transmissão e segmentação de dados para manuseio das camadas superiores; a

camada de **rede**, que busca estabelecer a rota pela qual a transmissão de dados ocorrerá a fim de que os dados cheguem ao destino final; a camada de **enlace**, que tem como função garantir a qualidade da conexão de nó a nó da rede, uma vez que as rotas decididas pela camada de rede tendem a passar por vários nós antes de chegar ao destino final; e por fim a camada **física**, que nada mais é que o meio físico pelo qual a transmissão de dados ocorre, como por exemplo o fio de fibra ótica ou o ar no caso de conexões *wireless*. Estas camadas recebem números para mais rápida identificação, sendo representadas pelos números de 1 a 7, com 1 representando a camada física e 7 de aplicação, com o número crescendo conforme a proximidade com a aplicação de fato aumenta.

Para o estudo e compreensão de protocolos de comunicação, as camadas de 2 a 4, ou de enlace, rede e transporte, são de principal interesse, uma vez que abrangem os principais e mais famosos protocolos, como o Ethernet para a camada de enlace, o IP para a camada de rede e o TCP ou UDP para a de transporte.

O protocolo de Ethernet conecta nós de uma rede através de um meio físico. Era comum que vários nós estivessem conectados ao mesmo cabo, porém o advento da tecnologia de switches, que recebem os quadros,como são chamadas as unidades transmitidas através do protocolo de Ethernet, e os redirecionam aos endereços de destino, tornou isso menos comum, por razões de segurança e de aumento da eficiência das transmissões, uma vez que desta forma ocorrem menos colisões. Ainda assim, pode-se observar esta implementação de Ethernet nos dias de hoje. Neste caso, as máquinas irão descartar quaisquer quadros que cheguem a elas cujo destino seja diferente de seu próprio endereço.

Para iniciar sua transmissão, o nó verifica se está ocorrendo a transmissão de dados pelo meio físico naquele momento, uma vez que duas transmissões simultâneas causarão erros e distorções em ambas as mensagens. Caso verifique que nada está chegando em sua saída para o cabo naquele momento, ele inicia uma transmissão. Isto, é claro, não impede que dois nós na rede, mesmo nos casos em que só existem dois nós, comecem a transmitir dados aproximadamente ao mesmo tempo. Porém, caso os nós detectem que outros nós estão tentando se comunicar no mesmo momento que ele, todos os nós que estão se comunicando pelo meio físico imediatamente param o envio de dados, e recomeçam depois de um tempo randomizado. A randomização do momento de reinício é muito importante para o correto funcionamento de redes Ethernet, já que se o tempo de reinício fosse padronizado, seria provável que os nós continuariam a colidir ao tentar reiniciar suas comunicações ao mesmo momento, porém desta forma garante-se que eventualmente um deles iniciará a transmissão com tempo o suficiente para o outro nó identificar que uma transmissão está ocorrendo e esperar pelo seu término.

Quadros de Ethernet que aderem ao padrão IEEE 802.3 apresentam um formato específico que ocupa de 72 a 1530 bytes de informação. Os primeiros 7 bytes

transmitidos são o preâmbulo que realiza a sincronização dos clocks, que consiste de bits alternando entre os valores 1 e 0, formando o byte 10101010, ou 0xAA em hexadecimal, transmitido 7 vezes. Ao final do preâmbulo é transmitido um byte que visa sinalizar o início da mensagem, que é bastante similar aos bytes do preâmbulo, porém mudando o último bit de 0 para 1, formando 10101011, ou 0xAB. A partir desse momento começa a transmissão da mensagem de fato.

Os primeiros 12 bytes do quadro são reservados para os endereços MAC de aparelhos, os 6 primeiros dedicados ao endereço do destino do quadro e os outros 6 à sua origem. O que segue estes endereços varia. Obrigatoriamente, existe um campo de 2 bytes cujo uso depende de seu valor. Tradicionalmente este campo indica o tamanho da mensagem, podendo variar entre 46 e 1500 bytes. Porém, valores maiores que 1500 bytes podem ser utilizados para significar outras coisas. A maioria dos valores após o 1500 indicam o Ethertype, indicando qual protocolo que encapsula a mensagem que está sendo enviada, como o IP por exemplo. Neste caso, o tamanho da mensagem será determinado de outras maneiras. Alternativamente, caso o valor neste dois bits seja 0x8100, isso indicará que o frame está usando uma tag de acordo com o padrão IEEE 802.1Q, e que portanto seus próximos 2 bytes serão compostos de 3 bits que indicam a prioridade do frame, 1 bit que indica a possibilidade do quadro ser descartado no caso de congestionamento de rede e 12 bits que indicam sua VLAN de origem. neste caso, os dois bits que seguem compõem o segmento de tamanho ou Ethertype convencional.

Após tudo isso, segue a mensagem propriamente dita, com um tamanho que varia entre 46 e 1500 bytes, e por fim um segmento de 4 bytes que permite verificar que não houve erro na transmissão de dados. É comum também a transmissão de um símbolo que identifica o final da mensagem, ainda que isso seja mais comumente lidado pela camada física. Também é de costume deixar um espaço inter pacotes de 12 bits sem transmissão na linha antes de iniciar a transmissão de um novo quadro.

o IP, ou *Internet Protocol* (literalmente, Protocolo de Internet), é provavelmente o protocolo de rede mais conhecido. Hoje em dia, são utilizados tanto o IPv4 quanto o IPv6 em aplicações reais, com o IPv4 sendo o mais popular, apesar de esforços para a implementação do IPv6, que visa completamente substituir esse no futuro.

O IPv4 usa um endereçamento de 32 bits, permitindo 4.294.967.296 de endereços possíveis. Por causa do crescimento da população com acesso a aparelhos capazes de comunicação via internet, este número de endereços se provou inadequado ao atual cenário mundial, o que provocou o desenvolvimento do protocolo IPv6, que visa resolver este problema definitivamente com um endereçamento de 128 bits, permitindo 3.4 e38 endereços (aproximadamente 340 undecilhões de endereço possíveis). Ainda assim, por causa da sua ampla

implementação prévia, o IPv4 ainda é muito utilizado hoje em dia e é importante compreendê-lo.

A melhor maneira de explicar as funcionalidades do protocolo IPv4 é examinar seu formato, portanto se começará daqui. Os pacotes IPv4 tem 2 partes principais, o *header*, ou cabeçalho, e os dados. O IPv4, diferente do protocolo de comunicação via Ethernet explicitado anteriormente não apresenta um *checksum* ou similar que garanta que a transmissão de dados ocorreu corretamente, uma vez que a maioria dos protocolos de enlace já se encarregam deste trabalho, detectando a maioria dos erros que ocorreriam neste caso.

O cabeçalho começa com um campo de 4 bits que indica a versão do protocolo IP. Se tratando do IPv4, este campo sempre será 4, ou 0100, seguido por mais um campo de 4 bits que indica o tamanho de cabeçalho em múltiplos de 32 bits. Este campo é importante pois o último campo do cabeçalho apresenta tamanho variável, e portanto precisa-se delimitar seu tamanho. O mínimo valor do campo IHL (*Internet Header Length*, literalmente comprimento do cabeçalho de Internet) é 5, indicando um cabeçalho de 160 bits, ou 20 bytes, e seu valor máximo é 15, representando um cabeçalho de 480 bits/60 bytes

O próximo campo, chamado de DSCP, ou *Differentiated Services Code Point* (literalmente Ponto de Código de Serviços Diferenciados) e composto de 6 bits, identifica o uso de *DiffServ* pelo pacote, que caracteriza serviços de visam melhorar a qualidade do serviço prestado à aplicação, como por exemplo aplicações que necessitam de baixa latência podem sinalizar à rede esta necessidade e a rede tentará fornecer serviço priorizado a estas aplicações. Segue então um campo denominado ECN, *Explicit Congestion Notification* (Notificação de Congestão Explícita), de 2 bits, que permite o envio de notificações em caso de congestionamento de rede.

O que segue é um campo de 2 bytes, ou 16 bits, que indica o tamanho do pacote IP, cabeçalho incluso, em bytes. Seu valor mínimo é 20, que indicaria apenas um header sem adições e mensagem nula, mas pode chegar a até 65.535, tamanho máximo definido para um pacote IPv4.

Tudo isto é seguido por um novo campo de 2 bytes que fornece uma identificação única a todos os fragmentos de um mesmo datagrama, que ocorre porque no percurso da rede os pacotes estarão sujeitos ao processo de fragmentação. A fragmentação é um dos processos pelos quais os roteadores, que servem como conexões e portas de entradas às redes, podem garantir que diferentes redes comuniquem-se entre si e que mantenham sua comunicação de acordo com o padrão das demais sem que seja necessário que a máquina inicial conheça todas as redes pelas quais os pacotes passaram antes de chegar ao seu destino. Isso ocorre pois, apesar do tamanho máximo do pacote pelo padrão IP ser de 65.535

bytes, nem todas as redes são capazes de operar com o mesmo tamanho de pacotes. Assim, caso roteadores notem que para chegar ao destino é necessário passar por uma rede que não suporta o tamanho atual do pacote, o roteador retira todo o cabeçalho do protocolo IP, fragmenta os dados em pedaços menores que sejam viáveis para envio por essa rede e reaplica o cabeçalho IP nos novos pacotes fragmentados. Os próximos campos no cabeçalho também são relacionados à fragmentação, com o primeiro deles sendo um campo de 3 bits, cada qual representando uma flag que pode ser setada. O primeiro desses bits é reservado e deve ser sempre 0, o segundo indica que este pacote não deve ser fragmentado pelos roteadores, e o terceiro que existem mais fragmentos referentes a este datagrama. Se o pacote não foi fragmentado, esta flag será 0. caso contrário, será 1 para todos os fragmentos exceto o último. O próximo campo é um campo que informa o offset a ser aplicado ao fragmento encontrado neste pacote em relação ao primeiro fragmento, com seu valor representando o número de blocos de 8 bytes que deve-se deslocar os componentes do conteúdo presente no pacote.

Os próximos campos são mais simples, com um campo de 1 byte que representa o tempo de expiração do pacote, que conta regressivamente até o 0, quando o pacote expira e um alerta é enviado para a origem, implementado para que pacotes que não chegam a seu destino não persistam na rede; um campo de 1 byte que indica o protocolo utilizado no datagrama, por exemplo TCP; um campo de 2 bytes que consiste em um *checksum* para verificar a integridade da informação do cabeçalho, dois campos de 4 bytes contendo endereços IPv4, um da origem e um do destino; e um campo opcional que determina opções extras para o pacote, como por exemplo uma opção para que a rota que o pacote tomou seja traçada e informada à origem, etc. Depois do cabeçalho segue os dados enviados pela fonte, em geral provenientes da camada de transporte.

Já o IPv6 é um protocolo mais complexo e flexível do que o IPv4. similar ao IPv4, seu primeiro campo também identifica em 4 bits a versão do IP sendo utilizada, que no caso sempre será 6, ou 0110. Segue então um campo de 8 bits que identifica tanto o *DiffServ* que a aplicação deseja para seus pacotes como o ECN, como já descrito para o IPv4. O que segue é um campo de 20 bits que fornece identificação a um fluxo de informação entre destino e fonte. O próximo campo tem 2 bytes e fornece o tamanho do pacote.

O campo que segue então é um campo que determina o próximo cabeçalho. Seu valor padrão é na verdade qual o protocolo utilizado no datagrama que está sendo enviado, porém com certo valores específicos ele comunica ao roteador como deve tratar a informação que segue. Por exemplo, caso este campo indique que o pacote foi fragmentado, o roteador então interpretará os campos que seguem como o offset que deve ser dado a este fragmento em relação ao fragmento 0, se existem mais pacotes e a identificação do datagrama completo, similarmente ao processo que se

dá em IPv4, porém neste caso apenas sendo ativado quando necessário, reduzindo gasto com informações no cabeçalho que são na prática inúteis.

Os últimos campos do cabeçalho do IPv6 são um campo que limita a quantidade de envios a outros roteadores e máquinas, similar ao campo equivalente em IPv4, de 1 byte, e dois campos de 16 bytes contendo os endereços de origem e destino do pacote. Diferente do IPv4, o IPv6 não contém um *checksum* que cheque a integridade da informação do cabeçalho, confiando nas tecnologias atuais de transmissão e verificações de integridade realizadas pelos protocolos de enlace o suficiente ao ponto que considera tais funcionalidades supérfluas.

Os protocolos TCP e UDP estão entre os mais notáveis protocolos de transporte em uso hoje, e são utilizados pelas mais diversas aplicações como uma forma de abstrair a comunicação da aplicação com a rede de telecomunicações.

O protocolo TCP, ou *Transmission Control Protocol* (literalmente Protocolo de Controle de Transmissão), é bastante utilizado por sua confiabilidade. Ele se encarrega de estabelecer a comunicação com os servidores remotos, reorganizar pacotes que chegam fora de ordem pela rede, requisitar pacotes perdidos ou que chegaram com erros, entre outras funções, oferecendo uma boa interface entre a rede e a aplicação. Ele estabelece a comunicação entre máquinas através de um processo conhecido como *three-way handshake*, literalmente aperto de mão de 3 vias, no qual o cliente envia que deseja abrir a comunicação com o servidor, o servidor responde para o cliente positivamente também requisitando a conexão e por fim o cliente envia uma mensagem reconhecendo o pedido do servidor, a partir da qual o canal de comunicação fica estabelecido, caracterizando as 3 "vias" da comunicação inicial, que estabelece os parâmetros de comunicação tanto de cliente para servidor como de servidor para cliente.

No cabeçalho do protocolo TCP, seus 4 primeiros bytes são os identificadores das portas de origem e destino nas respectivas máquinas, cada um ocupando 2 bytes. Os 4 bytes seguintes são o que é chamado de um número de sequência. No caso da flag SYN, que será abordada mais adiante, estar setada com 1, este estabelece o número de sequência inicial da conexão para a máquina, e caso contrário este é o número de sequência da mensagem, que cresce conforme mais mensagens são enviadas durante a sessão. Essa funcionalidade, combinada com a função de *acknowlegde*, do inglês reconhecer, permite manter sincronicidade entre os pacotes enviados, e garantir que todos chegaram adequadamente. O próximo campo é novamente um campo de 4 bytes, que só é utilizado no caso da flag ACK estar setada, no qual se envia o último número de sequência recebido que enviado pela máquina somado de 1, de tal forma a reconhecer que aquele foi o último pacote recebido.

O próximo campo no cabeçalho estabelece o tamanho do cabeçalho em 4 bits, similar ao cabeçalho do IPv4 já discutido, este também possui um campo de tamanho variável que detalha opções que o protocolo tem de funcionalidades extras que podem ser requisitadas pelo usuário. Este campo também detalha o tamanho do cabeçalho em grupos de 32 bits, tendo um tamanho mínimo de 5, ou 20 bytes, e tamanho máximo de 15, ou 60 bytes, também muito similar ao IPv4. Os 3 bits que seguem este campo estão reservados e devem ser setados para 0, enquanto os outros 9 representam flags de diversas utilidades. As 3 primeiras se referem a funcionalidades do ECN já mencionado e de notificação de congestionamento. As demais indicam funcionalidades como indicar que os dados deste pacote são urgente e que o receptor deve processá-los assim que recebê-los, indicar que este pacote está reconhecendo o recebimento do último pacote recebido e que o campo de reconhecimento já mencionado deve ser checado para verificar sincronização entre os dois aparelhos em termos de pacotes recebidos (flag ACK mencionada anteriormente), requisitar que os dados no buffer sejam empurrados para a aplicação, requisitar o reinício da conexão, requisitar o estabelecimento da sessão pelo *three-way handshake* (flag SYN já mencionada) e a requisição do término da sessão.

Os próximos campos incluem um campo de 2 bytes que informa a quantidade de janelas de informação que aquele que enviou pode receber no momento, a fim de impedir situações em que um dos lados da conexão envie dados mais rapidamente do que o outro pode processar, o que pode levar a perda de dados, um campo de 2 bytes que inclui um checksum para os dados transmitidos, o cabeçalho TCP e um pseudo cabeçalho IP que contém informações básicas de roteamento, e por fim um campo de 2 bytes só ativo se a flag que indica processamento urgente está ativa e que aponta para o último byte de dados urgente. Após isso, vem a parte de opções do cabeçalho, de tamanho variável como no IPv4.

O protocolo UDP, ou *User Datagram Protocol* (literalmente Protocolo de Datagrama do Usuário), é bastante diferente do TCP, por várias razões. Em primeiro lugar, ele não checa para garantir que está recebendo todos os datagramas que deveria estar recebendo do outro aparelho, não requisita reenvio de dados que chegaram corrompidos, e não mantém nenhuma mudança de estado, como o TCP faz quando estabelece uma conexão, se fechando a outras, por exemplo. Em um primeiro momento poderia se perguntar o porquê deste protocolo existir e ser tão utilizado, afinal não receber todos os pacotes não parece algo que jamais seria útil, porém o UDP se mostra bastante eficaz em aplicações que necessitam se comunicar com vários *endpoints* ao mesmo tempo, já que sempre se mantém no mesmo estado, e portanto não bloqueando outros usuários de se conectarem ao mesmo serviço, e em aplicações de comunicação em tempo real, já que nestas aplicações manter-se sincronizado com o outro lado é frequentemente mais importante do que garantir o recebimento de toda a informação, como por exemplo em streaming de vídeos em sites como a Twitch ou em aplicações como o Zoom ou o Skype, em que perder um

segundo de vídeo e áudio devido a congestionamento da rede é aceitável desde que assim que a conexão for restabelecida volte-se a ter sincronicidade em tempo real o mais rápido possível, o que seria atrasado por todos os procedimentos de reconhecimento e recebimento de pacotes antigos do TCP.

Justamente por isso, o cabeçalho do UDP é extremamente simples e de extremamente fácil análise e processamento, sendo constituído de apenas 8 bytes, 2 para a porta de origem, 2 para a porta de destino, 2 para o tamanho da mensagem e 2 para um checksum que checa a integridade dos dados e do cabeçalho.

Como já foram abordados os protocolos mais conhecidos das diferentes camadas, este final será dedicado à exploração de algumas das funcionalidades mais interessantes da camada de apresentação do modelo OSI, mais especificamente a compressão de imagens e a criptografia.

A compressão de dados é uma das funcionalidades que mais auxilia no correto funcionamento das redes de telecomunicações, uma vez que pode drasticamente diminuir a quantidade de banda necessária para a transmissão da mensagem de forma efetiva. Ainda que não se abordará todos algoritmos pelo qual tal compressão é feita, tentará-se explicar os conceitos por trás desta abordagem.

Algoritmos de compressão se utilizam da redundância nos dados a fim de reduzir a quantidade de bytes necessária para a transmissão da mensagem final. Pode-se visualizar isto por exemplo pensando em uma foto. É bem provável que caso tenha-se tirado uma foto de uma paisagem que existam grandes sequências de pixels nesta foto que apresentam a mesma cor. Neste caso, ao invés de codificar para armazenamento os píxeis sequencialmente, cada um repetindo a mesma cor, poderia apenas armazenar a quantidade de pixels em sequência em que a mesma cor se repete. Além disso, caso se esteja disposto a perder a qualidade de imagem, pode-se equalizar cores suficientemente próximas como sendo a mesma, e armazená-las seguindo o princípio acima, já que o olho humano não é capaz de perceber todas as minutas variações de cor que são representáveis pela pixels na tela de um computador.

Outra técnica comum é a conversão de dados para o domínio das frequências. Seu uso é bastante intuitivo para dados de áudio, utilizando-se de janelamento de pequenas partes do áudio para gerar uma transformada de Fourier que aproximadamente resulte no espectro desejado, talvez com algumas perdas se não se utilizar todas as frequências necessárias para a representação perfeita do sinal de áudio, mas resultando em algo que representa o sinal original de maneira mais ou menos fiel, porém mesmo alguns formatos de compressão de outros tipos de dados usam Transformadas similares às de Fourier. Um exemplo é a compressão com o formato JPEG, que tem em um dos passos de seu algoritmo a transformação

de uma janela 8x8 de pixels do encoding de luminosidade ou de cor da imagem original em uma soma de frequências, na qual cada frequência representa um padrão 8x8 possível que utiliza apenas os dois extremos do encoding utilizado (puro branco e puro preto no encoding de luminosidade, por exemplo), resultando em uma matriz com índices que representam estes padrões, que se somados levando seus índices em consideração se obterá o janelamento 8x8 original.

Outra forma de se pensar em compressão de dados pode vir da natural repetição dos dados na amostra. Se se codificar os dados de tal forma que dados similares tem uma representação mais curta e dados que se mostram raros são representados por representações mais longas, pode-se chegar a um resultado em que quando compilados, se os dados com representação maior forem suficientemente incomuns e os de representação menor dominarem a amostra comprimida, resulta em um conjunto muito menor do que a amostra inicial. Esta forma de compressão também é muito utilizada em diversos formatos.

Compressão de vídeo naturalmente faz bastante uso dos métodos de compressão de imagem e áudio já explicitados, uma vez que os formatos de vídeo se utilizam dos dois, porém são capazes de realizar compressão também pela relação entre os quadros de um vídeo. Novamente, pode-se visualizar isso pensando em um vídeo normal. Se estiver filmando uma pessoa falando, é bem provável que no decorrer da fala desta pessoa vários pixels se mantenham o mesmo por grandes períodos de tempo, como por exemplo o cabelo ou partes do rosto que não realizam movimentos para a fala. Os formatos de vídeo podem então realizar um encoding relativo, em que apenas sinalizam mudanças de quadro a quadro, não informando nada sobre os pixels que se mantém os mesmos, afinal não é necessário mudar nada sobre eles. Esta técnica se mostra extremamente eficaz em comprimir a quantidade de dados necessária para armazenamento e envio de vídeos, em especial se combinada com técnicas já discutidas como por exemplo ignorar mudanças muito pequenas de coloração e considerar como sendo a mesma cor dado a incapacidade do olho humano de distinguir entre estas, ocasionando é claro em pequena perda de qualidade.

Deve-se considerar, porém, que abuso de técnicas que levam à perda da qualidade dos dados eventualmente leva à mudanças perceptíveis ao usuário final, e que o uso dessas técnicas de compressão deve ser estudado para que o usuário final não seja afetado negativamente. Ainda assim, técnicas de compressão são essenciais para o bom funcionamento das redes de telecomunicações atuais, uma vez que diminuem muito o estresse ao qual estas seriam expostas a caso tais artifícios permitem reduzir, e garantem o correto funcionamento de diversas aplicações extremamente importantes nos dias de hoje, em especial neste tempo da pandemia do novo coronavírus

Finalmente, a criptografia também é algo essencial nos dias de hoje. Em um mundo em que todas as pessoas estão conectadas pela rede, as pessoas estão mais expostas do que nunca a ataques cibernéticos que visam expor informação sensível, e dados sobre hábitos e usos são de cada vez maior interesse a grandes corporações que desejam utilizar essas informações para maior lucro sem consideração pelo indivíduo prejudicado.

Assim, a importância da proteção de informações e mensagens veiculadas na internet se mostra óbvia, e apesar dos mecanismos pelo qual se tenta manter esta segurança no mundo atual sejam vários, gostaría-se de trazer especial atenção ao algoritmo AES, ou *Advanced Encryption Standard* (literalmente Padrão de Encriptação Avançado), que é o algoritmo criptográfico usado para criptografar as mensagens trafegadas da web, tido como o padrão de segurança cibernética.

O AES é na verdade um algoritmo bem simples, com seus processos podendo facilmente ser explicados para alunos do Ensino Médio, porém sua genialidade vem do fato que é basicamente impossível reverter as transformações sem o conhecimento da chave de bits utilizado na sua codificação, criando uma criptografia praticamente inquebrável se se desconhece a chave, porém extremamente fácil de se reverter com o conhecimento desta.

O algoritmo se baseia em 4 passos repetíveis, cada um deles de extrema simplicidade. Uma vez adquirida a chave criptográfica, que varia entre 128, 196 ou 256 bits, ainda que as implementações menores estejam caindo em desuso, realiza-se uma operação xor bit-a-bit entre os dados a serem encriptados e a chave. Caso os dados sejam maiores do que a chave, os dados são fragmentados em pedaços compatíveis, e caso sejam menores, zeros são adicionados ao final para completar até que ambos fiquem do mesmo tamanho. Após realizada esta operação inicial, dividem-se os bits em 16 grupos que formam uma matriz 4x4, sobre o qual realizaram-se o resto das operações.

A primeira operação realizada com a matriz é uma substituição de valores. Cada elemento da matriz é pego e substituído por um novo elemento por um valor tabelado e distinto para cada possibilidade. O valor 0x00 seria substituído por 0x63, por exemplo. Esta tabela é aberta e pode ser consultada por qualquer um, e foi feita especificamente para criar uma transformação não-linear de difícil, que apresente pouco padrão lógico por trás e que cuja reversão só possa ser feita de forma eficaz consultando a própria tabela, não havendo função que seja capaz de realizar isso. A segunda operação envolve o deslocamento de cada linha da matriz independentemente. a primeira linha não sofre deslocamento, a segunda é deslocada uma casa para a esquerda, a terceira duas casas e a quarta 3 casas. Isso garante que as colunas da matriz não serão encriptadas individualmente, o que seria muito mais fácil de resolver do que uma encriptação envolvendo a matriz completa. O terceiro passo envolve a transformação da coluna ao realizar a

multiplicação de matrizes entre a coluna e uma matriz 4x4 fixa, resultando em uma nova coluna. Como na multiplicação de matrizes todos os elementos da coluna afetam o resultado final de cada elemento na nova coluna, este passo combinado com o anterior garante que mudanças minutas entre um grupo de bits e outro resultem em resultados finais completamente diferentes, ofuscando a relação entre os dados cifrados e os dados iniciais para aqueles que desconhecem a chave. O quarto e último passo é simplesmente realizar a operação xor bit a bit novamente com a nova matriz, com as substituições, deslocamentos e multiplicações já realizadas. Depois disso, repete-se este ciclo 9, 11 ou 13 vezes, dependendo do tamanho da chave, e no final realiza-se mais uma rodada de substituição, deslocamento e xor para finalizar a encriptação dos dados. Este processo de repetição não só faz com que pequenas mudanças nos dados ocasionem em grandes mudanças no resultado final, mas também faz com que todas as partes da chave afetem de maneira direta, por causa dos deslocamentos de linhas, ou indireta, através da multiplicação de matrizes, também ocasionando em grandes diferenças com chaves similares. Tudo isso faz com que seja difícil traçar relações entre os dados encriptados e o original ou à chave utilizada, tornando este algoritmo muito efetivo na encriptação de dados.

Infelizmente, o AES não será capaz de oferecer a mesma proteção em um mundo em que computadores quânticos sejam uma realidade, o que seria muito ruim uma vez que a gigantesca maioria das aplicações e serviços, incluindo coisas extremamente importantes como bancos, utilizam o AES como seu principal algoritmo de encriptação de dados. Felizmente, já se está trabalhando em algoritmos que se mostram eficazes em cifrar dados mesmo contra os avanços oferecidos pela computação quântica.

Este mundo interconectado em que se vive hoje em dia e tudo aquilo que o rege e o permite funcionar são tópicos fascinantes, e seus avanços são contínuos e sempre interessantes. Só resta esperar para saber aonde que isto irá na próxima vez.

Considerações finais: Obrigado pela oportunidade, professor. Havia tantos tópicos a mais que gostaria de ter abordado, como ir mais a fundo em como o IP e os roteadores de fato encontram rotas para enviar dados, como o NAT funciona e como ele foi capaz de estender a vida do IPv4, integração de IPv4 e IPv6, como funciona o WiFi, mas o texto já estava ficando demasiadamente longo como estava e no caso do WiFi envolvia conhecimentos e linguagem que não me senti preparado para tentar explicar e utilizar. Ainda assim, a experiência de pesquisa deste texto foi bem proveitosa, e aprendi e complementei conhecimentos pré-existentes muito bem. Obrigado pela disciplina