Estudo bibliográfico do RFC 4340 - Protocolo DCCP (Datagram Congestion Control Protocol)

Autores: Felipe T. Malacarne, Mateus R. Zanella

***RESUMO**: O presente artigo descreve as principais funcionalidades, história e especificações técnicas do protocolo DCCP (RFC 4340).*

***Palavras Chave:** DCCP, RFC 4340.*

INTRODUÇÃO

O *Datagram Congestion Control Protocol* (DCCP) é um protocolo de transporte que fornece conexões de um para um, bidirecionais de pacotes de dados não confiáveis de congestão controlada. O DCCP é adequado para aplicativos que transferem quantidades relativamente grandes de dados e que podem se beneficiar do controle sobre o equilíbrio entre prontidão e confiabilidade. (IETF, 2020). Mais especificamente, o protocolo oferece os seguintes pontos: fluxo de datagramas não confiáveis, estabelecimento e destruição de conexões confiáveis, controle de congestionamento incorporando *Explicit Congestion Notification* (ECN) [RFC 3168] e o *ECN Nonce* [RFC 3540], mecanismos de reconhecimento de perda de pacotes, mecanismos adicionais que informam com alta confiabilidade quais pacotes atingiram o destino, e quais pacotes estavam marcados com ECN, corrompidos ou largados no buffer de recebimento. (IETF, 2020).

O DCCP é intencionado para aplicações que podem se beneficiar sobre o controle entre as vantagens e desvantagens entre atraso e entrega confiável e ordenada, como streaming de mídia, por exemplo. Uma alternativa comum para essas aplicações é fazer o uso do protocolo UDP, porém aplicações que implementam esse protocolo devem fazer o controle de congestionamento por conta própria, a vantagem do DCCP é que ele oferece essa funcionalidade de maneira integrada, além de implementar ECN. (IETF, 2020).

Hoje, o protocolo DCCP não é amplamente utilizado, porém ainda possui compatibilidade com o sistema operacional Linux e em algumas redes experimentais ou de pesquisa. (Linux Kernel Docs, 2016). Algumas distribuições desse sistema operacional possuem um utilitário chamado "*dccp*", que fornece uma funcionalidade similar ao comando "*cp*" (copy), no *filesystem* dCache, sendo possível copiar um arquivo fonte para um arquivo de destino. (Die.net, 2020).

HISTÓRICO

O DCCP foi desenvolvido como um protocolo de controle de congestionamento independente e não confiável desde cerca de 2001, possuindo uma dependência subjacente no protocolo IP (RFC 791). Ele foi totalmente padronizado como RFC de rastreamento de padrões em 2006. (Wireshark, 2020)

O começo do desenvolvimento do protocolo DCCP se deu ao fato de que a maioria dos protocolos de internet utilizados atualmente (TCP [RFC 793], UDP [RFC 768], SCTP [RFC 4960]) utilizam números de portas "publicados", ou seja, a porta do servidor deve ser conhecida para que o cliente possa estabelecer uma conexão. (RFC Editor, 2009).

No rascunho mais antigo do DCCP, os autores buscavam solucionar o problema das portas publicadas de um modo à prova de futuro, pois esse método enfrenta alguns problemas, como dito no guia de referência "The Datagram Congestion Control Protocol (DCCP) Service Codes":

> - O espaço de portas não é suficientemente grande para permitir uma alocação fácil e regulamentada, levando muitas aplicações a operarem em portas não registradas, potencialmente gerando conflitos com outras aplicações.
> - O uso de firewalls baseados em portas incentiva os desenvolvedores de aplicativos a disfarçarem uma aplicação como outra na tentativa de contornar as regras de filtragem do firewall, resultando em uma maior utilização da inspeção profunda de pacotes para identificar o serviço associado a um número de porta.
> - Provedores de serviços de Internet frequentemente implementam proxies transparentes, principalmente para melhorar o desempenho e reduzir custos. Por exemplo, solicitações TCP destinadas à porta TCP 80 são frequentemente redirecionadas para um proxy web.

De curto modo, a solução encontrada foi utilizar um Código de Serviço de 32 bits, que é incluído apenas no pacote da requisição DCCP. O motivo do uso desse valor é que se torna extremamente simples criar um valor único para cada aplicação, e adicionar esse código não adiciona nenhum *overhead* para o fluxo de dados real. Desse modo, esse valor de 32 bits é utilizado para identificar as conexões, fazendo com que o servidor possa ouvir apenas no Código de Serviço específico. O cliente DCCP então envia o Código na requisição, atingindo o servidor de destino apenas. (RFC Editor, 2009).

Essa abordagem possui uma desvantagem: não é possível hospedar dois servidores para o mesmo serviço em uma mesma máquina, tarefa que é trivial com portas. Além disso, o

protocolo era suficientemente diferente para levantar indagações sobre se a sua implementação dificultaria o seu uso através de NAT 's ou *middlewares*.

O RFC 4340 abandonou o uso do Código de Serviço de 32 bits, e optou por utilizar dois valores de portas de 16 bits, um escolhido pelo servidor e outro pelo cliente. Isso soluciona o problema de implementação com *middleboxes*, porém cria um novo desafio: *"Como a porta do servidor se relaciona com o Código de Serviço?"*. A ideia dessa mudança é que as duas portas de 16 bits foram juntas um identificador de conexão único de 32 bits, que é único entre um par de sistemas. (RFC Editor, 2009).

O grande número de Códigos de Serviço únicos disponíveis permite que todas as aplicações possuam um identificador único, porém, ainda existe um problema: *"Se o Código de Serviço é escolhido pelo servidor, como o cliente pode saber a porta para estabelecer a conexão?"*. (RFC Editor, 2009). A solução encontrada foi registrar as portas DCCP, o que entra em contradição com a sua proposta inicial, caindo no problema de que, conforme os números de porta se tornam escassos, há a motivação da necessidade de associar mais de um Código de Serviço a uma porta de escuta (por exemplo, duas aplicações diferentes poderiam ser atribuídas à mesma porta de servidor e precisar de ser executadas no mesmo host ao mesmo tempo, diferenciadas pelos seus diferentes Códigos de Serviço associados).

Desse modo, os Códigos de Serviço providenciam flexibilidade para que os clientes possam identificar o servidor para estabelecer uma comunicação, permitindo um número maior de conexões simultâneas para um serviço específico do que é possível quando o serviço é identificado por um único número de porta Publicado.

ESPECIFICAÇÕES

Nesta seção serão discutidas as especificações do protocolo DCCP.

Formato dos pacotes

O cabeçalho do DCCP pode variar de 12 a 1020 bytes de comprimento e pode ser dividido em três partes. A parte inicial dele tem a mesma semântica para todos os tipos de pacotes. A seguir, temos uma seção onde se encontram quaisquer campos de comprimento fixo adicionais exigidos pelo tipo de pacote, e então uma lista de opções de comprimento variável. [RFC 4340, seção 5.0]

Cabeçalho Genérico
Campos Adicionais (dependendo do tipo)
Opções (opcional)
Área de Dados da Aplicação

O cabeçalho genérico do DCCP pode assumir diferentes formas a depender de X, o *bit de sequência estendido,* quando X é 1, o cabeçalho genérico ocupa 16 bytes, e o número de sequência possui 48 bits.

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Porta Origem								Porta Destino							
	Desloc. Dados				CCVal		CsCov		Checksum							
	Reservado			Tipo				X	Número de Sequência (alta)							
	Número de Sequência (baixa)															

Caso o X for 0, apenas os 24 bits de baixo do Número de Sequência são transmitidos, e o header genérico ainda possui 12 bytes.

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Porta Origem								Porta Destino							
	Desloc. Dados				CCVal		CsCov		Checksum							
	Reservado			Tipo				X	Número de Sequência (alta)							

Sistema de Controle de Congestão

As conexões DCCP possuem controle de congestionamento, porém, ao contrário das conexões TCP e UDP, as conexões DCCP possuem uma escolha no mecanismo de controle. As duas conexões podem possuir mecanismos diferentes, denominados *one-byte congestion control identifiers* ou CCIDs. Os CCIDs 0,1 e 4-255 são reservados, os 2 e 3 são definidos. (IETF, 2020).

O CCID 2 possui um sistema de congestionamento similar ao TCP, onde o remetente mantém uma janela de congestionamento e envia pacotes até aquela janela estiver cheia. Os reconhecimentos contém a sequência de todos os pacotes recebidos na mesma janela de tempo, similar ao *reconhecimento seletivo* (SACK) [RFC2018]. (IETF, 2020).

O CCID 3 providencia o TCP-Friendly Rate Control (TFRC), uma forma de controle de congestionamento baseada em equações, pensada para responder a congestionamentos de maneira mais suave quando comparada ao CCID 2. O remetente mantém um ritmo de transmissão, que é atualizado utilizando uma estimativa de perda de pacotes do destinatário. (IETF, 2020).

REFERÊNCIAS

Linux Kernel Documentation: DCCP. Disponível em: https://docs.kernel.org/networking/dccp.html. Acesso em: 18 abr. 2024.

RFC 4340 (IETF): Handley, M., Floyd, S., Whetten, B., Kermode, R. Datagram Congestion Control Protocol (DCCP). RFC 4340, IETF, março de 2006. Disponível em: https://datatracker.ietf.org/doc/rfc4340/. Acesso em: 18 abr. 2024.

Datagram Congestion Control Protocol: Internet Engineering Task Force (IETF). Sobre o Grupo de Trabalho Datagram Congestion Control Protocol (DCCP). Disponível em: https://datatracker.ietf.org/wg/dccp/about/. Acesso em: 18 abr. 2024.

Datagram Congestion Control Protocol (DCCP): Handley, M., Floyd, S., Whetten, B., Kermode, R. Datagram Congestion Control Protocol (DCCP). RFC 4340, IETF, março de 2006. Disponível em: https://datatracker.ietf.org/doc/html/rfc4340. Acesso em: 18 abr. 2024.

Página do manual do DCCP no Linux: Linux. Manual do DCCP (Protocolo de Controle de Congestionamento de Datagrama). Disponível em: https://linux.die.net/man/1/dccp. Acesso em: 18 abr. 2024.

Datagram Congestion Control Protocol (DCCP). Wireshark. Disponível em: https://wiki.wireshark.org/DCCP. Acesso em 23 abr. 2024.

The Datagram Congestion Control Protocol (DCCP) Service Codes. RFC-Editor. Disponível em: https://www.rfc-editor.org/rfc/rfc5595#section-1.1. Acesso em 23 abr. 2024.