

Sistemas Multimídia

Aula 3: Protocolos de Rede, IPv6 e Organização de Projetos

Professora Ana Luiza Scharf

IFSC - São José
Departamento de Telecomunicações

Semestre 2026.1

Agenda da Aula

- 1 Revisão de conceitos: IPv4 e IPv6
- 2 Arquiteturas de comunicação multimídia
- 3 Protocolos de sinalização (SIP e alternativas)

Evolução dos Protocolos IP

IPv4 (1981)

- 32 bits (4.3 bi endereços)
- Classes A, B, C, D, E
- **Problema:** Escassez de endereços
- **Solução:** NAT, CIDR

IPv6 (1998)

- 128 bits (3.4×10^{38} endereços)
- Estrutura hierárquica
- **Vantagem:** Endereços abundantes
- **Recurso:** Autoconfiguração

Comparação Detalhada

Característica	IPv4	IPv6
Tamanho do endereço	32 bits	128 bits
Notação	Decimal pontuada	Hexadecimal com dois-pontos
Exemplo	192.168.1.1	2001:db8::1
Máscara de rede	Necessária	Incorporada
Configuração	Manual ou DHCP	Autoconfiguração (SLAAC)
Broadcast	Sim	Não (usa multicast)
NAT	Necessário	Opcional
Segurança	IPSec opcional	IPSec integrado

Endereçamento IPv4 - Classes

Classes Tradicionais

- **Classe A:** 1.0.0.0 - 126.255.255.255
- **Classe B:** 128.0.0.0 - 191.255.255.255
- **Classe C:** 192.0.0.0 - 223.255.255.255
- **Classe D:** Multicast
- **Classe E:** Reservado

Redes Privadas (RFC 1918)

- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16

Característica: Não roteáveis na internet

Formato do Endereço IPv6

Estrutura

8 grupos de 4 hexadecimais separados por :

2001:0db8:85a3:0000:0000:8a2e:0370:7334

Regras de Abreviação

- 1 Zeros à esquerda podem ser omitidos
- 2 Uma sequência de zeros pode ser substituída por ::
- 3 :: só pode aparecer uma vez

Exemplos de Abreviação IPv6

Endereço Completo	Endereço Abreviado
2001:0db8:0000:0000:0000:0000:1428:57ab	2001:db8::1428:57ab
fe80:0000:0000:0000:0202:b3ff:fe1e:8329	fe80::202:b3ff:fe1e:8329
2001:0000:0000:0000:0000:0000:0000:0001	2001::1
0000:0000:0000:0000:0000:0000:0000:0001	::1

Tipos de Endereços IPv6

Unicast

- **Global:** 2000::/3
- **Link-local:** fe80::/64
- **Unique Local:** fc00::/7
- **Loopback:** ::1/128

Multicast

- **Solicited-node:** ff02::1:ff00:0/104
- **All-nodes:** ff02::1
- **All-routers:** ff02::2

Anycast

- Mesmo prefixo em múltiplos locais
- Pacote vai para o mais próximo

Endereços Especiais IPv6

Endereço	Descrição
::1/128	Loopback (equivalente ao 127.0.0.1)
::/128	Endereço não especificado
::ffff:0:0/96	Mapeamento IPv4
2001:db8::/32	Prefixo para documentação
fc00::/7	Unique Local Addresses (ULA)
fe80::/64	Link-local (não roteável)
ff00::/8	Endereços multicast
2000::/3	Global unicast (roteável)

Autoconfiguração IPv6 (SLAAC)

Stateless Address Autoconfiguration

- 1 Dispositivo gera endereço link-local (fe80::/64)
- 2 Envia solicitação de roteador (Router Solicitation)
- 3 Roteador responde com prefixo da rede
- 4 Dispositivo monta endereço global
- 5 Verifica unicidade (Duplicate Address Detection)

Vantagem

Não requer servidor DHCP. Roteadores anunciam prefixos automaticamente.

SLAAC vs DHCPv6

SLAAC

- Stateless (sem estado)
- Roteador anuncia prefixo
- Dispositivo gera seu endereço
- Não fornece outras informações

DHCPv6

- Stateful (com estado)
- Servidor atribui endereços
- Pode fornecer DNS, NTP, etc.
- Mais controle centralizado

Comandos Práticos IPv6

Linux

- `ip -6 addr show`
- `ping6 endereco-ipv6`
- `traceroute6 endereco-ipv6`
- `ss -6 -tln`

Windows

- `ipconfig /all`
- `ping -6 endereco-ipv6`
- `tracert -6 endereco-ipv6`
- `netstat -an | findstr LISTENING`

Desafios da Comunicação em Tempo Real

- **Latência:** Tempo entre envio e recebimento
- **Jitter:** Variação na latência
- **Pacotes perdidos:** Afeta qualidade da mídia
- **NAT e Firewalls:** Barreiras à conectividade
- **Escalabilidade:** Suporte a múltiplos usuários

Arquiteturas Básicas

1. Ponto a Ponto

- Comunicação direta
- Sem servidor intermediário
- Simples mas limitada
- Problemas com NAT

2. Cliente-Servidor

- Servidor central
- Todos conectam ao servidor
- Fácil gerenciamento
- Ponto único de falha

Arquitetura Ponto a Ponto (Peer-to-Peer)

[Diagrama ilustrativo de comunicação P2P]

- **Vantagem:** Baixa latência, descentralizado
- **Desvantagem:** Problemas com NAT e firewall
- **Uso:** WebRTC para comunicação direta

Arquitetura Cliente-Servidor

[Diagrama ilustrativo de arquitetura Cliente-Servidor]

- **Vantagem:** Controle central, fácil deploy
- **Desvantagem:** Latência maior, escalabilidade limitada
- **Uso:** Jogos MMO tradicionais

Arquitetura Híbrida

[Diagrama ilustrativo de arquitetura Híbrida]

- **Sinalização:** Via servidor central
- **Mídia:** Direta entre clientes quando possível
- **Fallback:** Via servidor quando NAT bloqueia

Componentes de um Sistema Multimídia

Sinalização

- Estabelece/termina sessões
- Negocia codecs e parâmetros
- Troca endereços de network
- Exemplos: SIP, WebSocket

Transporte de Mídia

- Transmite áudio/vídeo/dados
- Controla qualidade (QoS)
- Lida com perdas e atrasos
- Exemplos: RTP, WebRTC

SIP - Session Initiation Protocol

Características

- Protocolo de sinalização para sessões multimídia
- Inspirado em HTTP (textual, métodos, respostas)
- Extensível através de headers
- Usa UDP, TCP ou SCTP

Métodos Principais

INVITE, ACK, BYE, CANCEL, OPTIONS, REGISTER

Mensagem SIP - Exemplo

INVITE

```
INVITE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.example.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.example.com>
From: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.example.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.example.com>
Content-Type: application/sdp

Content-Length: 142
```

SDP - Session Description Protocol

Função

Descrever parâmetros da sessão multimídia

Exemplo

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.atlanta.example.com
s=Conversa SIP
c=IN IP4 192.0.2.4
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 51372 RTP/AVP 31
a=rtpmap:31 H261/90000
```

WebSocket - Alternativa Moderna

Vantagens para Jogos

- Conexão bidirecional persistente
- Baixa sobrecarga (headers pequenos)
- Suporte nativo em navegadores
- Fácil implementação em JavaScript

Diagrama WebSocket

[Ilustração WebSocket]

WebRTC - Web Real-Time Communication

Componentes

- **getUserMedia**: Captura de áudio/vídeo
- **RTCPeerConnection**: Conexão P2P
- **RTCDataChannel**: Canal de dados
- **Ice, Stun, Turn**: NAT traversal

Uso em Jogos

Ideal para jogos que precisam de comunicação direta com baixa latência.

Tecnologías Recomendadas

Front-end

- HTML5 + CSS3
- JavaScript/TypeScript
- Phaser.js (framework)
- Canvas/WebGL

Back-end

- Node.js + Express
- Socket.IO
- WebSocket
- Redis (opcional para cache)

Phaser.js - Framework de Jogos

Vantagens

- Especializado para jogos HTML5
- Física integrada (Arcade, Matter)
- Animações e sprites fáceis
- Comunidade ativa, muitos exemplos

Código Básico

```
const config = {  
  type: Phaser.AUTO,  
  width: 800,  
  height: 600,  
  scene: { preload, create, update }  
};  
  
const game = new Phaser.Game(config);
```

Socket.IO - Comunicação em Tempo Real

Funcionalidades

- Conexão bidirecional
- Rooms (salas) para agrupar clientes
- Broadcast para múltiplos clientes
- Reconexão automática

Servidor Básico

```
const io = require('socket.io')(3000);
io.on('connection', (socket) => {
  socket.on('move', (data) => {
    socket.broadcast.emit('playerMoved', data);
  });
});
```

Dúvidas?

Vamos começar o desenvolvimento!