UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
COLEGIADO DE CIÊNCIA DA COMPUTAÇÃO

Trabalho Prático - Cliente e servidor NTP

Título: Trabalho Prático - Cliente e servidor NTP
Aluno: Pedro Henrique de Oliveira Berti
Data: 13/03/2025

---

1. INTRODUÇÃO

A sincronização do tempo é essencial para o funcionamento de redes de computadores, garantindo a consistência temporal entre dispositivos e permitindo a correta execução de protocolos e serviços distribuídos. O Network Time Protocol (NTP) é um dos protocolos mais utilizados para esse fim, permitindo a sincronização precisa de relógios por meio da comunicação com servidores confiáveis. Segundo Kurose e Ross (2021), "a sincronização de tempo é crítica para diversas aplicações, incluindo segurança de redes, sistemas distribuídos e registros de eventos".

O NTP opera utilizando um modelo hierárquico de servidores e clientes, onde os dispositivos ajustam seus relógios com base em timestamps trocados via UDP na porta 123. Neste trabalho, foi desenvolvida uma implementação manual de um cliente e um servidor NTP, conforme especificação do NTPv4 (RFC 5905). O sistema foi projetado para enviar e receber pacotes NTP formatados corretamente, calcular offset e round-trip delay (RTT) e, opcionalmente, utilizar autenticação via HMAC-SHA256 para garantir a integridade das mensagens.

A implementação foi testada tanto com servidores oficiais para verificar sua compatibilidade e precisão quanto localmente, permitindo a validação do mecanismo de autenticação via HMAC.

2. OBJETIVOS

O objetivo deste trabalho é implementar um cliente e um servidor NTP (Network Time Protocol), seguindo a especificação do NTPv4 (RFC 5905), para possibilitar a sincronização de tempo entre dispositivos em uma rede. Para isso, são definidos os seguintes objetivos específicos:

- Desenvolver um cliente NTP capaz de enviar pacotes formatados corretamente para servidores oficiais e receber as respostas para calcular offset e round-trip delay (RTT).
- Implementar um servidor NTP capaz de processar requisições de clientes e fornecer

respostas compatíveis com o protocolo NTP.

- Construir manualmente os pacotes NTP, incluindo os campos obrigatórios, sem o uso de bibliotecas específicas para NTP.
- Implementar um mecanismo opcional de autenticação via HMAC-SHA256, garantindo a integridade e autenticidade das mensagens trocadas entre cliente e servidor.
- Realizar testes de comunicação com servidores NTP oficiais, verificando a precisão da sincronização.
- Realizar testes em um ambiente local, validando o funcionamento da autenticação e garantindo que apenas clientes autorizados possam sincronizar o tempo.

3. MATERIAL UTILIZADO

Para a realização desta prática, utilizei o computador desktop da minha casa. O sistema operacional utilizado foi o Windows 10 Home, versão 22H2. A conexão de rede foi cabeada, fornecida pela Certto Fibra, com uma velocidade de 100 Mbps. As configurações do meu computador são:

- Processador: Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz
- Memória RAM: 8 GB
- SSD: Kingston 220 GB
- HDD: Samsung 1 TB
- Placa de vídeo: NVIDIA GeForce GTX 660

4. METODOLOGIA

Para o desenvolvimento deste trabalho, foram seguidas diversas etapas para garantir a correta implementação do cliente e servidor NTP, respeitando a especificação do NTPv4 (RFC 5905). A seguir, são descritos os procedimentos adotados:

4.1. **Estudo do Protocolo NTP**

Inicialmente, foi realizado um estudo sobre o Network Time Protocol (NTP), com base na RFC 5905, para compreender seu funcionamento, estrutura de pacotes e cálculos necessários para sincronização do tempo.

4.2. **Implementação do Cliente NTP**

O cliente foi implementado em Python, utilizando sockets UDP para enviar pacotes NTP a servidores de tempo. O pacote NTP foi construído manualmente, incluindo os seguintes campos:

- Leap Indicator (LI), Version Number (VN) e Mode
- Timestamps de referência, origem, recebimento e transmissão
- Cálculo do offset e round-trip delay (RTT) para ajuste do tempo

O cliente foi projetado para se comunicar com **servidores oficiais NTP** e exibir a hora sincronizada. Além disso, foi adicionada uma funcionalidade opcional para comunicação com um servidor local utilizando **autenticação HMAC-SHA256**.

4.3. **Implementação do Servidor NTP**

O servidor NTP também foi desenvolvido em Python, utilizando sockets UDP para escutar requisições de clientes e responder com pacotes corretamente formatados. O servidor recebe a requisição, extrai o timestamp enviado pelo cliente e responde com os valores necessários para a sincronização.

5. **RESULTADOS E DISCUSSÃO**

5.1. **Funcionamento da Implementação**

Após a implementação do cliente e servidor NTP, foram realizados testes para avaliar sua funcionalidade e precisão. O cliente foi capaz de enviar pacotes corretamente formatados e interpretar as respostas dos servidores, enquanto o servidor conseguiu responder às requisições de forma compatível com o protocolo NTPv4 (RFC 5905).

Além da comunicação básica, a implementação também incluiu a opção de autenticação via HMAC-SHA256, garantindo que apenas clientes autorizados pudessem sincronizar o tempo com o servidor local. Isso aumentou a segurança da troca de mensagens, prevenindo manipulações indevidas nos dados transmitidos.

5.2. **Testes com Servidores Oficiais**

O cliente foi testado com servidores NTP oficiais, como a.st1.ntp.br. Nos testes, a comunicação ocorreu sem falhas, e a hora sincronizada foi exibida corretamente. Os cálculos de offset e round-trip delay (RTT) foram realizados com sucesso, demonstrando que a implementação

conseguiu obter o tempo com precisão.

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python cliente.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 1
Informe o endereço do servidor oficial NTP (ou pressione Enter para usar o padrão): a.st1.ntp.br
Servidor NTP: a.st1.ntp.br
Hora sincronizada: 2025-03-13 01:17:44
Round-Trip Delay (RTT): 0.156461 segundos
Offset calculado: 1.209475 segundos
```

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python cliente.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 1
Informe o endereço do servidor oficial NTP (ou pressione Enter para usar o padrão): pool.ntp.br
Servidor NTP: pool.ntp.br
Hora sincronizada: 2025-03-13 01:19:05
Round-Trip Delay (RTT): 0.033010 segundos
Offset calculado: 0.480381 segundos
```

```
Informe o endereço do servidor oficial NTP (ou pressione Enter para usar o padrão): time.windows.com
Servidor NTP: time.windows.com
Hora sincronizada: 2025-03-13 01:20:01
Round-Trip Delay (RTT): 0.164470 segundos
Offset calculado: 0.261709 segundos
```

5.3. **Testes com Servidor Local e Autenticação**

Para validar a funcionalidade da autenticação via HMAC-SHA256, foi realizada uma simulação onde o cliente se conectava ao servidor local configurado para exigir autenticação.

5.3.1. **Caso de Cliente Autorizado**

Quando o cliente utilizava a chave secreta correta, a verificação HMAC era bem-sucedida e a sincronização ocorria normalmente.

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python servidor.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 0
Servidor NTP rodando na porta 123 - Modo: local (com autenticação)
Resposta enviada para ('127.0.0.1', 58401)
```

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python cliente.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 0
Informe o endereço do servidor local NTP: 127.0.0.1
Servidor NTP: 127.0.0.1
Hora sincronizada: 2025-03-13 01:22:19
Round-Trip Delay (RTT): 0.001012 segundos
Offset calculado: -0.664387 segundos
```

5.3.2. **Caso de Cliente Não Autorizado**

Ao tentar se conectar com um cliente sem a chave correta, o servidor rejeitou a requisição e não enviou uma resposta válida. Isso demonstrou que a autenticação foi eficaz para impedir acessos não autorizados.

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python servidor.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 0
Servidor NTP rodando na porta 123 - Modo: local (com autenticação)
Falha na verificação HMAC de ('127.0.0.1', 61232). Ignorando requisição.
```

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python cliente.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 0
Informe o endereço do servidor local NTP: 127.0.0.1
Erro: Erro ao obter hora NTP: timed out
```

5.4. **Teste do Cliente NTP no Windows**

Para validar a compatibilidade do servidor local com um cliente NTP oficial, utilizou-se o serviço Windows Time (w32time) para sincronizar a hora com o servidor implementado.

```
PS C:\Users\fangy\Desktop\Trabalho Redes> python servidor.py
Digite 1 para servidor oficial ou 0 para servidor local (com autenticação): 1
Servidor NTP rodando na porta 123 - Modo: oficial (sem autenticação)
Resposta enviada para ('127.0.0.1', 56001)
Resposta enviada para ('127.0.0.1', 56002)
Resposta enviada para ('127.0.0.1', 62565)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Experimente a nova plataforma cruzada PowerShell https://aka.ms/pscore6

PS C:\Users\fangy> w32tm /stripchart /computer:127.0.0.1 /dataonly
Acompanhando 127.0.0.1 [127.0.0.1:123].
A hora atual é 13/03/2025 01:30:43.
01:30:43, -00.3586670s
01:30:45, -00.3601732s
01:30:47, -00.3625700s
```

6. **CONCLUSÕES**

A implementação do cliente e servidor NTP permitiu compreender na prática o funcionamento do Network Time Protocol (NTP) e a importância da sincronização precisa do

tempo em redes de computadores.

Os testes realizados demonstraram que o cliente NTP foi capaz de se comunicar corretamente com servidores oficiais, obtendo e exibindo o horário sincronizado com precisão.
Além disso, o servidor NTP local respondeu às requisições de clientes implementados manualmente, permitindo validar sua compatibilidade com o protocolo.

Adicionalmente, foi possível testar a implementação utilizando o cliente NTP nativo do Windows (w32tm), demonstrando que o servidor desenvolvido é compatível com o sistema operacional e pode ser utilizado para sincronização de tempo.

BIBLIOGRAFIA

KUROSE, ROSS. **Redes De Computadores E A Internet Uma Abordagem Top Down, Editora: Pearson,** 2013.

TANENBAUM. **Redes de Computadores, Editora: Pearson,** 2011.

COMER. **Redes de Computadores e Internet, Editora: Bookman,** 2016.

rfc5905. Disponível em: <https://datatracker.ietf.org/doc/html/rfc5905>.