Análise das versões do TLS para o Node.js (VERSÃO PÚBLICA)

Sumário

1.TLS

TLS é o acrônimo de *Transport Layer Security* (Segurança em Camada de Transporte), um protocolo publicado em 1999 na RFC 2246. O objetivo principal do protocolo TLS é fornecer privacidade e integridade de dados entre dois aplicativos em comunicação. O protocolo é composto por duas camadas: o protocolo de registro TLS e o protocolo de handshake TLS. No nível mais baixo, sobreposto a algum protocolo de transporte confiável (por exemplo, TCP[TCP]), está o protocolo de registro TLS. A Figura 1 mostra como o TLS é implementado na camada de Sessão do modelo Open systems interconnection (OSI)[1].

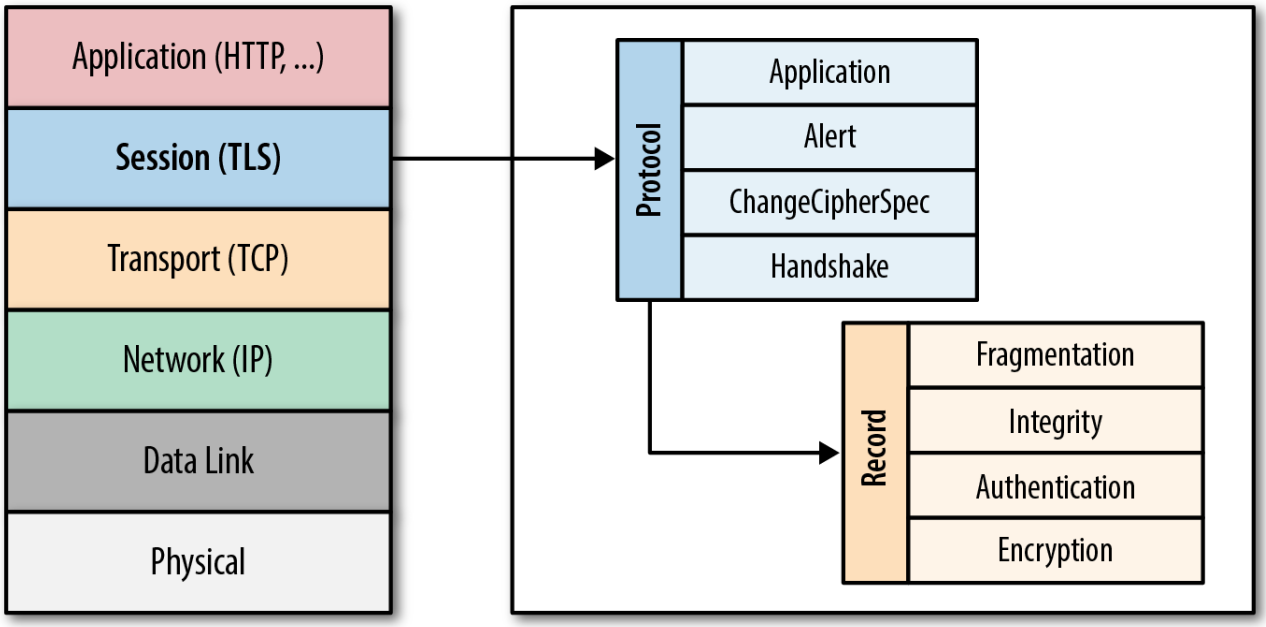

Figura 1: Arquitetura do protocolo Transport Layer Security (TLS)

1.1. Versões do TLS

O protocolo tem mais de uma versão, e assim, mais de uma implementação disponível em diversas linguagens de programação. O uso da versão mais recente é recomendável porque ela implementa o estado da arte da segurança em comunicação.

Tabela 1: Versões do TLS

Versão	RFC	Definida em
1.0	2246	1999
1.1	4346	2006
1.2	5246	2008
1.3	8446	2018

1 A especificação OSI está disponível em: https://www.iso.org/ics/35.100/x/

Há uma implementação de TLS em código livre e aberto, o OpenSSL. No lugar de implementar diretamente o TLS, as linguagens de programação podem usar o OpenSSL. A versão estável mais recente do OpenSSL é a série 3.2 com suporte até 23/11/2025. Também está disponível a série 3.1 com suporte até 14/03/2025, e a série 3.0 que é uma versão de suporte de longo prazo (LTS) com suporte até 07/09/2026. Todas as versões mais antigas versões (incluindo 1.1.1, 1.1.0, 1.0.2, 1.0.0 e 0.9.8) agora estão sem suporte e não devem ser usadas. Os usuários dessas versões mais antigas são incentivados a atualizar para 3.2 ou 3.0 o mais rápido possível. O sítio do OpenSSL contém a lista de vulnerabilidades da implementação e quais são as versões que as corrigem. A Tabela 2 mostra em quais versões de OpenSSL foi adicionado suporte às versões de TLS.

Tabela 2: Versões do OpenSSL e suporte a TLS

OpenSSL	Lançamento	Adicionou suporte a TLS
0.9.0	Não houve	1.0
0.9.1	Não houve	1.1
1.0.1	14/03/2012	1.2
3.0	07/09/2021	1.3

SSL é o acrônimo de *Secure Sockets Layer*, um protocolo publicado na versão 2.0 por uma equipe da Netscape liderada pelo criptógrafo egípcio Taher Elgamal, em 1995. O protocolo TLS foi lançado em 1999 como sucessor do SSL 3.0. O TLS é o sucessor direto do SSL, e todas as versões do SSL agora estão obsoletas. No entanto, é comum encontrar o termo *SSL* descrevendo uma conexão TLS. Na maioria dos casos, os termos *SSL* e *SSL/TLS* referem-se ao protocolo TLS e aos certificados TLS. Assim, o OpenSSL deveria se chamar OpenTLS, pois já nasceu após a obsolescência do SSL.

1.2. Versões do Node.js

A versão mais recente do Node.js é a 22.5.1, mas a versão recomendada é a 20.16.0 (LTS)[2], com suporte ativo até janeiro de 2024 e suporte de manutenção até julho de 2026. Essa é a versão recomendada por ter um período de suporte longo. A recomendação geral para o Node.js é usar as versões LTS que tem suporte longo (*Long Term Support*), que são as versões com o número *major* par. As versões ímpares são recomendadas apenas para prospecção de novas funcionalidades a serem incorporadas nas futuras versões de suporte longo. É vital que a versão de Node.js em produção esteja pelo menos na fase de suporte *Maintenance*, em que ainda há atualizações para corrigir vulnerabilidades. Recomenda-se, entretanto, a atualização para a versão com suporte longo para evitar possíveis explorações de vulnerabilidades pela demora na migração de versões.

O gráfico da Figura 2 apresenta o planejamento de suporte das versões de Node.js até 2026. A próxima versão de suporte longo será lançada após julho de 2025. Sobre a versão 20.16.0, há uma observação importante sobre TLS: O OpenSSL 3 descontinuou o suporte para mecanismos personalizados com uma recomendação para mudar para seu novo modelo de provedor. A opção *clientCertEngine* para *https.request()*, *tls.createSecureContext()* e *tls.createServer()*; o *privateKeyEngine* e *privateKeyIdentifier* para *tls.createSecureContext()*; e *crypto.setEngine()* dependem dessa funcionalidade do OpenSSL, assim, essas opções estão obsoletas.

2 https://nodejs.org/pt/download/package-manager

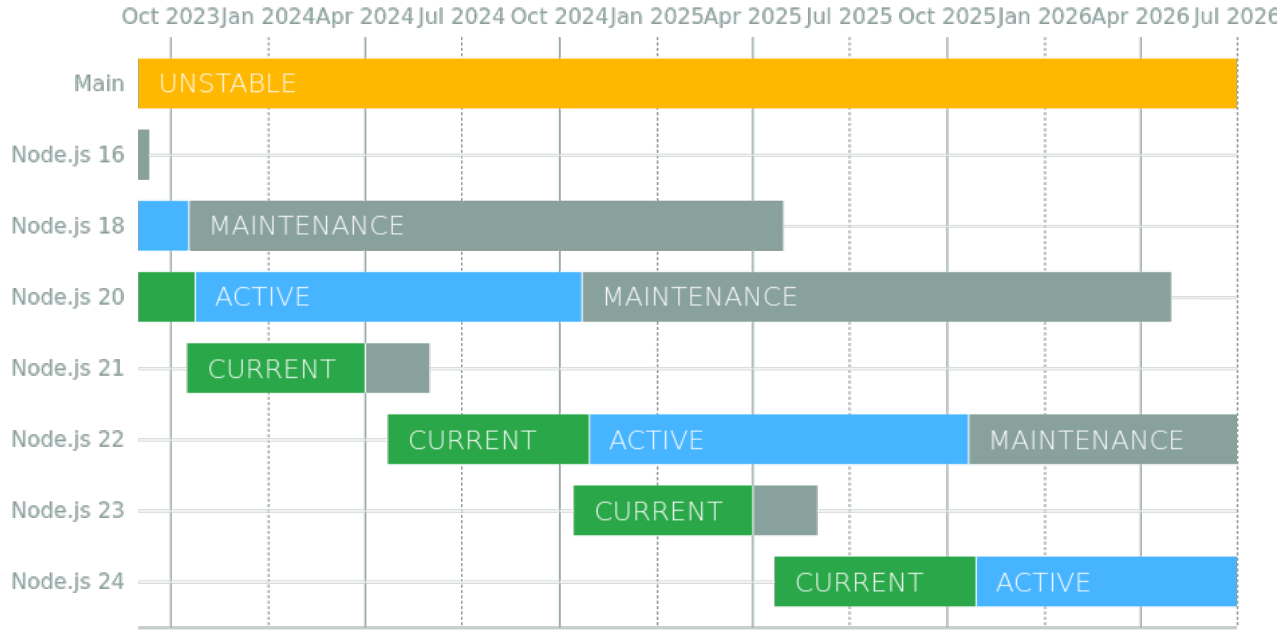

Figura 2: Agenda de suporte para as versões de Node.js

O Node.js passou a suportar TLS 1.2 desde a versão 0.10 (+OpenSSL 1.0.1) e TLS 1.3 desde a versão 12 (+OpenSSL 1.1.1). As versões suportadas de TLS retornadas pelo método *tlsSocket.getProtocol()*[3] são:

- 'SSLv3'
- 'TLSv1'
- 'TLSv1.1'
- 'TLSv1.2'
- 'TLSv1.3'

O método *tlsSocket.getSession()* funciona somente para TLSv1.2 e abaixo. Para TLSv1.3, as aplicações Node.js devem usar o evento *'session'* (que também funciona para TLSv1.2 e abaixo).

A documentação de TLS do Node.JS está disponível em: https://nodejs.org/api/tls.html.

Para garantir que os aplicativos Node.JS permaneçam seguros, a versão TLS não deve ser codificada permanentemente. Os aplicativos Node.JS devem usar a versão TLS compatível com o sistema operacional (SO).

A recomendação para a construção de aplicações Node.js é não especificar a versão do TLS, mas configurar o código para permitir que o sistema operacional decida a versão do TLS. É recomendável executar uma auditoria de código completa para verificar se você não está especificando uma versão TLS ou SSL em sua aplicação Node.js.

3 https://nodejs.org/api/tls.html#tlssocketgetprotocol

2.Análise

Conforme relatamos no capítulo anterior, todas as versões de Node.js a partir da 12 suportam TLS 1.3. A recomendação é que todas as aplicações Node.js em produção no Serpro utilizem preferencialmente a versão estável com suporte ativo mais longo (a 20.16.0). O uso de versões de Node.js sem suporte de segurança é um risco para a empresa e para seus clientes.

Considerando que o suporte da versão 20.16.0 de Node.js termina em julho de 2026, devem ser tomadas providências para migrar as aplicações para a versão 24 até o final de 2025. A versão mais antiga com suporte ativo é a 18, suportada até julho de 2025.

A existência de versões de Node.JS anteriores à 18 é uma situação temerosa, com gravidade inversamente proporcional ao número da versão. Se existem aplicações em produção nessa situação, elas devem ser migradas com máxima urgência. Caso não seja possível uma migração direta para a versão 20.0 por causa do acúmulo de incompatibilidades, a migração pode ser planejada para ser feita mais de uma etapa (por exemplo, migrando para a versão 18 e depois para a versão 20).

3.Conclusão

Para um uso funcional e seguro de TLS com Node.js, é recomendável o uso mínimo da versão 18, considerando que o suporte ativo será encerrado em alguns meses.
A recomendação para versões de Node.js inferiores a 18 é apenas uma: migração imediata para a versão com suporte ativo mais recente possível.

Este documento foi finalizado em 6 de agosto de 2024.