摘 要

本协议基于 AES-GCM 和 x25519 实现了类似 TSL1.3 协议的功能，在仅两方知道密码又不要求证书的情况下，仅需 1-RTT 做到内容安全、端点可靠认证，无法通过主动探测、重放攻击、机器学习等方法区分本协议与真实 TLS1.3。并且为了提高机器和网络性能，在不影响安全的前提下，可以选择性地加密和使用网络多路复用。

关键词：TLS1.3；信道安全；伪造 TLS

Abstract

This protocol implements functions similar to TSL1.3 protocol based on AES-GCM and x25519. When only two parties know the password and do not require a certificate, only 1-RTT is required to achieve content security and reliable endpoint authentication. It is impossible to distinguish this protocol from real TLS1.3 through active detection, Replay attack, machine learning and other methods. And in order to improve machine and network performance, selective encryption and network multiplexing can be used without affecting security.

Keywords: TLS1.3; Channel Security

目 录

符号和缩略语说明

Server	服务端
Client	客户端
TLS	传输层安全性协议 (Transport Layer Security)
ECDHE	椭圆曲线迪菲-赫尔曼金钥交换 (Elliptic-curve Diffie–Hellman)

第 1 章　协议现状

最新版本为 V3，最后更新于 2023 年 9 月 2 日，无协议其他变种。唯一官网：https://github.com/JimmyHuang454/JLS。文档语言：中文简体。

目前支持 JLS 协议的软件工具有：

- https://github.com/vincentliu77/quinn-jls
- https://github.com/JimmyHuang454/sing-box

1.1　Copyright

使用最宽松、最自由的版权协议 WTFPL。

1.2　贡献与反馈

请移步到 github 官网，根据需要提出问题或贡献。所有贡献代码都应符合 WTFPL。

1.3　Change Log

1.3.1　V3.1

2023 年 9 月 2 日

废弃 V2 版

2023 年 7 月 28 日

修改密码的生成算法。

1.3.2　V3

2023 年 7 月 2 日

完全不同于 V2，V2 的设计理念是在安全的情况下，尽可能地简化流程。V3 版本基于 golang 的标准库，完整地实现了 TLS 的各种功能，实际代码量很少，完美替换原有 TLS。

1.3.3　V2

2023 年 6 月 30 日

更新 Application Data 加解密时所使用的 IV，使其更加随机。

1.3.4　V1

2023 年 6 月 24 日，初始版本

第 2 章　Specification V3

第 3 个版本跟 TLS 1.3 完全一模一样，除了 random 字段是特殊生成的。

2.1　基本原理

因为 TLS1.3 会生成 KeyShare 来保证此次对话是唯一的。因为 random 生成时加入了 KeyShare，所以攻击者是无法伪装成 Client 或 Server。

2.2　random 生成算法

生成一个随机数 N（16 字节），通过用户输入的 userIV 和 userPWD，使用 AES_256_GCM 对 N 加密。加密时，先把 ClientHello 和 ServerHello 中的 random 字段的 32 字节全部置换为 0，得出 Hello。

实际使用的:

- 原文 plainText=N
- 随机数 iv=sha256(utf8.encode(userIV) + Hello)
- 密码 pwd=sha256(utf8.encode(userPWD) + Hello)
- 附加消息 additionalText=null

加密后生成出默认长度为 16 字节的消息认证码 MAC 和 16 字节的密文 cipherText，通过拼接 cipherText + MAC 得出 32 字节的 FakeRandom 后填充进 Hello。

2.3　PSK 处理

如果 Client Hello 包含 Pre_Shared_Key 拓展，生成 FakeRandom 时，Pre_Shared_Key 应该包含真实的 Identity 参数，但 Binder 参数应该全置为 0（长度应与真实长度一致）；生成完 FakeRandom 并填充进 Client Hello 后，再计算并填入真实 Binder，因为 Binder 会验证实际使用的 Client Hello。

2.4　Server 验证

Server 根据收到 Client Hello 的 random 来判断是否有效 Client，如果不是有效 Client，直接转发到伪装站。如果是有效 Client，则不需转发到伪装站，使用自签证书，完整按照 TLS 的流程处理。Server 发送的证书可以是任意内容，Client 无需

验证该证书是否有效。

2.5　Client 验证

Client 根据收到的 Server Hello 的 random 来判断是否有效 Server，如果不是有效 Server，则应表现成一个正常 HTTP 请求。如果是有效 Server，则不需要验证证书是否有效（直接信任），最后按照正常 TLS 流程处理即可。

2.6　安全建议

- 如果开启了 0-RTT 功能，那么可能遭遇重放攻击
- 如果选取支持 0-RTT 的伪装网站，相应的，你应该开启 0-RTT 功能来避免特征，而且如果伪装网站支持 Session 共享（即不同机器不同 IP 也接受 PSK），那么可能会暴露特征
- 最好每三个月更换一次密码，密码长度应该大于 64 字节

参考文献