HybriLink：融合对称密码与公钥密码的可验证安全信息传输系统

作者：NeuroSaki987

日期：2026-01-11

摘要

本文提出并实现一个面向点对点信息传输的混合密码系统 HybriLink。系统在握手阶段利用公钥密码完成身份认证与会话密钥协商，在数据阶段使用对称认证加密（AEAD）提供高吞吐的机密性与完整性保护。为降低单一会话密钥长期使用带来的风险，HybriLink 在数据通道中引入简化的对称链式密钥演进（symmetric-key ratchet），实现“每消息一把密钥”。此外，本文提出一个更偏工程落地的原创要点：将握手 transcript 绑定到密钥派生（Transcript-Bound Key Schedule, TBKS），使派生出的会话密钥显式绑定协商上下文，降低跨协议复用与降级误配造成的风险。论文给出协议消息格式、密钥派生、重放保护、安全性分析以及 Python 工程实现与测试。

关键词：混合加密；X25519；Ed25519；HKDF；AES-GCM；前向安全；密钥演进

1 引言

在不可信网络上进行信息传输时，攻击者可能被动窃听或主动篡改数据。工程上常用的解决思路是混合加密：用公钥密码在会话建立阶段进行认证与密钥交换，用对称密码在会话数据阶段进行高速加密。TLS 1.3 是该范式的典型代表 [1]。

HybriLink 的目标是提供一个工程可运行且结构清晰的端到端加密传输系统。

2 相关工作

- TLS 1.3：基于（EC）DHE 建立共享秘密，提供前向安全，使用签名/证书体系认证服务器 [1]。

- SSH 传输层：通过密钥交换协商会话密钥，并使用对称加密与完整性校验保护连接 [7]。

- Signal Double Ratchet：在消息系统中通过对称 ratchet 与 DH ratchet 组合提供更强的安全性属性（如 post-compromise security）[6]。

HybriLink 并不替代上述成熟协议；它更接近“可学习、可移植”的原型，用于展示混合结构的核心机制。

3 威胁模型与安全需求

假设攻击者具备以下能力：

- 被动监听：读取链路上全部数据；

- 主动攻击：篡改、插入、重放、丢弃数据帧（典型 Dolev-Yao 网络模型）。

系统安全需求：

- 机密性：未持有会话密钥者无法得到明文；

- 完整性：篡改应被检测；

- 认证：客户端可验证“对端确为预期服务器”（防止 MitM）；

- 有限前向安全：链密钥推进后，泄露当前链密钥不应解密历史消息。

4 系统架构概述

HybriLink 由两部分构成：

1) 握手层（Public-Key Handshake）

- 双方生成临时 X25519 密钥对并执行 ECDH [2]；

- 服务器使用长期 Ed25519 身份密钥对握手字段签名 [3]；

- 使用 HKDF-SHA256 派生会话 master key 与双向链密钥 [4]。

2) 记录层（Symmetric Record Layer）

- 每帧都采用 length-prefix framing（4B 长度 + payload）；

- 数据加解密采用 AES-256-GCM（AEAD）[5]；

- 每条消息使用对称 ratchet 推导独立 message key，随后推进 chain key。

5 协议设计

5.1 密钥材料

- 服务器长期身份密钥：Ed25519 私钥（服务器保存），公钥通过“预置/证书/指纹”方式交付客户端。

- 会话临时密钥：X25519 临时密钥对，按连接/会话生成并只使用一次。

5.2 消息格式

5.2.1 ClientHello

字段（固定长度）：

- type=1 (1B)

- version (1B)

- suite (1B)

- client_nonce (16B)

- client_eph_pub (32B, X25519)

5.2.2 ServerHello

字段：

- type=2 (1B)

- version (1B)

- suite (1B)

- server_nonce (16B)

- server_eph_pub (32B, X25519)

- sig_len (2B) + signature (Ed25519)

签名覆盖：

- tbs = ClientHello || ServerHello_without_signature

- signature = Sign_ed25519(server_id_priv, tbs)

客户端验证：

- Verify_ed25519(server_id_pub, signature, tbs)

该设计确保中间人无法替换 server_eph_pub，否则签名验证失败。

5.3 原创要点：TBKS（Transcript-Bound Key Schedule）

TBKS 的核心思想是：将握手 transcript（包含双方 hello 与签名）纳入密钥派生的上下文绑定。

- 让会话密钥“绑定”已协商的版本、套件、随机数与临时公钥；

- 降低跨协议/跨上下文复用共享秘密时的误配风险；

- 为未来扩展（算法协商、KEM 混合等）提供稳定的 key schedule 框架。

定义：

- transcript = tbs || signature

- salt = SHA256(transcript)

- IKM = ECDH_shared || client_nonce || server_nonce

- master = HKDF(IKM, salt, info="hybrilink|master", 32B)

- CK_c2s = HKDF-Expand(master, info="hybrilink|ck|c2s", 32B)

- CK_s2c = HKDF-Expand(master, info="hybrilink|ck|s2c", 32B)

- session_id = SHA256("hybrilink|sid|" || transcript)[0:16]

5.4 记录层与 AEAD

DataRecord payload：

- type=3 (1B)

- session_id (16B)

- counter (8B, u64)

- flags (1B)

- ciphertext (变长，含 GCM tag)

AEAD 设计：

- aad = flags || session_id || counter

- nonce(12B) = session_id[0:4] || counter(8B)

- key = message_key（来自对称 ratchet）

5.5 对称 ratchet（每消息密钥）

每方向维护链密钥 CK_i：

- MK_i = HMAC-SHA256(CK_i, "msg|" || i)

- CK_{i+1} = HMAC-SHA256(CK_i, "ck|" || i)

性质：

- 若攻击者在时刻 t 获得 CK_t，则无法回推 CK_{t-1}，因此历史消息密钥更难被回溯；

- 但攻击者可计算未来链密钥，因此不具备完整 post-compromise security（需引入 DH ratchet 才可增强）。

5.6 重放与乱序

原型实现采用严格顺序：

- 接收端要求 record.counter == 期望 counter，否则拒绝。

生产级实现建议：

- 维护滑动窗口（replay window）；

- 缓存 skipped keys（适配乱序与丢包）；

- 根据字节数/时间触发 DH rekey 或 KeyUpdate。

6 安全性分析（要点）

6.1 机密性与完整性

在随机数与 nonce 使用正确的前提下，AES-GCM 提供认证加密，能同时保证机密性与完整性 [5]。

6.2 抗中间人

服务器签名覆盖双方 hello 字段。攻击者若替换临时公钥将导致签名验证失败，从而无法诱导客户端导出攻击者可控的会话密钥 [3]。

6.3 前向安全

会话共享秘密基于 X25519 临时密钥交换 [2]，并结合对称 ratchet 的单向推进，使得在“密钥推进后泄露”的情况下，历史消息更难被解密。

6.4 实现风险与工程注意事项

- nonce 复用：AES-GCM 对同一 key 的 nonce 复用极其敏感 [5]。本方案每条消息使用独立 key，但仍建议在工程上坚持 nonce 唯一的编码规范。

- 随机数质量：client_nonce/server_nonce 与临时密钥必须来自安全 RNG。

- 密钥托管：服务器 Ed25519 私钥必须妥善保护；客户端必须以可信方式获得服务器公钥（预置/证书/TOFU 等）。

7 工程实现与复现实验

7.1 代码结构

- src/hybrilink/crypto.py：X25519/Ed25519/HKDF/AESGCM + ratchet

- src/hybrilink/handshake.py：握手流程

- src/hybrilink/channel.py：数据加解密通道

- src/hybrilink/server.py 与 client.py：可运行 demo

- tests/：单元测试

7.2 运行方式

1) 生成密钥：python -m hybrilink.gen_keys --outdir keys

2) 启动服务器：python -m hybrilink.server --server-ed25519 keys/server_ed25519.pem

3) 启动客户端：python -m hybrilink.client --server-ed25519-pub keys/server_ed25519_pub.pem --message "hello"

8 结论与未来工作

HybriLink 以“可验证、可运行、可扩展”为设计主线，展示了混合密码结构在传输系统中的工程落地方式，并通过 TBKS 将会话密钥与握手上下文绑定，增强协议层面的防误用能力。未来可扩展方向包括：

- 引入 DH ratchet（参考 Double Ratchet）以提升 post-compromise security [6]；

- 引入后量子 KEM 并与 ECDH 混合（hybrid PQ）；

- 支持乱序/丢包/多路复用与更完善的错误处理；

- 与现有框架（Noise/TLS）对比评测。

参考文献

[1] RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3.

[2] RFC 7748: Elliptic Curves for Security (X25519).

[3] RFC 8032: Edwards-Curve Digital Signature Algorithm (EdDSA).

[4] RFC 5869: HMAC-based Extract-and-Expand Key Derivation Function (HKDF).

[5] NIST SP 800-38D: Recommendation for Block Cipher Modes of Operation: GCM and GMAC.

[6] Signal Specifications: The Double Ratchet Algorithm.

[7] RFC 4253: The Secure Shell (SSH) Transport Layer Protocol.