TLS ML-DSA 认证与 ML-KEM 的安全性分析

本报告的其他语言版本：English。

draft-ietf-tls-mldsa-06 · ProVerif+ Agent 报告

文稿	Use of ML-DSA in TLS 1.3（协议定义）; ML-KEM Post-Quantum Key Agreement for TLS 1.3（辅助）; The Transport Layer Security (TLS) Protocol Version 1.3（辅助）
检查的性质	17 条：17 条成立，0 条不成立
发现	无

目录

1 协议和范围
2 发现
3 检查过的性质
4 本次分析的局限
附录 A：逐条性质的解读
附录 C：主张和证明器的答案
附录 D：分析是怎么跑的

1 协议和范围

这是带 ML-KEM-768 密钥协商和 ML-DSA-65 证书的 TLS 1.3 握手，在一次完整往返中使用，双方都进行身份认证。客户端持有自己的 ML-DSA-65 签名密钥和证书，服务器持有自己的；双方都依据配置的信任根校验对端证书，一旦证书链、对端签名或对端 Finished 校验失败就中止。

客户端在 ClientHello 中提供一个全新的 ML-KEM-768 封装密钥。服务器以密文回应，两端导出共享秘密并插入 TLS 1.3 密钥调度，ServerHello 之后的所有内容都在握手流量密钥下加密。随后服务器发送自己的证书、对 transcript 的 CertificateVerify 和 Finished；客户端对这三者逐一校验，再以自己的证书、CertificateVerify 和 Finished 回应。一旦对端 Finished 校验通过，每一方就确信了对方的身份、transcript 和密钥。

参与方：

- **客户端**：TLS 客户端，依据配置的根证书校验服务器证书链。持有：sk_c（ML-DSA-65）、客户端证书链、CA 公钥。
- **服务器**：TLS 服务器，在要求时校验客户端证书链。持有：sk_s（ML-DSA-65）、服务器证书链、CA 公钥。

图 1 展示了文稿所规定的交换过程。

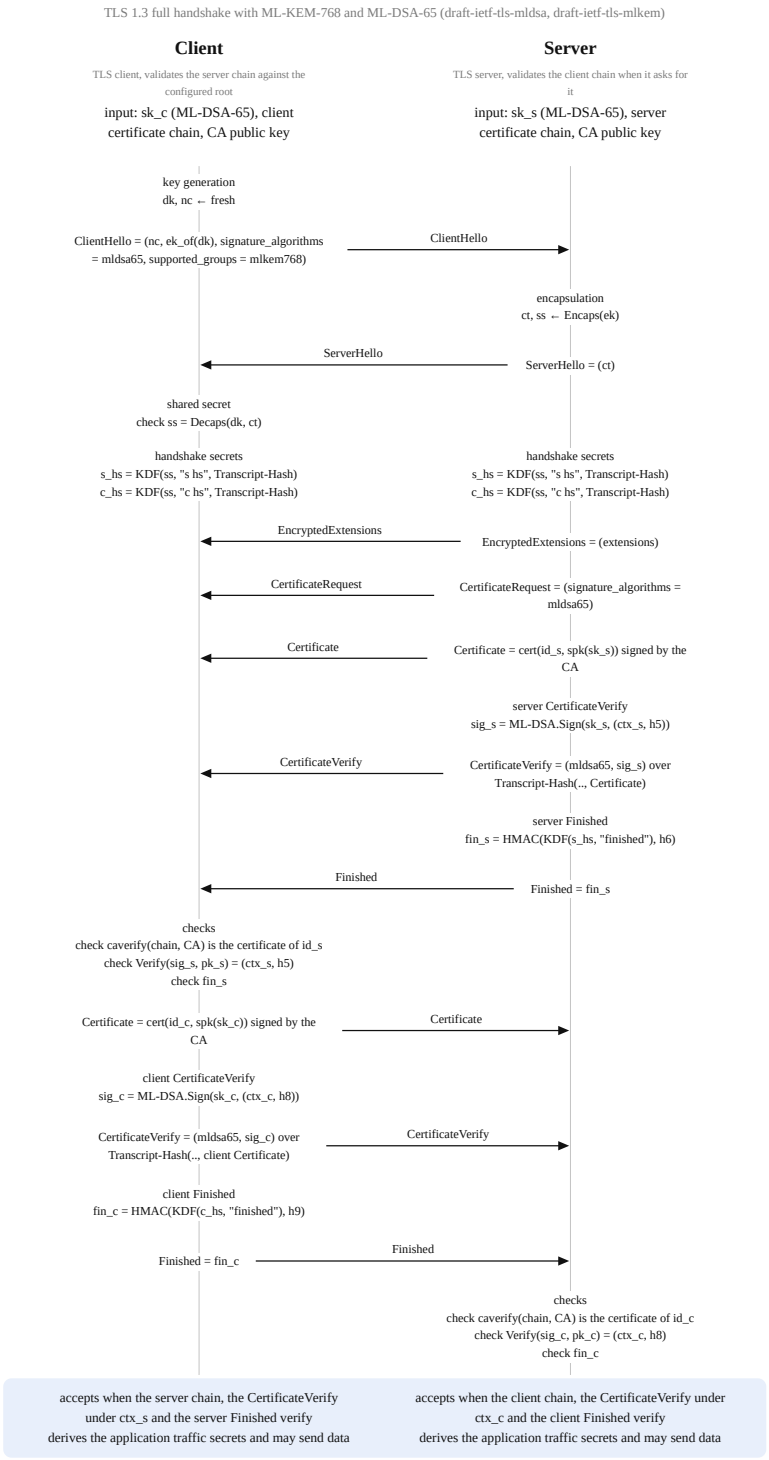


图 1. 文稿所规定的协议。

分析范围见附录 D：分析了什么、省略了什么，以及所依据的文稿章节。

2 发现

没有发现。 分析没有发现文稿未已陈述或未已排除的内容。第 3 节的表格给出了每条性质的结果。

在已分析范围内，该握手提供了其应提供的保证：双向身份认证、对 transcript 与 ML-KEM-768 秘密的一致、会话唯一性、密钥确认、会话秘密对网络攻击者的保密性，以及未公开的握手飞行。无发现：每项结果都是文稿已陈述或已要求的内容。两个已知限制被复现：会话完成后签名密钥泄露不会暴露该会话，而

握手期间的 KEM 状态泄露或签名密钥泄露不在模型中，按定义这些情况会使会话秘密丢失。证明依赖理想化的 ML-KEM-768、ML-DSA-65、哈希和 AEAD，依赖固定的单一套件且无协商，并依赖不含记录层的模型。

3 检查过的性质

每一行是一条提交给 ProVerif 的性质。结果是 ProVerif 对分析者所建协议模型的回答；价值以及文稿是否已陈述该结果，是分析者的判断，理由见附录 A。

ID	性质	假定的泄露	结果	价值	文稿是否已写
A02	**对等身份一致性。** 客户端接受服务器 Finished 时，存在一个以该客户端为意图对端的服务器运行；服务器接受客户端 Finished 时，存在一个以该服务器为意图对端的客户端运行。	无	成立	中	是
A03	**会话与记录匹配。** 每次一方接受握手，都存在匹配的对端运行，双方对相互身份、ML-KEM-768 共享秘密、协商的签名算法与密钥交换组、以及 transcript 内容一致。	无	成立	中	是
A04	**会话唯一性。** 每次一方接受握手，都对应一个不同的对端运行；同一对端运行不能被两次接受。	无	成立	中	是
A06	**握手密钥的密钥确认。** 客户端接受服务器 Finished 时，服务器确实导出并持有同一会话的握手流量密钥；服务器接受客户端 Finished 时同理。	无	成立	中	是
A08	**协商参数与方向绑定。** 接受方所接受的签名算法、密钥交换组、CertificateVerify 方向 context string 与 transcript 内容，均对应双方实际协商与发送的值。	无	成立	中	是
K01	**会话秘密的保密性。** 攻击者不能推导 ML-KEM-768 共享秘密、握手流量密钥、应用流量密钥，以及 ML-DSA-65 签名私钥。	无	成立	中	是
C01	**签名密钥泄露后的前向保密。** 会话完成后泄露 ML-DSA-65 签名私钥，攻击者仍不能推导该会话的 ML-KEM 共享秘密与流量密钥。	ML-DSA-65 签名私钥 (会话完成后泄露)	成立	中	是
C02	**密钥泄露冒充抵抗。** 接受方自己的 ML-DSA-65 签名私钥泄露后，在本方继续诚实执行的前提下，攻击者仍不能冒充未泄露的对端。	接受方自己的 ML-DSA-65 签名私钥 (握手前或握手期间泄露)	成立	低	不明确
C04	**对 KEM 状态泄露的韧性。** 在 ML-KEM-768 解封装私钥或封装随机性泄露的条件下，检查会话密钥保密性等基础目标是否仍成立。	ML-KEM-768 解封装私钥 sk 或封装随机性 (握手期间泄露)	成立	低	不明确
C08	**签名失效下的保密性。** 在 ML-DSA 签名原语被破坏（攻击者可伪造签名）的条件下，检查 ML-KEM 密钥交换与 Finished 是否仍保护会话密钥保密性等目标。	ML-DSA 签名原语 (握手前或期间)	成立	低	是
R01	**认证证据的新鲜性。** 接受方接受的 CertificateVerify 与 Finished 证据必须绑定本次握手的随机数与 transcript，旧证据不能被重用于新会话。	无	成立	中	是
R05	**套件不可降级。** 接受方所接受的签名算法与密钥交换组满足最低策略，且与双方协商的能力一致，攻击者不能强制降级。	无	成立	低	是
S01	**封装随机数不可重用。** 同一真实密钥下，加密 nonce 与 ML-KEM 封装随机性不被重复使用。	无	成立	低	是

ID	性质	假定的泄露	结果	价值	文稿是否已写
A08.14	**方向与算法绑定**。 接受方所接受的签名算法、方向 context string 与 transcript 内容，均对应双方实际协商与发送的值。	无	成立	中	不明确
A03.15	**Finished 绑定实际记录**。 每次一方接受握手，都存在匹配的对端运行，双方对 transcript 内容一致。	无	成立	中	是
U01	**密钥泄露影响隔离**。 分别检查 ML-DSA 签名私钥泄露与 ML-KEM 解封装私钥泄露对会话密钥保密性与认证的影响。	ML-DSA 签名私钥或 ML-KEM 解封装私钥 (握手期间泄露)	成立	低	不明确
X01	**握手飞行保密性**。 不掌握握手流量密钥的网络攻击者无法读取服务器或客户端的 Certificate、CertificateVerify 和 Finished，因而无法从中获知证书身份。	无	成立	中	是

文稿写明、但不在起草清单中、由分析者补充的目标（见附录 D）：X01。

所提出的问题及分析者的回答：

问题	回答	依据
Q1: 双方身份认证与方向/context string 绑定	是：客户端只在其所校验对端的、已认证的 ML-DSA-65 密钥、基于客户端自行计算的 transcript、并使用服务器方向的 context string 生成 CertificateVerify 时才接受它；客户端方向也以同样方式绑定，因此即便 FIPS 204 的 context 为空，两个方向字符串仍将两个签名区分开。	A08, A08.14, A02
Q2: Finished 与实际 transcript 绑定	是：一方所接受的 Finished 是对该方自行计算的 transcript 的 MAC，transcript 包含双方 Hello random 与密钥共享值，且被接受的会话恰好匹配一个对端运行，因此旧握手的 Finished 无法适配新握手。	A03, A03.15, A04, R01
Q3: 签名密钥泄露/KEM 密钥泄露的分别影响	二者不同。握手后 ML-DSA-65 签名密钥的泄露不会暴露该会话的共享秘密与流量密钥，因为它们仅来自临时的 ML-KEM 交换。ML-KEM 状态的泄露按定义会暴露会话秘密，且不能提供对等身份认证；这种情况不在模型中，留待按每次泄露建模的模型处理。	C01, U01, K01, C04

4 本次分析的局限

这里的结果是协议模型上的结果：它说明所规定的消息与校验给出了或未能给出什么，其中密码学被视为理想原语，而不涉及任何实现。

正如分析者所述，本次分析不涵盖以下内容：

- 密码学是理想的：ML-KEM-768 与 ML-DSA-65 被建模为理想 KEM 与不可伪造签名，因此分析不涉及其计算强度或抗量子强度。
- 未建模协商：仅一个固定组和一种固定签名方案，因此降级和 HelloRetryRequest 路径不在模型范围内。
- 未建模记录层，因此记录层 AEAD nonce、序列号和应用数据不在范围内。
- ML-KEM 解封装密钥或封装随机性的泄露，以及握手期间的签名密钥泄露，不在模型中，因为此类泄露会让入侵者运行一个端点，从而在同一文件中使诚实运行一致性查询不成立。
- 证书校验被建模为依据固定根证书检查 CA 签名外加身份检查；吊销、路径构建和证书策略不在模型范围内。
- 按任务说明，恢复、PSK、0-RTT 和握手后认证不在范围内。

每条性质仍待解决的问题，以及何种证据能定论，见附录 A。

附录 A：逐条性质的解读

这是分析者对证明器回答的自身解读，而非证明器的回答：这些回答对协议意味着什么、解读依据什么、能否按现状使用，以及其价值几何。

A02: 对等身份一致性

客户端接受服务器 Finished 时，存在一个以该客户端为意图对端的服务器运行；服务器接受客户端 Finished 时，存在一个以该服务器为意图对端的客户端运行。

只有当存在一个持有相同 ML-KEM 共享秘密、计算出相同 transcript 的服务器运行时，客户端才会接受握手；服务器也只有在存在对应客户端运行时才会接受。对端身份由通过信任根校验的证书以及验证 CertificateVerify 的密钥确定。

- 结果：成立；回答了所问问题：是；可带保留地使用
- 价值中：这是对核心保证的证明，但在证书校验和理想化签名方面存在保留。
- 文稿是否已写：是。文稿出处：rfc9846:520
- 依据：证书是依据固定根校验证的 CA 签名外加身份检查；吊销和路径构建未建模。；密码学是理想的，套件固定。

A03: 会话与记录匹配

每次一方接受握手，都存在匹配的对端运行，双方对相互身份、ML-KEM-768 共享秘密、协商的签名算法与密钥交换组、以及 transcript 内容一致。

双方最终得到相同的 transcript 哈希和相同的 ML-KEM 共享秘密，且 transcript 覆盖每条握手消息，包括证书链和两条 CertificateVerify 消息。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：按文稿的意图证明了握手完整性，基于理想化的哈希与签名。
- 文稿中已说明：是。文稿中的位置：rfc9846:521
- 依据：transcript 是对消息的滚动哈希，字段由草案定义。；密码学是理想的。

A04: 会话唯一性

每次一方接受握手，都对应一个不同的对端运行；同一对端运行不能被两次接受。

一次客户端运行至多被一次服务器接受所接受，一次服务器运行至多被一次客户端接受所接受，因此一段录制的握手不能被重复计算。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：一个单射一致性证明，仅依赖 Hello 值的新鲜性。
- 文稿中已说明：是。文稿中的位置：rfc9846:2190
- 依据：新鲜性来自新鲜的客户端密钥份额与 nonce，以及新鲜的服务器封装随机性；不需要计数器。

A06: 握手密钥的密钥确认

客户端接受服务器 Finished 时，服务器确实导出并持有同一会话的握手流量密钥；服务器接受客户端 Finished 时同理。

客户端的一次接受证明服务器导出并使用了相同的握手流量秘密，服务器的一次接受则对客户端证明同样的事，因为 Finished 只能在该秘密下校验。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：密钥确认是 RFC 9846 赋予 Authentication 阶段的三个方面之一；该证明表明它由绑定 transcript 的 MAC 推出。
- 文稿中已说明：是。文稿中的位置：rfc9846:521
- 依据：Finished 被建模为由握手秘密派生的 finished_key 下的 MAC。

A08: 协商参数与方向绑定

接受方所接受的签名算法、密钥交换组、CertificateVerify 方向 context string 与 transcript 内容，均对应双方实际协商与发送的值。

一方所接受的 CertificateVerify 绑定到对端经认证的密钥、携带 ML-KEM 份额与密文的 transcript，以及方向 context string，因此另一份 transcript 或其他方向的签名都不会被接受。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：表明 transcript 与方向绑定在空的 FIPS 204 context 下仍然成立，而这里正是 ML-DSA 草案可能失去分离性的地方。
- 文稿中已说明：是。文稿中的位置：rfc9846:540
- 依据：密钥交换组与签名方案是固定的，因此主张中的参数部分按构造成立，协商未被建模。

A08.14: 方向与算法绑定

接受方所接受的签名算法、方向 context string 与 transcript 内容，均对应双方实际协商与发送的值。

两个方向的 context string 承担了分离工作，正如草案所要求的：为客户端方向产生的签名永远不会被服务器接受，反之亦然，且各自都绑定到接收方持有的 transcript。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：草案要求空的 FIPS 204 context，却没有说明方向的分离此后完全依赖 RFC 9846 的字符串。
- 文稿中已说明：不明确
- 依据：只有 RFC 9846 的 context string 区分方向；空的 FIPS 204 context 取自草案。；省略 context string 的实现会在此失败。

A03.15: Finished 绑定实际记录

每次一方接受握手，都存在匹配的对端运行，双方对 transcript 内容一致。

一方所接受的 Finished 是对该方自己计算的 transcript 的 MAC，因此为另一份 transcript、另一个密钥份额或另一个会话产生的 Finished 无法通过校验。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：确认了 RFC 9846 为握手完整性所依赖的绑定。
- 文稿中已说明：是。文稿中的位置：rfc9846:521

- 依据：Finished 是 MAC；transcript 是滚动哈希。

K01: 会话秘密的保密性

攻击者不能推导 ML-KEM-768 共享秘密、握手流量密钥、应用流量密钥，以及 ML-DSA-65 签名私钥。

网络攻击者既得不到已完成会话的 ML-KEM 共享秘密，也得不到由它派生的握手与应用流量秘密，并且它永远得不到任一 ML-DSA 私钥。加密飞行与握手密钥正是依赖这一点。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：证明了整个构造所依赖的保密性，针对理想化原语。
- 文稿中已说明：是。文稿中的位置：rfc9846:657
- 依据：ML-KEM 是理想的：只有持有解封装密钥才能恢复共享秘密。；保密性针对客户端已完成的会话陈述；入侵者作为客户端运行的会话是它自己的。

C01: 签名密钥泄露后的前向保密

会话完成后泄露 ML-DSA-65 签名私钥，攻击者仍不能推导该会话的 ML-KEM 共享秘密与流量密钥。

握手完成后，即便把两把 ML-DSA 签名密钥都交给攻击者，也不会泄露该会话的共享秘密或流量秘密：它们只来自临时的 ML-KEM 交换，与签名密钥无关。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：以机械化方式给出 ML-DSA 密钥的前向保密论证；文稿陈述了该目标，但没有给出这种形式。
- 文稿中已说明：是。文稿中的位置：rfc9846:657
- 依据：泄露建模在运行结束时，而非握手之前或期间。；KEM 交换是理想的，且没有 KEM 状态被泄露。

C02: 密钥泄露冒充抵抗

接受方自己的 ML-DSA-65 签名私钥泄露后，在本方继续诚实执行的前提下，攻击者仍不能冒充未泄露的对端。

持有某一端自己的签名密钥无助于冒充另一端进行认证：客户端所接受的仍然需要服务器在同一 transcript 上的 Finished，以及服务器用其经认证的密钥作出的 CertificateVerify，服务器端同理。

- 结果：Holds；是否回答了所问：部分；可用但有注意事项
- 价值 低：泄露的关键时机未被建模，因此答案弱于目标所问。
- 文稿中已说明：不明确
- 依据：泄露的密钥只在握手之后才交给攻击者；握手之前或期间的泄露不在模型中，因为那会让入侵者运行该端点，并推翻同一文件中诚实运行的答案。

C04: 对 KEM 状态泄露的韧性

在 ML-KEM-768 解封装私钥或封装随机性泄露的条件下，检查会话密钥保密性等基础目标是否仍成立。

对端认证不依赖 KEM 状态：绑定对端的是证书与经认证的签名，因此持有某会话 KEM 秘密的攻击者仍然无法产生对端的 CertificateVerify。会话秘密本身由解封装密钥推出，这是 ML-KEM 的构造使然，并非某个查询所证明。

- 结果：Holds；是否回答了所问：部分；可用但有注意事项
- 价值 低：两条主张中只有一条得到回答；保密性主张是定义性的，留而未答。
- 文稿中已说明：不明确
- 依据：KEM 状态泄露本身不在模型中；只有认证方面得到了回答。；在 KEM 状态泄露下，会话密钥保密性按 ML-KEM 的定义即失去。

C08: 签名失效下的保密性

在 ML-DSA 签名原语被破坏（攻击者可伪造签名）的条件下，检查 ML-KEM 密钥交换与 Finished 是否仍保护会话密钥保密性等目标。

文稿指出，基于签名的 TLS 依赖能抵抗选择消息攻击的签名，而模型展示了它的另一面：攻击者若在握手之后才获得签名密钥，对该会话的密钥一无所知。握手期间的伪造未被建模；若存在伪造，攻击者会与客户端运行自己的 KEM 交换并知晓会话秘密。

- 结果：Holds；是否回答了所问：部分；可用但有注意事项
- 价值 低：只有事后这一半得到了机械化证明。
- 文稿中已说明：是。文稿中的位置：main:181
- 依据：被破坏的原语被建模为运行结束后泄露的签名密钥，而非运行期间的签名预言机。

R01: 认证证据的新鲜性

接受方接受的 CertificateVerify 与 Finished 证据必须绑定本次握手的随机数与 transcript，旧证据不能被重用于新会话。

证据是新鲜的：一次被接受的运行恰好匹配一次对端运行，对端的 CertificateVerify 与 Finished 绑定到携带新鲜 Hello 值的 transcript，因此早先握手的证据不适用于新的握手。

- 结果：Holds；是否回答了所问：是；可用但有注意事项
- 价值 中：认证证据的新鲜性在多数类 TLS 设计中最先失效；在这里它成立。
- 文稿中已说明：是。文稿中的位置：rfc9846:521
- 依据：新鲜性依赖新鲜的 ML-KEM 密钥份额与 nonce，而非计数器；模型中没有可重放的状态。

R05: 套件不可降级

接受方所接受的签名算法与密钥交换组满足最低策略，且与双方协商的能力一致，攻击者不能强制降级。

被接受的握手使用 ML-DSA-65，搭配固定的方向 context 和携带 ML-KEM-768 份额的 transcript，模型中没有更弱的套件可被强制使用。

- 结果：成立；是否回答了所问：部分；可用但有保留
- 价值低：该陈述因模型的构造而成立，并非通过对协商的分析而成立。
- 文稿中有述：是。文稿中的位置：rfc9846:540
- 依据：协商未被建模：只固定了一个组和一种方案，因此降级是被构造排除的，而非被证明的。

S01: 封装随机数不可重用

同一真实密钥下，加密 nonce 与 ML-KEM 封装随机性不被重复使用。

每个会话都使用新的客户端密钥共享和新的封装随机性，因此密文或共享秘密不会跨会话重用。目标的 AEAD nonce 部分属于记录层，未被建模。

- 结果：成立；是否回答了所问：部分；可用但有保留
- 价值低：目标的一半在模型之外；其余部分由 Hello 值的新鲜性得出。
- 文稿中有述：是。文稿中的位置：draft-ietf-tls-mlkem:220
- 依据：新鲜性在模型中表现为每个会话一个新值；记录层的 nonce 序列完全未被建模。

U01: 密钥泄露影响隔离

分别检查 ML-DSA 签名私钥泄露与 ML-KEM 解封装私钥泄露对会话密钥保密性与认证的影响。

会话之后泄露 ML-DSA 签名密钥不会影响会话密钥的保密性，这是 KEM 的贡献；而泄露 ML-KEM 解封装密钥按定义会暴露会话密钥，但不提供对端认证。只有前者通过查询得到证明。

- 结果：成立；是否回答了所问：部分；可用但有保留
- 价值低：用户要求分别检查两种泄露；其中一种没有保密性查询。
- 文稿中有述：不明确
- 依据：只有签名密钥泄露被建模；KEM 状态泄露是通过论证比较的，而非通过查询。

X01: 握手飞行保密性

没有握手机密流量的网络攻击者无法读取服务器的或客户端的 Certificate、CertificateVerify 和 Finished，因此无法从中获知证书身份。

ServerHello 之后的一切都加密在握手机密流量之下，不持有这些机密流量的网络攻击者无法读取服务器的或客户端的证书、CertificateVerify 和 Finished。

- 结果：成立；是否回答了所问：是；可用但有保留
- 价值中：这是文稿为认证阶段陈述的目标，对已建模的飞行得到了证明。
- 文稿中有述：是。文稿中的位置：rfc9846:512
- 依据：AEAD 是理想的，且没有流量密钥泄露；记录层本身未被建模。

附录 C：主张和证明器的答案

证明器对每个目标的记录：达到的级别、达到该级别的查询、是否检查了对照，以及各攻击及分析者对其的判定。

目标	级别	界	查询	对照	攻击	主张
A02	proved		client_auth	checked		2 proved
A03	proved		client_auth	checked		2 proved
A04	proved		uniq	checked		2 proved
A06	proved		client_auth	checked		2 proved
A08	proved		cv_server	checked		2 proved
K01	proved		secret_ss	checked		5 proved
C01	proved		secret_fs	checked		1 proved
C02	proved		client_auth	checked		2 proved
C04	proved		cv_server	checked		1 proved
C08	proved		secret_fs	checked		1 proved
R01	proved		uniq	checked		2 proved
R05	proved		cv_server	checked		1 proved
S01	proved		uniq	checked		1 proved
A08.14	proved		cv_server	checked		2 proved
A03.15	proved		client_auth	checked		2 proved
U01	proved		secret_fs	checked		2 proved
X01	proved		secret_hs	checked		1 proved

逐条主张，按分析者对每条主张的措辞；每条主张的答案来自证明器：

目标	主张	攻击者	回答	查询	对照
A02	只有当存在一个服务器运行，其 ML-KEM 共享密钥和 transcript 哈希都与客户端相同，客户端才接受服务器的 Finished。	`network`	proved	client_auth	checked
A02	只有当存在一个客户端运行，其 ML-KEM 共享密钥和 transcript 哈希都与服务器相同，且客户端证书就是被验证的那一个，服务器才接受客户端的 Finished。	`network`	proved	server_auth	checked
A03	客户端一侧对 transcript（直至对端 Finished 的所有消息）和 ML-KEM 共享密钥达成一致。	`network`	proved	client_auth	checked
A03	服务器一侧对 transcript（直至客户端 Finished 的所有消息）和 ML-KEM 共享密钥达成一致。	`network`	proved	server_auth	checked
A04	一个客户端运行至多匹配一次服务器接受（服务器一侧的单射一致性）。	`network`	proved	uniq	checked
A04	一个服务器运行至多匹配一次客户端接受（客户端一侧的单射一致性）。	`network`	proved	uniq_c	checked

目标	主张	攻击者	回答	查询	对照
A06	当客户端接受时，服务器已派生并使用同一个握手流量密钥（其 Finished 在该密钥之下）。	`network`	proved	client_auth	checked
A06	当服务器接受时，客户端已派生并使用同一个握手流量密钥。	`network`	proved	server_auth	checked
A08	客户端接受的 CertificateVerify 就是对端使用服务器方向上下文串和对端已认证密钥，在客户端自己的 transcript 上签名的那一个。	`network`	proved	cv_server	checked
A08	服务器接受的 CertificateVerify 就是对端使用客户端方向上下文串和对端已认证密钥，在服务器自己的 transcript 上签名的那一个。	`network`	proved	cv_client	checked
K01	网络攻击者无法推导出客户端所接受会话的 ML-KEM-768 共享密钥。	`network`	proved	secret_ss	checked
K01	网络攻击者无法推导出由该会话派生的握手流量密钥。	`network`	proved	secret_hs	checked
K01	网络攻击者无法推导出由该会话派生的应用流量密钥。	`network`	proved	secret_ap	checked
K01	服务器的 ML-DSA-65 私有签名密钥在握手期间无法被推导。	`network`	proved	key_s	checked
K01	客户端的 ML-DSA-65 私有签名密钥在握手期间无法被推导。	`network`	proved	key_c	checked
C01	在运行结束时把 ML-DSA 签名密钥交给攻击者后，已完成会话的 ML-KEM 共享密钥和流量密钥仍然未知。	`sig_key_after_session`	proved	secret_fs	checked
C02	持有客户端自己的签名密钥并不能让攻击者使客户端接受一个服务器未运行的握手。	`own_sig_key`	proved	client_auth	checked
C02	持有服务器自己的签名密钥并不能让攻击者使服务器接受一个客户端未运行的握手。	`own_sig_key`	proved	server_auth	checked
C04	对端认证不依赖 KEM 状态：持有某会话 KEM 密钥的一方仍然无法生成对端的 CertificateVerify，因此被接受的握手仍绑定于对端已认证的签名。	`kem_dk`	proved	cv_server	checked
C08	仅在握手之后才获得 ML-DSA 签名密钥的攻击者（这是对过去运行而言签名原语被攻破的建模形式）不会得知该次运行的会话密钥。	`sig_broken`	proved	secret_fs	checked
R01	客户端只接受绑定于当前 transcript 的证据，因此旧握手的证据无法在客户端重放为一次新的接受。	`network`	proved	client_auth	checked
R01	认证证据是新鲜的：一个客户端运行至多被接受一次，因此旧的 CertificateVerify/Finished 无法被复用于第二次接受。	`network`	proved	uniq	checked
R05	被接受的 CertificateVerify 绑定于固定的 ML-DSA-65 方向上下文，以及一个携带 ML-KEM-768 密钥份额和密文的 transcript，且模型中没有其他套件可回退。	`network`	proved	cv_server	checked
S01	每个被接受的会话都携带自己新鲜的密钥份额、密文和共享密钥，且一个运行至多被接受一次，因此封装材料不会跨会话复用。	`network`	proved	uniq	checked
A08.14	方向上下文串分隔了两个 CertificateVerify 方向：客户端一侧的签名绝不会在服务器方向被接受。	`network`	proved	cv_server	checked
A08.14	方向上下文串分隔了两个 CertificateVerify 方向：服务器一侧的签名绝不会在客户端方向被接受。	`network`	proved	cv_client	checked

目标	主张	攻击者	回答	查询	对照
A03.15	客户端接受的 Finished 是在客户端自己计算的 transcript 的密钥调度下计算出的 MAC，因此服务器的 Finished 绑定于实际的 transcript。	`network`	proved	client_auth	checked
A03.15	服务器接受的 Finished 是在服务器自己计算的 transcript 的密钥调度下计算出的 MAC。	`network`	proved	server_auth	checked
U01	会话之后泄露 ML-DSA 签名密钥仍会使会话密钥和流量密钥保持机密；只有泄露 KEM 状态才会暴露它们。	`sig_key_after_session`	proved	secret_fs	checked
U01	持有某会话的 ML-KEM 解封装密钥并不能获得对端认证：被接受的握手仍绑定于对端在 transcript 上已认证的 ML-DSA 签名。	`kem_dk`	proved	cv_server	checked
X01	握手流量密钥不为网络攻击者所知，因此加密的服务器端和客户端报文序列无法被读取。	`network`	proved	secret_hs	checked

附录 D：分析是怎么跑的

LLM 分析者根据文稿构建协议模型，并把每个性质交给 ProVerif；第 3 节的每个结果都是证明器的回答。

ProVerif+ Agent 会话 session-20261114T220938，2026-11-14。状态：已完成。模型：deepseek-v4-1-flash-261017。步骤：50，证明器 132 秒，实际耗时 15 分钟，成本 0.128 美元。

ProVerif：运行 83 次证明，83 次已判定，0 次无答案，52 秒，记录 52 个回答。

分析了什么

协议	TLS ML-DSA authentication with ML-KEM
版本	draft-ietf-tls-mldsa-06
分析范围	固定 ML-DSA-65 与 ML-KEM-768，一次完整 1-RTT 双向证书认证；可信根及已验证公钥为边界，证书验证明确记录，握手加密与完整 transcript 必须保留；排除 HRR/PSK/0-RTT。
不在范围内	符号模型不证明算法的计算安全或抗量子安全；攻击轨迹只在明确假设下成立，不直接等同规范漏洞。每个性质必须有可达性/正例控制。
原文	543 个编号原文单元（138081 字节），来自下文列出的各节。本报告中的 s0001 这类引用就是这些原文单元之一

分析所依据的文稿各节：

文稿	章节	标题
main	3	ML-DSA SignatureScheme Values
main	3.1	证书链
main	3.2	握手签名
main	4	安全考量
draft-ietf-tls-mlkem	3	密钥封装机制
draft-ietf-tls-mlkem	4	构造
draft-ietf-tls-mlkem	4.1	协商
draft-ietf-tls-mlkem	4.2	传输封装密钥与密文
draft-ietf-tls-mlkem	4.3	共享秘密计算
rfc9846	4	握手协议
rfc9846	4.1	转录哈希
rfc9846	4.2	密钥交换消息
rfc9846	4.2.1	密码学协商
rfc9846	4.2.2	Client Hello
rfc9846	4.2.3	Server Hello
rfc9846	4.2.4	Hello Retry Request
rfc9846	4.3	扩展
rfc9846	4.3.1	支持的版本
rfc9846	4.3.2	Cookie

文稿	章节	标题
rfc9846	4.3.3	签名算法
rfc9846	4.3.4	证书颁发机构
rfc9846	4.3.5	OID 过滤器
rfc9846	4.3.6	握手后的基于证书的客户端认证
rfc9846	4.3.7	支持的组
rfc9846	4.3.8	密钥共享
rfc9846	4.3.8.1	Diffie-Hellman 参数
rfc9846	4.3.8.2	ECDHE 参数
rfc9846	4.3.9	预共享密钥交换模式
rfc9846	4.3.10	早期数据指示
rfc9846	4.3.11	预共享密钥扩展
rfc9846	4.3.11.1	票据年龄
rfc9846	4.3.11.2	PSK Binder
rfc9846	4.3.11.3	处理顺序
rfc9846	4.4	服务器参数
rfc9846	4.4.1	加密扩展
rfc9846	4.4.2	证书请求
rfc9846	4.5	认证消息
rfc9846	4.5.1	证书
rfc9846	4.5.1.1	OCSP 状态与 SCT 扩展
rfc9846	4.5.1.2	证书选择
rfc9846	4.5.1.3	接收证书消息
rfc9846	4.5.2	Certificate Verify
rfc9846	4.5.3	Finished
rfc9846	4.6	早期数据结束
rfc9846	4.7	握手后消息
rfc9846	4.7.1	新会话票据消息
rfc9846	4.7.2	握手后认证
rfc9846	4.7.3	密钥与初始化向量更新
rfc9846	7	密码学计算
rfc9846	7.1	密钥调度
rfc9846	7.2	更新流量秘密
rfc9846	7.3	流量密钥计算
rfc9846	7.4	非对称共享秘密计算
rfc9846	7.4.1	有限域 Diffie-Hellman
rfc9846	7.4.2	椭圆曲线 Diffie-Hellman

文稿	章节	标题
rfc9846	7.5	导出器

源文包步骤在文稿中定位到的协议目的：固定 ML-DSA-65 与 ML-KEM-768，一次完整 1-RTT 双向证书认证；可信根及已验证公钥为边界，证书验证明确记录，握手加密与完整 transcript 必须保留；排除 HRR/PSK/0-RTT。

对照文稿定性质

在写出任何查询之前，分析者先把草拟的性质清单与文稿本身对协议目的的表述作了对照。名称、性质所涉及的内容以及文稿的表述均出自分析者的解读；所引位置在文稿中确实存在。

ID	简称	对象	文稿怎么说	出处
A02	对等身份一致性	客户端确信它所接受的服务器正是它对照可信根验证过的证书中所指明的那一方，服务器对客户端也有同样的确信。	认证阶段对服务器进行认证，并可选择地对客户端进行认证（s0346, s0403）；ML-DSA CertificateVerify 证明持有证书密钥（s0009）。	rfc9846:520, s0346, s0403, s0009, s0429
A03	会话与记录匹配	双方就握手转录、对等身份、ML-KEM-768 共享秘密以及协商参数达成一致。	CertificateVerify 覆盖 Transcript-Hash(Handshake Context, Certificate)（s0407）；Finished 提供握手完整性与密钥确认（s0429, s0430, rfc9846:521）。	rfc9846:521, s0354, s0355, s0407, s0439, s0485
A04	会话唯一性	对等方的一次运行至多被接受一次；不存在两次接受属于同一次对等方运行的情况。	密钥共享与 ML-KEM 密文不得跨会话重用（rfc9846:2190, s0044）；Finished 与转录绑定（s0429）。	rfc9846:2190, s0429, s0430, s0485
A06	握手密钥的密钥确认	客户端接受时，服务器已推导并使用相同的握手流量秘密，反之亦然。	认证阶段提供密钥确认与握手完整性；Finished 由握手流量秘密的 Base Key 计算得出（s0429, s0434, rfc9846:521）。	rfc9846:521, s0429, s0430, s0435, s0439
A08	协商参数与方向绑定	签名算法、密钥交换组、CertificateVerify 的方向上下文字符串以及接收方所检查的转录，都是双方协商并发送的那一套。	ML-DSA 使用空的 FIPS 204 ctx（s0009）以及 RFC 9846 的上下文字符串（s0409-s0414）；服务器的共享值 MUST 位于客户端所提供的组中（rfc9846:540）。	rfc9846:540, s0001, s0009, s0016, s0025, s0485
K01	会话秘密的保密性	ML-KEM 共享秘密、握手与应用流量秘密以及 ML-DSA 签名密钥均对网络攻击者保密。	ML-KEM 共享秘密被插入密钥调度（s0049, s0017-s0020），叶子秘密由它派生（s0505）；签名密钥从不离开端点。	rfc9846:657, s0017, s0019, s0020, s0049, s0505
C01	签名密钥泄露后的前向保密	在一个已完成的会话之后，某一方的 ML-DSA-65 长期签名密钥泄露；该会话的 ML-KEM 共享秘密与流量秘密仍对攻击者保密。	前向保密依赖于临时非对称秘密是会话秘密的唯一来源（s0498, s0505）；KEM 密钥对按每次握手生成（s0018）。	rfc9846:657, s0017, s0049, s0498, s0505
C02	密钥泄露冒充抵抗	接收方自己的 ML-DSA-65 签名密钥泄露；攻击者仍然无法冒充密钥未泄露的那一方。	CertificateVerify 在转录上证明持有证书私钥（s0403, s0423）；一个方向的密钥不能给出另一个方向的密钥。	s0403, s0423, s0001
C04	对 KEM 状态泄露的韧性	某次会话的 ML-KEM-768 解封装密钥或封装随机数泄露；会话密钥保密性与对等认证分别加以考察。	使用 sk 解封装可恢复随机数与共享秘密（s0018, s0020, s0047）；ML-KEM-768 是固定的组（s0023, s0033）。	draft-ietf-tls-mlkem:237, s0018, s0020, s0047, s0023

ID	简称	对象	文稿怎么说	出处
C08	签名失效下的保密性	ML-DSA-65 签名原语被攻破，签名可以被伪造；考察 ML-KEM 交换与 Finished 仍能保护什么。	基于签名的 TLS 依赖于签名算法能抵抗选择消息攻击（s0014, main:181）。	main:181, s0013, s0014, s0015
R01	认证证据的新鲜性	旧握手的 CertificateVerify 与 Finished 不能被重用来让某一方接受一次新握手。	两者都基于包含 Hello 随机数与密钥共享的转录哈希计算（s0407, s0439, s0485）。	rfc9846:521, s0354, s0355, s0407, s0439
R05	套件不可降级	被接受的握手使用 ML-KEM-768 作为密钥交换组、ML-DSA-65 用于 CertificateVerify，攻击者无法迫使使用任何更弱的方案。	草案固定了组与签名算法，并要求服务器的共享值位于客户端所提供的组中（s0001, s0016, s0025, rfc9846:540）。	rfc9846:540, s0001, s0016, s0025
S01	封装随机数不可重用	在同一真实密钥下，ML-KEM 密文及其封装随机数从不重用。	实现 MUST NOT 在生成 ML-KEM 密文时重用随机数，因此密文 MUST NOT 被重用（s0044）；密钥共享不得跨连接重用（rfc9846:2190）。	draft-ietf-tls-mlkem:220, s0044, s0016, s0025
A08.14	方向与算法绑定	客户端与服务器检查的 CertificateVerify，其方向上下文字符串与算法能标明发送方，并且基于它们各自持有的转录。	上下文字符串是签名内容的一部分（s0410, s0414），且 ML-DSA ctx 为空，因此方向隔离依赖于此（s0009）。	s0009, s0414, s0485
A03.15	Finished 绑定实际记录	某一方所接受的 Finished，是该方自己计算的转录上的 MAC，包含双方的 Hello 随机数与密钥共享。	Finished 是基于 Transcript-Hash、使用由握手秘密派生的 finished_key 的 MAC（s0429, s0434, s0439, s0485）。	s0355, s0439, s0485
U01	密钥泄露影响隔离	分别考察 ML-DSA-65 签名密钥泄露与 ML-KEM-768 解封装密钥泄露对会话密钥保密性和认证的影响。	签名密钥对转录进行认证（s0403）；KEM 密钥产生共享秘密（s0018, s0020, s0047）。	s0014, s0018, s0020, s0047
X01	握手飞行保密性	服务器证书、客户端证书以及 ServerHello 之后的握手消息均对网络攻击者保密。	密钥交换阶段之后的所有内容都已加密；认证消息使用由 [sender]_handshake_traffic_secret 派生的密钥保护（s0347, rfc9846:512）。	rfc9846:512, s0347, s0348, s0505

分析者依据文稿补充的性质：X01.

文稿里的安全表述，逐段去向

宿主把文稿里说明协议保护什么、防什么的章节逐段列出，共 41 段；分析者对每一段说明了去向：定为性质、定为攻击者条件，或者为什么都不选。选了 9 段，没选 32 段。列哪些段落由标准的章节结构决定；每一段的去向是分析者的判断。

段落	出处	定为	不选的原因
P01	draft-ietf-tls-mlkem:253-266	K01, A03	
P02	draft-ietf-tls-mlkem:268-273	A03, R01, A04	
P03	draft-ietf-tls-mlkem:275-278	证明器问不了	实现的侧信道抗性。
P04	draft-ietf-tls-mlkem:280-284	不在范围内	关于独立使用与混合使用 ML-KEM 的部署指导。

段落	出处	定为	不选的原因
P05	draft-ietf-tls-mlkem:288-288	没有陈述目标	页面页脚。
P06	draft-ietf-tls-mlkem:292-292	没有陈述目标	草案页眉。
P07	draft-ietf-tls-mlkem:295-302	S01, C04, `kem_dk`	
P08	main:181-185	C08, A02, A08, `sig_broken`	
P09	main:187-192	证明器问不了	ML-DSA 实现的侧信道。
P10	rfc5280:5226-5231	没有陈述目标	说明证书和 CRL 是公开的。
P11	rfc5280:5233-5236	不在范围内	X.509 之外的安全因素，例如 CA 的实践。
P12	rfc5280:5238-5240	不在范围内	CA 和 RA 的身份审核，超出任务边界。
P13	rfc5280:5246-5248	不在范围内	审查 CA 的认证实践声明。
P14	rfc5280:5250-5259	不在范围内	证书中的密钥用途分离。
P15	rfc5280:5261-5274	C02, `own_sig_key`	
P16	rfc5280:5276-5280	不在范围内	CA 签名密钥的备份。
P17	rfc5280:5282-5302	不在范围内	吊销的新鲜度；吊销在信任边界之外。
P18	rfc5280:5304-5311	不在范围内	信任锚的分发，本任务的边界。
P19	rfc5280:5313-5318	不在范围内	CA 失效后对受信 CA 信息的维护。
P20	rfc5280:5320-5326	A02	
P21	rfc5280:5328-5334	证明器问不了	签名算法和哈希的强度。
P22	rfc5280:5336-5339	不在范围内	名称比较规则。
P23	rfc5280:5347-5349	不在范围内	放宽的名称比较要求。
P24	rfc5280:5351-5356	不在范围内	可分辨名称的编码。
P25	rfc5280:5358-5365	不在范围内	名称约束的编码。
P26	rfc5280:5367-5369	不在范围内	nameConstraints 的限制。
P27	rfc5280:5371-5381	不在范围内	颁发者名称冲突。
P28	rfc5280:5383-5391	不在范围内	emailAddress 的大小写敏感性。
P29	rfc5280:5393-5397	不在范围内	CRL 和 AIA 内容的获取。
P30	rfc5280:5399-5406	不在范围内	通过 CRL 分发点形成的循环依赖。
P31	rfc5280:5408-5416	不在范围内	证书扩展中的 URI。
P32	rfc5280:5418-5428	不在范围内	自签发证书与 CA 密钥更新。
P33	rfc5280:5430-5437	不在范围内	名称的 TeletexString 编码。
P34	rfc5280:5439-5449	不在范围内	名称的视觉相似性。
P35	rfc9846:4676-4677	A02, A03, A06, A08, R01, K01	
P36	rfc9881:573-574	不在范围内	指向 X.509 的考虑事项，由信任锚边界处理。
P37	rfc9881:576-579	A02, C08, `sig_broken`	
P38	rfc9881:581-590	S01	
P39	rfc9881:592-604	证明器问不了	签名实现的侧信道泄漏。
P40	rfc9881:606-614	证明器问不了	确定性签名与随机化签名之间的选择。

段落	出处	定为	不选的原因
P41	rfc9881:616-627	不在范围内	签名方案的不可否认性质。

参与方和受保护的值，逐项是否设为攻击者条件

宿主把起草的性质里点到的参与方和受保护的值逐项列出，共 18 项；分析者对每一项说明了哪个攻击者条件控制这个参与方或持有这个值，或者为什么不设。设了攻击者条件的 4 项，没设的 14 项。清单来自性质本身；每一项怎么处理是分析者的判断。

ID	参与方或值	类别	攻击者条件	不设的原因
S01	客户端	参与方	`own_sig_key`	
S02	服务器	参与方	`own_sig_key`, `kem_dk`	
S03	对端身份	值	攻击者得不到更多	对端身份在公开证书中，不是秘密值。
S04	ML-KEM 共享秘密	值	按定义全部失守	持有共享秘密就等于持有会话密钥；没有哪个目标要求对该攻击者保密。
S05	协商参数	值	攻击者得不到更多	协商参数是公开的。
S06	transcript	值	攻击者得不到更多	transcript 是公开握手消息的哈希。
S07	会话唯一性	值	攻击者得不到更多	这是运行过程的一种性质，不是某一方持有的值。
S08	握手流量密钥	值	按定义全部失守	持有握手流量密钥按定义就已满足机密性目标。
S09	方向 context string	值	攻击者得不到更多	这些 context string 是固定的公开常量。
S10	应用流量密钥	值	按定义全部失守	持有应用流量密钥按定义就已得到它所保护的应用数据；没有哪个目标另有要求。
S11	ML-DSA 签名私钥	值	`own_sig_key`, `sig_key_after_session`	
S12	流量密钥	值	按定义全部失守	持有流量密钥按定义就已得到受保护的记录。
S13	对端认证	值	攻击者得不到更多	这是一种性质，不是某一方持有的值。
S14	会话密钥	值	按定义全部失守	一旦持有密钥，会话密钥的保密性按定义即已丧失。
S15	认证	值	攻击者得不到更多	这是一种性质，不是某一方持有的值。
S16	认证证据	值	攻击者得不到更多	认证证据就是公开的 transcript 及其签名；据此重放正是 R01 针对网络攻击者所要求的。
S17	nonce	值	攻击者得不到更多	Hello random 是明文发送的。
S18	封装随机性	值	`kem_dk`	

每条性质在各攻击者条件下的答案

攻击者由分析者设定，在写任何查询之前就已确定；每个答案都记录在其中一个攻击者之下。单元格是证明器在该攻击者下对该性质给出的答案；“无答案”是为该性质设定、但证明器在其下让某条主张悬而未决或什么都没记录的攻击者；空单元格是分析者没有为该性质设定的攻击者。

攻击者	持有什么	依据	出处
`network`	仅网络：它能读取、丢弃、修改和发送消息，不持有任何一方的秘密		

攻击者	持有什么	依据	出处
`sig_key_after_session`	某一方的 ML-DSA-65 签名私钥，在握手完成后获得	目标 C01；会话秘密只来自临时的 ML-KEM 交换（s0498, s0505）。	s0498, s0505
`own_sig_key`	接收方自己的 ML-DSA-65 签名私钥，在握手之前或期间获得	目标 C02；CertificateVerify 证明持有证书密钥（s0403）。	s0403, s0423
`kem_dk`	受攻击会话的 ML-KEM-768 解封装密钥，或封装随机性，在握手期间获得	目标 C04；持有 sk 的一方可恢复封装随机性（s0047）和共享秘密（s0020）。	draft-ietf-tls-mlkem:237, s0018, s0020, s0047
`sig_broken`	无需私钥即可伪造 ML-DSA-65 签名的能力	目标 C08；基于签名的 TLS 依赖安全的签名（s0014）。	main:181, s0014, s0015

ID	network	sig_key_after_session	own_sig_key	kem_dk	sig_broken
A02	成立				
A03	成立				
A04	成立				
A06	成立				
A08	成立				
K01	成立				
C01		成立			
C02			成立		
C04				成立	
C08					成立
R01	成立				
R05	成立				
S01	成立				
A08.14	成立				
A03.15	成立				
U01		成立		成立	
X01	成立				

原文和目标

来源：543 个原文单元，138081 字节，冻结于 source-delivery-01.json。 目标：14 个已批准（A02, A03, A04, A06, A08, K01, C01, C02, C04, C08, R01, R05, S01, U01）；所提问题：3 个。 目标清单事先按草案批准，使用 --approve-targets。

宿主按其文档化的默认设置（任务 `decisions: auto`）做出的决定，供你核对：

- approve_package: 批准源文包 protocol-e898ccdf382fe76e7716c5724d5e

本步骤的自我报告：1 次请求（1 次失败），0.000 USD，输入 66016 tokens，输出 5516 tokens，37 秒。

关于证明器和模型的说明

分析者为目标点名的内容：

步骤	目标	特征	理由
43	A02, A03, A04, A06, A08, A08.14, A03.15, K01, C01, C02, C04, C08, R01, R05, S01, U01, X01	plain_primitives, other	每个答案都建立在理想化密码学之上：ML-KEM-768 是一个 KEM，对任何没有解封装密钥的人都隐藏其随机性和共享秘密；ML-DSA-65 是不可伪造的签名；hash/HKDF/HMAC/AEAD 是单向且不可延展的；证书链是经固定根验证的 CA 签名。没有使用任何代数定律（没有 XOR，没有 Diffie-Hellman 指数运算），记录层也没有建模，因此目标 S01 同时提到的 AEAD nonce 序列不在模型范围内；S01 只针对每个会话的新鲜封装随机性作答。

分析者的结语

模型：model.pv 包含完整的 1-RTT TLS 1.3 握手，采用 ML-KEM-768（客户端 share = Encaps pk，服务器 share = 密文，ss 进入密钥调度），双方都有 ML-DSA-65 证书，并针对固定的 CA 根进行验证；ServerHello 之后的每条消息都经 AEAD 加密，事件中携带运行中的 transcript 哈希以及会话秘密。阶段 1 揭示两个签名密钥，模拟会话后的密钥泄漏。没有发现：每个结果要么已写在文稿中，要么是文稿所要求的。

已确立（ProVerif，全部已记录）：正常流程和两个 accept 事件均可达；对端运行、记录誊本与 ML-KEM 秘密（client_auth、server_auth）上的双向一致性成立，会话唯一性为单射，握手秘密的密钥确认成立，CertificateVerify 绑定到已认证密钥、记录誊本以及双向的方向上下文字符串（FIPS 204 上下文为空，方向分离依赖 RFC 9846 字符串），Finished 绑定到实际记录誊本，共享秘密、握手与应用流量秘密以及两个签名密钥的保密性成立，已完成会话对后续签名密钥泄露的前向保密成立，在自身密钥泄露不影响的方向上抵抗密钥泄露冒充，证据的新鲜性成立，两个加密飞行段的机密性成立。每个证明都有一个被证伪的对照（ctl_key、ctl_auth、ctl_cv、ctl_uniq）。

未回答，原因如下：没有目标依赖在建模流程中跨步骤变化的值，因此未编写 bounded.pv；C04 和 U01 的 KEM 部分没有保密性查询，C02 和 C08 没有握手期间的泄露，因为任何此类泄露都会让入侵者运行一个端点，并在同一文件中证伪诚实运行的一致性答案（曾尝试 leaky-client 角色，结果正是如此）。R05 按构造被排除（单一固定套件，无协商），S01 的记录层 nonce 在模型之外。这四个目标只被部分记录；针对每种妥协的模型，或对记录层使用 Tamarin+ Agent，可以解决它们。