DiStefano: Decentralized Infrastructure for Sharing Trusted Encrypted Facts and Nothing More

Morad Cristina [*], El Hassanie Noah[*]
[*]EPFL, Switzerland
Emails: {cristina.morad, noah.elhassanie}@epfl.ch

***Abstract*—DiStefano is an efficient, maliciously-secure *Designated-Commitment TLS* (DCTLS) framework that generates binding, private commitments on TLS 1.3-encrypted data for subsequent verification by a *designated* third party. It preserves the client's browsing privacy while offering robust security guarantees under malicious adversaries. DiStefano's modular design supports diverse zero-knowledge proofs and two-party computation methods, extending functionalities such as privacy-preserving credential issuance and selective data disclosure. This paper presents the architecture of DiStefano, which includes a multi-phase protocol design (handshake, query execution, and commitment), a precise security analysis over TLS 1.3 and an open-source implementation integrated into BoringSSL. The empirical evaluation under both LAN and WAN conditions demonstrates very low online latency, moderate bandwidth overhead and strong privacy, showing DiStefano is practical for modern browser integrations.**

I. Introduction

Transport Layer Security (TLS) [1] is a critical protocol for securing connections over the Internet, providing confidentiality and authenticity for communication between clients (C) and servers (S). TLS 1.3 incorporates numerous improvements over earlier versions, reducing handshake latency and removing weakened primitives. Despite its widespread use, scenarios arise where a client must export a *commitment* to confidential session data. Examples include credential verification (e.g., proving one's age or membership) and compliant data-sharing (e.g., private audits).

Designated-Commitment TLS (DCTLS) protocols, also called three-party TLS protocols, extend TLS by allowing a designated verifier (V) to partially observe or assist in the TLS session through *secure two-party computation* (2PC). This mechanism enables verifiable evidence of session data while ensuring that V doesn't learn additional plaintext content or the exact server identity. Existing DCTLS solutions (such as DECO, PageSigner, TownCrier, and Janus) only partially address relevant requirements, often restricting supported TLS versions, exposing the server's identity, or requiring specialized hardware.

This paper introduces *DiStefano*, a maliciously-secure DCTLS framework specifically designed for TLS 1.3:

- **Enhanced security:** DiStefano leverages state-of-the-art maliciously-secure 2PC and zero-knowledge proofs to protect both the content of TLS sessions and the identity of the server.
- **Cryptographic agility:** The framework supports standard ciphersuites (notably AES-GCM), and ECDSA certificate authentication.
- **Client privacy:** Through optional zero-knowledge certificate proofs, the verifier V is assured of a server's membership in a trusted set without knowing the specific server name.
- **Practicality:** Extensive experiments on LAN and WAN networks show that online phases complete in under one second. Our full open-source implementation is integrated into BoringSSL which is the only cryptographic library supported by Chromium-based Internet browsers.

A. Paper Outline

Section II states the overarching system objectives and adversarial model. Section III outlines the three-phase protocol approach (handshake, query execution and commitment). Section IV details the underlying 2PC building blocks, including optimized AES-GCM encryption/decryption, tag verification, and zero-knowledge TLS certificate proofs. Section V presents the formal security argument. In Section VI, it's described our open-source implementation in BoringSSL and empirically evaluate its performance under both LAN and WAN network settings. Section VII discusses practical deployment considerations and limitations. Finally, Section VIII concludes the paper with potential future directions.

II. System Goals

DiStefano aims to provide:

1) **Privacy-Preserving Commitments:** Client C should generate cryptographic commitments to TLS-encrypted data without revealing session plaintext or exact server identity and preserving private information.
2) **Malicious Security:** Even if either C or V behaves maliciously, the protocol protects integrity and authenticity: no party can forge or manipulate encrypted traffic or the derived commitments. Additionally, V must not learn information about the session beyond what is strictly allowed.
3) **Cryptographic Agility:** The design should allow for different elliptic-curve groups and ciphersuites, so that it can support evolving industry standards and offering robust long-term viability.
4) **Practical Performance:** Low latency online execution is desired, making DiStefano practicable for browsers

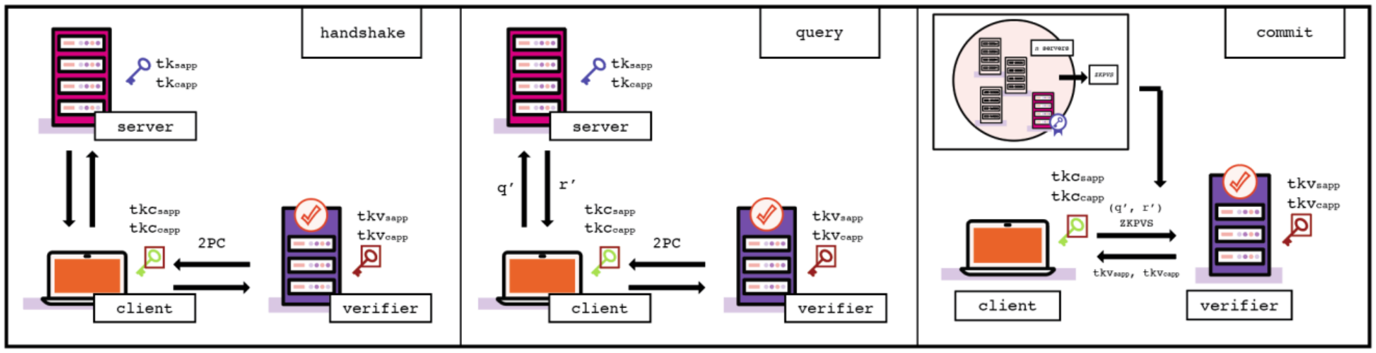

Fig. 1. High-level overview of DiStefano's three phases: Handshake, Query execution and Commitment. In each phase, *C* and *V* securely coordinate over shared TLS keys and produce verifiable commitments and tags.

and Internet services that impose stringent handshake and interaction timeouts.

III. Protocol Overview: Three-Phase Architecture

DiStefano's design follows the logic of TLS 1.3 but splits the client role across two parties, *C* and *V*, who coordinate via 2PC to share ephemeral key materials. Concretely, the protocol comprises three phases:

- **Handshake Phase (HSP):** A standard TLS 1.3 handshake is executed between *C* and *S*, except the key derivation is shared by *C* and *V* via 2PC. *V* receives only encrypted certificate data, preserving *S*'s identity unless a zero-knowledge proof discloses membership in a known set.
- **Query execution Phase (QP):** Using shared TLS record-layer keys, *C* and *V* cooperatively encrypt and decrypt request/response data to and from *S* in 2PC. GCM tag checks also occur in 2PC, ensuring no party can forge or manipulate record-layer data.
- **Commitment Phase (CP):** *C* provides binding commitments on the ciphertext blocks and optionally proves statements about them. *C* obtains *V*'s key shares only after the commitments are secured, guaranteeing that the committed data cannot be retroactively modified by *C*.

A high-level illustration of these three phases is shown in Fig. 1. Each phase is designed to be modular, allowing adjustments or replacements of specific cryptographic components (e.g., different ciphersuites or proof techniques).

IV. 2PC Foundations for DiStefano

A. Maliciously-Secure 2PC Primitives

DiStefano's security relies on standard 2PC protocols (garbled circuits, oblivious transfer, and additive/multiplicative secret sharing). Notable subprotocols include:

- **MtA:** Specialized multiplicative-to-additive transformations in $GF(2^{128})$ for efficiently generating shares of AES-GCM polynomials.
- **ECtF.** An "elliptic curve to field" routine converts elliptic-curve group elements into field representations in 2PC, enabling lightweight key derivation without expensive binary-circuit-based ECDH.

All these subprotocols carry malicious security, meaning if a party deviates from the specified computations then it will be detected.

B. Secure AES-GCM in 2PC

A large part of DiStefano's computation is 2PC-based AES-GCM encryption and decryption for TLS 1.3 records. Key techniques:

- *Counter-Mode Encryption in 2PC. C* and *V* hold additive shares of the AES key k. They evaluate AES.Enc(k, IV) in a garbled circuit, ensuring no plaintext is fully visible to *V*.
- *Tag Verification in 2PC.* GCM authentication tags are verified inside a 2PC environment to confirm integrity. Neither *C* nor *V* can modify the ciphertext blocks without being detected.
- *Commitments.* To render AES-GCM *binding*, DiStefano introduces ephemeral per-block commitments. These commitments prevent malicious re-interpretation of a ciphertext block under different keys.

Such optimizations ensure less than one second overhead for common record sizes (e.g., up to 16KiB).

C. Zero-Knowledge Proof of Certificate Validity (ZKPVS)

To preserve server anonymity, DiStefano uses an optional 1-out-of-N zero-knowledge proof of a valid TLS certificate signature. *C* proves to *V* that *S*'s certificate belongs to one of N authorized issuers or identities, without revealing which. The protocol relies on ECDSA-based certificate checks and proven ZK constructions.

V. Security Analysis

A. Handshake Security

The handshake phase extends formal TLS 1.3 security arguments by letting *C* and *V* share the client's ephemeral key material. This does not weaken the core integrity and confidentiality properties of TLS 1.3; rather, it introduces a unique property: V cannot learn the server's exact identity, but only that the server belongs to a trusted set of servers via a Zero Knowledge Proof.

B. Query Execution Security

Maliciously-secure 2PC ensures that AES-GCM operations on record-layer data remain authentic and confidential. If C or V attempts to substitute or modify plaintext/ciphertext, the GCM tag check would fail. Meanwhile, V does not learn the decrypted plaintext, preserving C's privacy.

C. Commitment Security

DiStefano enforces:

- **Binding property:** Once C commits to an encrypted record, it cannot modify the corresponding ciphertext afterward without being detected.
- **Hiding property:** V only sees additive shares of session keys and ephemeral commitments, gaining no additional knowledge of the plaintext unless C selectively reveals it.
- **Server anonymity:** The ZKPVS scheme assures that the server certificate is valid among N possibilities. The verifier learns only that the certificate is in the authorized set.

VI. Implementation and Performance Evaluation

A. Implementation in BoringSSL + emp

We implemented DiStefano in C++ (about 14k lines of code) within the BoringSSL library, which is a cryptographic library commonly used in Chromium-based browsers. For the 2PC primitives, we rely on the *emp* toolkit. Key technical highlights:

- **Carry-less Karatsuba Multiplications:** Used to expedite $GF(2^{128})$ multiplications in GCM tag verification, reducing the overall 2PC circuit size.
- **ECtF:** Converts elliptic-curve points to field elements in a maliciously-secure manner, avoiding a costly binary-circuit-based approach for ECDH.
- **Commitment Mechanism:** Employs lightweight AES-based hashing for ephemeral block keys, ensuring ciphertexts remain binding.

B. Experimental Setup

The performance is tested under both LAN (16 ms round-trip) and WAN scenarios (latencies up to 90–250 ms). Client (C) runs on a Macbook air M1 with 8 GB of RAM while the Server (S) and the Verifier (V) run on an Intel Xeon Gold 6138 with 32 GB of RAM. The measurements include:

- **Offline costs.** Circuit precomputation and OT setups, which can be amortized across sessions.
- **Online costs.** The time-sensitive handshake and record-layer operations.
- **Bandwidth usage.** The overhead introduced by 2PC data exchange.

C. Results

a) Handshake: Deriving TLS 1.3 handshake secrets in 2PC adds less than one second to typical handshake times under realistic latencies. Circuit precomputation can take a few seconds but is performed offline and can be reused across sessions.

b) Record-Layer Queries: For records up to 16KiB, encryption and decryption remain sub-second in LAN and at most a couple of seconds for higher-latency WAN links. This is acceptable for typical browsing, where individual page loading and multimedia streaming normally allow several seconds to receive the data.

c) ZKP-based Server Anonymity: The overhead from 1-out-of-N ZK proofs depends on the set size N. Typical usage (e.g., verifying the server belongs to a set of recognized identity providers) leads to manageable overhead. Larger sets incur higher computational cost.

Compared with previous DCTLS protocols, DiStefano shows strong malicious security, covers the entirety of TLS 1.3 and operates with practical overheads for both the handshake and query execution phases.

VII. Discussion and Limitations

Handling Large Sets of Servers. In advanced deployments, a verifier might permit a large set of potential servers, increasing the cost of 1-out-of-N certificate proofs. Systems requiring maximum anonymity, however, typically only use a moderate number of providers (e.g., identity providers or known membership organizations).

Denial-of-Service. The verifier must guard against adversaries that repeatedly request expensive 2PC sessions without legitimate cause. Rate-limiting or deposit-based strategies can be combined with DiStefano to mitigate these concerns.

Deployment Prospects. The strong security and user privacy of DiStefano suit many real-world scenarios:

- *Anonymous Credential Verification.* Age or membership checks without revealing other personal attributes.
- *Regulatory Data Audits.* A regulator verifies certain aspects of corporate data *without* the corporation revealing sensitive details.
- *Low-Trust Environments.* Partial trust relationships between client and verifier benefit from maliciously-secure 2PC to deter cheating.

VIII. Conclusion

This paper presented DiStefano, a maliciously-secure DCTLS protocol designed for TLS 1.3, which preserves privacy of both user data and server identity. Through two-party computation, DiStefano partitions the traditional TLS client role into client (C) and verifier (V), supporting efficient online protocols for handshake, record-layer encryption, and commitments. Experimental results in LAN/WAN conditions illustrate that DiStefano maintains less than one second overhead, enabling practical integration into web browsers. In future work, the plan is to explore additional multi-party or threshold variants and investigate post-quantum ciphersuites for protecting the long-term confidentiality of sessions.

References

[1] S. Celi, A. Davidson, H. Haddadi, G. Pestana, and J. Rowell, "DiStefano: Decentralized Infrastructure for Sharing Trusted Encrypted Facts and Nothing More," 2023. Available: https://github.com/brave-experiments/DiStefano