FINAL PROJECT: TAMPER-EVIDENT LOGS

CECS 478 SECTION 1

CHRISTOPHER BERMODES

WHY IS THIS IMPORTANT?

Logs play a vital role in current infrastructure

Monitoring, Detection, and Accountability

Recent example: SolarWinds cyber attack 2020

Security logs were manipulated for months to prevent early detection

It is estimated that attackers had more than 14 months of undetected access

Millions of dollars in damages

How tamper-evident logs can help

- Enhance Integrity of logs
- Can help reconstruct security incidents
- Validate incident timelines
- Enhance accountability of administrators

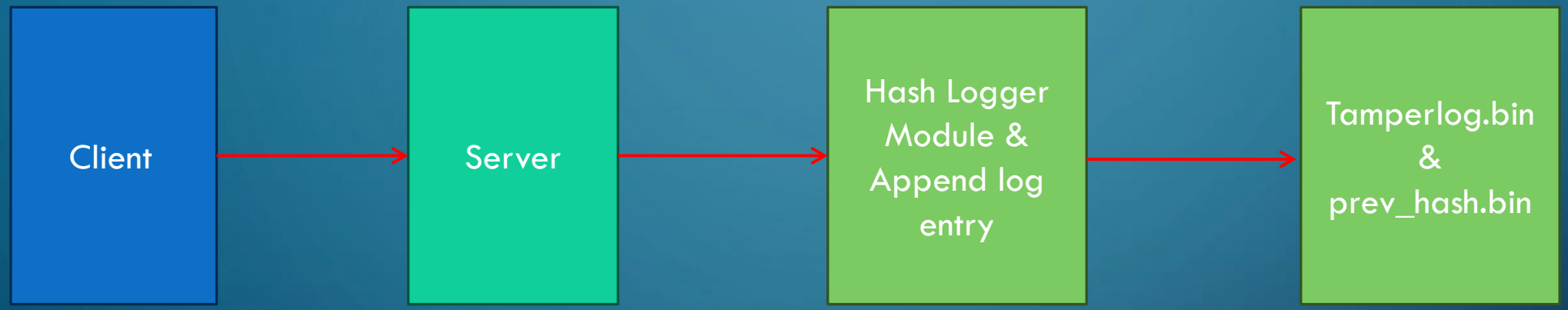
ASSETS

- Integrity of the event log
- Accuracy of recoded messages
- Hash chain continuity
- TLS-Secured communication

ASSUMPTIONS

- Main attacker is a malicious insider
- Server private key is protected
- File system access is restricted, but insider has access to it
- Attackers cannot make SHA-256 collisions (a.k.a. birthday attack) in a feasible timeframe

VERTICAL SLICE FLOW



1.Client sends TLS-encrypted message

2. Server receive message and records events using hash-chained logging

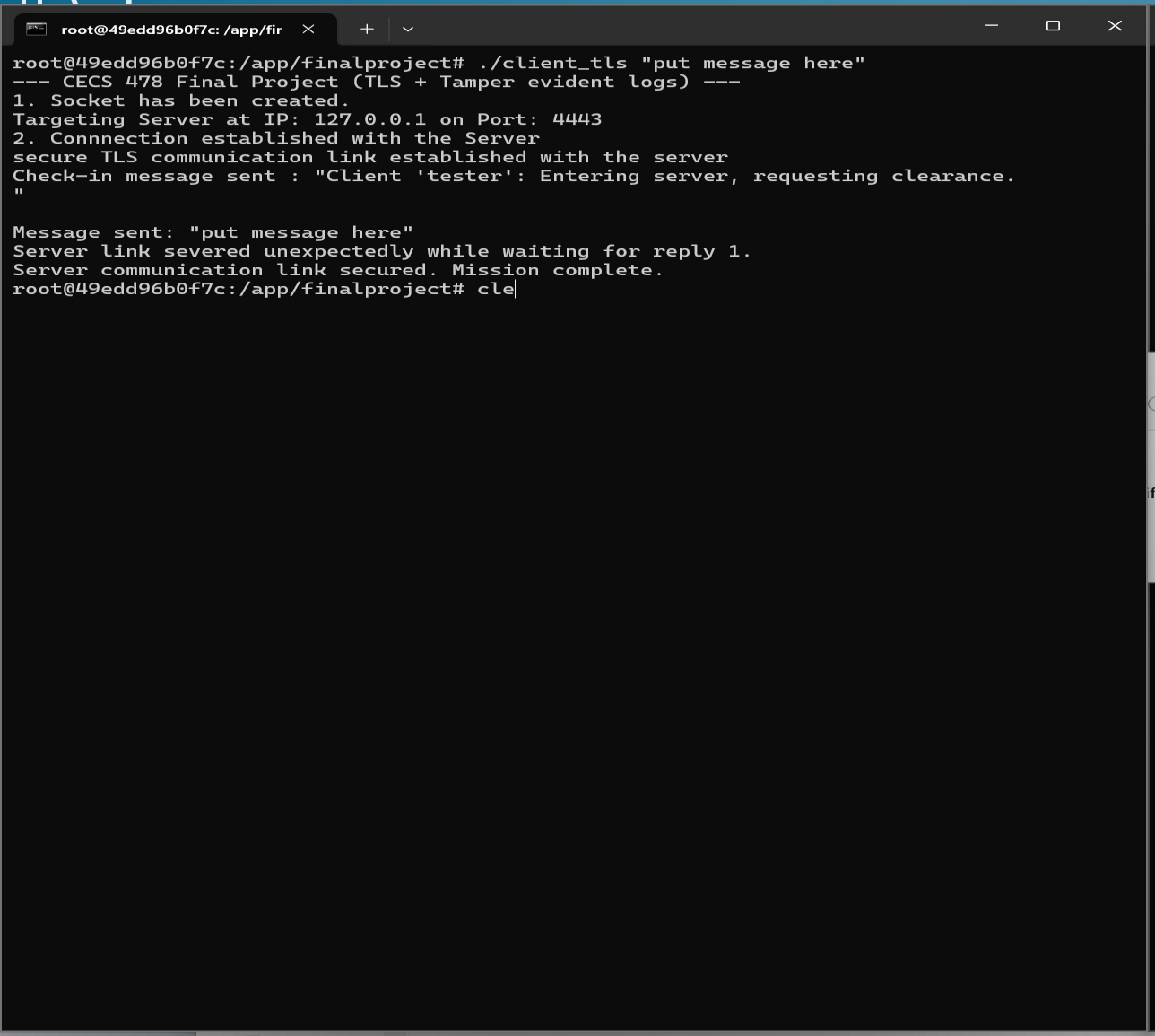
3. Logs are stored in binary format (tamperlog.bin, prev_hash.bin)

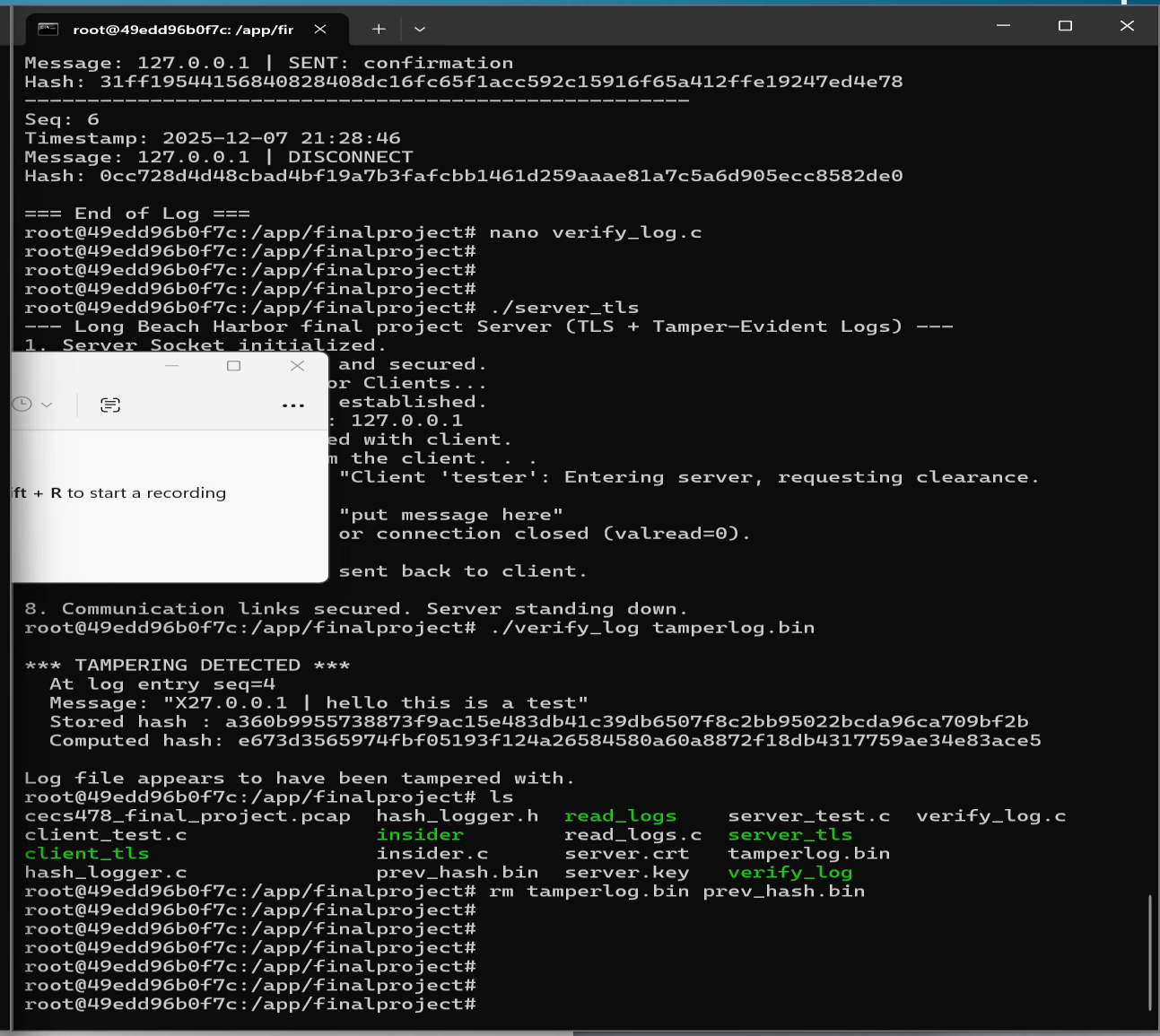
4. Insider tampers with message byte

5. verify_log tool detects the modification of log by recomputing entire hash-chain

6. Additional examination can be conducted with the use of read_logs tool – translate binary files into human readable language

MANUAL PROJECT DEMO RECORDING





LIVE DEMO

TESTS CONDUCTED

Alpha Tests (Functional Tests)

- **A1 — Happy Path Test**
  The server is launched, a TLS handshake is completed, and the client sends a message.
  The following were verified:
  - TLS handshake correctness
  - Log entry creation
  - Hash chain continuity
  - Verification success
  - No false detections
- **A2 — Negative Test: Insider Modification**
  A log entry was intentionally altered using the insider tool. Verification was expected to—and did—fail due to a hash mismatch.

Beta Tests (Unit + Edge Case Tests)

- **B1 — test_hash_logger**
  This test program writes multiple log entries using the logger module and confirms:
  - **Correct file growth**
  - **Correct struct formatting**
  - **Correct computation of curr_hash**
  - **Appending without corruption**
- **B2 — test_verify_log**
  This test loads both a clean log and a tampered log, verifying that:
  - **Clean log → accepted**
  - **Modified log → rejected**
- **B3 — Negative Test: Missing File**
  Verification was run on a non-existent log file.
  The program was expected to fail gracefully, confirming robust error handling.

TEST RESULTS - ALPHA

root@49edd96b0f7c: /app/fir

```
rm -f tamperlog.bin prev_hash.bin
== [A1] Alpha happy-path test ==
--- Long Beach Harbor final project Server (TLS + Tamper-Evident Logs) ---
1. Server Socket initialized.
2. Channel 4443 assigned and secured.
3. Server is listening for Clients...
--- CECS 478 Final Project (TLS + Tamper evident logs) ---
1. Socket has been created.
Targeting Server at IP: 127.0.0.1 on Port: 4443
2. Connnection established with the Server
4. New Client connection established.
 Client Location ID (IP): 127.0.0.1
secure TLS communication link established with the server
Check-in message sent : "Client 'tester': Entering server, requesting clearance.
"

Message sent: "alpha test message"
5. TLS handshake completed with client.
6. Awaiting messages from the client. . .
--- Received (56 bytes): "Client 'tester': Entering server, requesting clearance.
"
--- Received (18 bytes): "alpha test message"
 Client finished sending or connection closed (valread=0).

7. Official confirmation sent back to client.

Server link severed unexpectedly while waiting for reply 1.
Server communication link secured. Mission complete.
8. Communication links secured. Server standing down.
Verification complete: 6 entries verified.
Final chain head hash: 1401a24ac4d5d25882a2cd57343e3672cd418685f9a74203b39029ca731668f4
[A1] Alpha happy-path PASSED
== [A2] Alpha negative test (tampering) ==
[insider] Attempting to tamper with log file: tamperlog.bin
[insider] Target sequence number: 3
[insider] Found target entry (seq=3).
[insider] Original message: "127.0.0.1 | Client 'tester': Entering server, requesting clearance.
"
[insider] Modified message: "X27.0.0.1 | Client 'tester': Entering server, requesting clearance.
"
[insider] Tampering complete (hash NOT updated).

*** TAMPERING DETECTED ***
  At log entry seq=3
  Message: "X27.0.0.1 | Client 'tester': Entering server, requesting clearance.
"
  Stored hash : cb6baaf2a951d4c5d0f067daf24874de94dbc93a35c8b5696e5de46b62c98bde
  Computed hash: 2755ecbe4343248fe0228d09ce8158059dc64d575b67aa16bcbf5e79954e7ade

Log file appears to have been tampered with.
[A2] Alpha negative (tamper) PASSED
root@49edd96b0f7c:/app/finalproject_test1#
```

TEST RESULTS - BETA

root@49edd96b0f7c: /app/fir

```
root@49edd96b0f7c:/app/finalproject_test1# make beta
rm -f tamperlog.bin prev_hash.bin
gcc -Wall -Wextra -O2 --coverage -o test_hash_logger test_hash_logger.c hash_logger_build_1.0.c -lcrypto
gcc -Wall -Wextra -O2 --coverage -o test_verify_log test_verify_log.c hash_logger_build_1.0.c -lcrypto
== [B1] Unit tests for hash_logger ==
test_hash_logger: PASSED (log entries written, file size=230 bytes)
== [B2] Unit tests for verify_log ==
Verification complete: 2 entries verified.
Final chain head hash: 53aaa575455ab9fee48e1f8a11f09ccece95017007448e364a1f8514ca7f19b0

*** TAMPERING DETECTED ***
  At log entry seq=2
  Message: "127.0.0.1 | verify-test-message-◆"
  Stored hash : 53aaa575455ab9fee48e1f8a11f09ccece95017007448e364a1f8514ca7f19b0
  Computed hash: 6559fafdef0e46bede0f5ac9673b2016c527b9f365c243414278dfc1dd492148

Log file appears to have been tampered with.
test_verify_log: PASSED (clean log accepted, tampered log rejected)
== [B3] Extra negative test: verify on missing file ==
Failed to open log file: No such file or directory
[B3] Negative test (missing file) PASSED
root@49edd96b0f7c:/app/finalproject_test1#
```

CURRENT LIMITATIONS

- Current system has been only tested against manipulation attacks
- IP address is currently part of the message payload and can be tampered
- If attacker deletes tamperlog.bin and prev_hash.bin detection is not possible
- No real time detection of attack
  - Detection only occurs when verifier tool is activated manually

FUTURE WORK

- Testing against log deletion and insertion
  - Hash chaining should inherently detect these attacks, but further testing is required to confirm this assumption
- Stronger Insider attack models
  - Timestamp manipulation & header tampering
- Real time detection
- Multi-client support
- Log rotation system for better scalability
- Signed chain-head snapshots to prevent truncation and roll back attacks