Ahmed Rashwan

6051CEM Practical Cryptography – scenario

|  |  |
| --- | --- |
| Name | Ahmed Rashwan |
| SID | 12842713 |
| Date | 28/03/25 |
| Module Leader | Dr. Derrick Newton |
| Module Code | 6051CEM |

Contents

[Introduction 2](#_Toc194018606)

[Scenario Assumptions 2](#_Toc194018607)

[Custom Three-Way Communication Protocol (H&R Secure Contract Exchange Protocol - HRSCEP) 2](#_Toc194018608)

[Security applications justification 3](#_Toc194018609)

[Confidentiality 3](#_Toc194018610)

[Integrity 3](#_Toc194018611)

[Availability 3](#_Toc194018612)

[Authentication, Authorization, Accounting 3](#_Toc194018613)

[Non-Repudiation 4](#_Toc194018614)

[Chain of Trust via Third-Party Certificate Authority 4](#_Toc194018615)

[Proposed Exchange Communication Protocol Flow 5](#_Toc194018616)

[Diagram Workflow 5](#_Toc194018617)

[Limitations 6](#_Toc194018618)

[Potential vulnerabilities in Implementation 6](#_Toc194018619)

[Possible Latency 6](#_Toc194018620)

[Timestamping Authority is a Single Point of Trust 6](#_Toc194018621)

[Code Snippets of the Implementation 6](#_Toc194018622)

[Screenshots of Running Program Output 8](#_Toc194018623)

[Considerations for UK Laws 9](#_Toc194018624)

[UK eIDAS Regulation 9](#_Toc194018625)

[Electronic Communications Act 2000 9](#_Toc194018626)

[Data Protection Act 2018 9](#_Toc194018627)

[Conclusion 9](#_Toc194018628)

[References 10](#_Toc194018629)

[Appendix A 11](#_Toc194018630)

# Introduction

This purpose of this report is to propose a secure remote communication channel using a cryptographic protocol & key exchange signature between **H&R**, the **seller’s solicitor**, and **the buyer**. The report will recommend various suitable encryption algorithm to be used for the following:

* The buyer digitally signs the contract (legally binding) and returns it to H&R.
* H&R sends the signed contract to the seller’s solicitor.
* H&R will receive the contract from the seller’s solicitor and forward it to the buyer.

# Scenario Assumptions

* H&R and The solicitors are a wealthy financially and have a lot of financial freedom to invest into this project
* H&R and the sellers’ solicitors have a fully operational PKI already and all parties involved have already verified their digital certificates prior to starting this project
* TLS v1.3 is enforced across all parties involved therefore it effectively establishes confidentiality
* Every transaction that is made is timestamped using a third-party RFC 3161-complianct timestamping authority
* Third-party Certificate authority that handles giving out the digital certificates and is used to maintain a chain of trust and they will be the banks
* Tamper-proof logging is enforced and ensures all contract exchanges are securely recorded and can be audited in any future audit trails.
* The communication exchange protocol being proposed is compliant with ISO27001/2 and SOC 2 type II

# Custom Three-Way Communication Protocol (H&R Secure Contract Exchange Protocol - HRSCEP)

This is a proposed custom three-way secure communication protocol that will ensure the extended CIA model which will include confidentiality, integrity, availability, authorization, authentication & accounting as well as non-repudiation.

|  |  |
| --- | --- |
| **Principal** | **Application in protocol** |
| *Confidentiality* | Transport Layer Security (TLS v1.3) // End-to-End Encryption Using AES-256 |
| *Integrity* | SHA-3 for Hashing // Digital Signatures using RSA-4096 |
| *Availability* | Redundant secure servers & Timestamping servers that use RFC-3161 |
| *Authentication* | Public-Key Infrastructure (PKI) // Mutual TLS (mTLS) |
| *Authorization* | Role-Based Access Control |
| *Accounting* | Tamper-proof logging with signed syslog (RFC-5848) + SQL Ledger with cryptographic integrity checks |
| *Non-Repudiation* | Tamper-proof logs and SQL Ledger to verify integrity and provide non-repudiation |

Table : security principals and technical details

# Security applications justification

Each of the 7 stages from the extended CIA model which includes the CIA, AAA and non-repudiation has a corresponding technical security application that was chosen for this custom three-way security communication exchange, this section for the report will justify why each application was chosen and why it is the most suitable for this custom security mechanism that is being proposed.

## Confidentiality

Since H&R, the seller’s solicitor, and the buyer are all going to be working with sensitive information, confidentiality is a key aspect of the communication protocol that is being proposed. TLS version 1.3 is used as the main primary source of secure transport mechanism as it provides perfect forward secrecy (PFS) and has support for only up-to-date secure cryptographic algorithms, unlike TLS version 1.2 which included support for outdated hashing functions such as MD5, DSA, RC4, SHA1, and others which posed a major security issue as well as the version 1.2 of TLS did not support PFS and it was not heavily enforced, it was left optional (Malayek, 2024).

## Integrity

Integrity guarantees that the contracts that are exchanged have not been intentionally or unintentionally amended or altered during transmission, storage, or by someone eavesdropping on the communication. SHA-3 was chosen for this since It generates a 512-bit hash, the likelihood of a hash collision occurring is slim to none and makes for a better verification of integrity (Computer Security Division, 2017).

Moreover, the RSA-4096 digital signatures is being used for the benefit that RSA-4096 provides a stronger long-term security against the potential of brute-force attacks and man-in-the-middle attacks, it can also be used to alert and notify owners if an attacker tries to intercept or change the digital signature it will invalidate the digital signature which solidifies integrity for the purpose of this custom exchange protocol being developed.

## Availability

Availability in this specific scenario is ensuring that the contract exchange process and communication line are always accessible and fully operational, for the purpose of this custom exchange protocol being developed, redundant secure servers are deployed, there will be hot sites, cold sites, and warm sites to act as backups in the event of a disaster, and will be included as part of a disaster recovery and business continuity plan that the solicitors and H&R will have to be able to maintain availability and leave no room for single points of failure.

In addition, RFC-3161-compliant timestamping servers are going to provide proof that a contract was signed at a specific time by a specific person, and this will be used to prevent any disputes or any legal troubles related to the contract being signed, this will also support auditability with verifiable and non-deniable proof that a contract existed and was signed at a specific moment in time.

## Authentication, Authorization, Accounting

To maintain **Authentication** and make sure that only authorized parties can participate in the contract exchange. Public Key Infrastructure (**PKI**) is used to issue and manage digital certificates. Every party and entity involved will each have a cryptographic keypair to ensure all communication is authenticated securely. PKI is a trusted form of encryption and has benefits of scalability and wide applicability so it can easily be integrated and used by H&R, the solicitors, and other parties involved. mutual TLS (mTLS) will also be used this is to further improve and create a solid foundation for verification of identity as mTLS requires both sender and receiver to verify their identifies before a communication channel is created and established (Cloudflare, 2024).

In terms of **Authorization**, the specific type of access control recommended is role-based access control (RBAC) as this is the most secure and highly effective access control methodology which will ensure that by assigning specific roles and permissions prior, it will prevent unauthorized individuals gaining access and interfering in the process of exchange as well as block an attacker from being able to successfully privilege escalate to a more authorized user, RBAC also reduces the risk of any insider threats that may intrude and gain access or modify any files during the exchange that may interrupt the process.

**Accounting** ensures that all actions taken within the communication exchange protocol has been logged and can alter be verified. RFC 5848 signed syslog is used to maintain secure tamper-proof logs of contract exchanges by signing every log entry cryptographically which will prevent logs from being deleted or modified. Using the RFC 5848 signed logging guarantees that contracts signage can be audited and be used later in an audit trail for any legal and compliance matters (Clemm et al., 2025).

## Non-Repudiation

The SQL Ledger with cryptographic integrity checks will further ensure that every action is recorded and since the RSA-4096 are used for signing contracts, each contract that is signed is cryptographically linked to or bounded to its signer. Even if a party later denies signing the document, the signature and the logs In the SQL ledger will prove otherwise and server undeniable proof of the action taken.

# Chain of Trust via Third-Party Certificate Authority

Due to the sensitive nature of the transactions being processed as part of the communication protocol, the use of a third-party certificate authority (CA) is vital to ensure a valid chain of trust throughout the entire communication protocol. The certificate authority is important as they are the body that handles issuing, validating, and assigning a digital certificate to any party in need of a digital certificate for example before the buyer can send over the contract to H&R, they will need to obtain a digital certificate which will be issued via the third-party certificate authority.

Using a trusted third-party is involved in handling the digital certificates throughout the communication exchange protocol saves administrative overhead and can use that single verified party to ensure and verify integrity of the digital certificates being created and this establishes a secure chain of trust throughout the entire communication protocol.

# Proposed Exchange Communication Protocol Flow

This section of the report will breakdown the exact flow of the communication and exchange protocol proposed based on the specific scenarios mentioned.

When the buyer signs the contract, the buyer will generate a digital signature using RSA-4096, the hash of the contract being signed is hashing using the SHA-3 hashing algorithm. Both the contract and the signature will be stored in a tamper-proof SQL ledger with cryptographic security and integrity checks to be able to maintain integrity and non-repudiated. The contract is encrypted with AES-256 and transmitted via a TLS 1.3 secure communication channel over to H&R.

When H&R receive the contract & verify its integrity, H&R will use the SHA-3 hash to verify the integrity of the digital signature, if it is valid the contract will be stored in a redundant secure server to maintain availability and timestamp is recorded using RFC 3161-compliant timestamping authority which in this case will be a **third-party authority**. The event will be logged in a signed syslog (RFC-5848) and SQL Ledger for auditing purposes and maintaining an audit trail.

As soon as H&R are ready to forward the contract to the seller’s solicitor, the contract will be re-encrypted using AES-256 and sent over to the seller’s solicitor via TLS 1.3 with manual TLS to maintain confidentiality.

When the seller’s solicitor sends the contract back to H&R the contract’s integrity is verified again with a SHA-3 hash comparison to make sure that no integrity changes had occurred throughout the whole process and that H&R can securely forward the contract to the buyer for a final review. The SQL ledger will keep the logs of all the actions to maintain accountability and non-repudiation as well to act as a secondary source for audit trailing.

## A computer screen shot of a diagram AI-generated content may be incorrect.Diagram Workflow

Figure : Diagram of HRSCEP Workflow

# Limitations

Whilst this exchange communication protocol has been designed with strong security in mind, the communication exchange protocol does have some limitations and drawbacks that could be addressed in the future.

## Potential vulnerabilities in Implementation

The proposed communication exchange protocol is a very resource intensive protocol and very complex in its usage of different cryptographic tools, this may open the possibility of certain misconfigurations in the cryptographic libraries and can create different security risks if not implemented very carefully.

## Possible Latency

Since RSA-4096 is used for digital signatures and is resource intensive it may introduce the possibility of latency in big bulk transactions and operations especially since the contract exchange requires multiple levels of verification going from buyer → H&R → seller’s solicitor → H&R → buyer. This may be addressed in the future by switching to the elliptic curve digital signature algorithm like ECDSA-521 (cwd, 2011).

## Timestamping Authority is a Single Point of Trust

Every contract being exchanged relies heavily on the RFC3161-compliant third-party timestamping authority to verify and prove that a contract was signed, by who and when. If the timestamping authority (TSA) is compromised, suffers a data breach or becomes unavailable and availability is compromised it may cause legal issues and disputes over contracts validity and it may not be admissible in court anymore.

# Code Snippets of the Implementation

The code below explains the main functions of the program via comments in the code. Screenshots of the working code output is shown in the figures below and a full working version of the code can be found in [Appendix A](#_Appendix_A) .

# GLOBAL IMPORTS FOR PROGRAM  
from cryptography.hazmat.primitives.asymmetric import rsa, padding  
from cryptography.hazmat.primitives import hashes  
import base64  
import PyPDF2  
  
# =============================  
# FUNCTION DEFINITIONS SECTION  
# =============================  
  
# Function to extract readable text from each page in a PDF file  
def extract\_text\_from\_pdf(pdf\_path):  
 with open(pdf\_path, "rb") as file:  
 reader = PyPDF2.PdfReader(file) # Read the PDF  
 # Loop through all pages and extract text, then join with line breaks  
 text = "\n".join([page.extract\_text() for page in reader.pages if page.extract\_text()])  
 return text  
  
# Function to sign contract text using buyer's private key  
def sign\_contract(contract, private\_key):  
 # Convert the string contract into bytes (required by cryptography)  
 contract\_bytes = contract.encode("utf-8")  
  
 # Create a digital signature using PSS padding and SHA3-512 hash  
 signature = private\_key.sign(  
 contract\_bytes,  
 padding.PSS(  
 mgf=padding.MGF1(hashes.SHA3\_512()), # Mask Generation Function using SHA3-512  
 salt\_length=padding.PSS.MAX\_LENGTH  
 ),  
 hashes.SHA3\_512() # Secure hashing algorithm  
 )  
  
 # Encode the signature to base64 string for safe transmission/storage  
 return base64.b64encode(signature).decode()  
  
# Function to verify the integrity and authenticity of the signed contract  
def verify\_signature(contract, signature\_b64, public\_key):  
 # Encode the contract into bytes  
 contract\_bytes = contract.encode("utf-8")  
  
 # Decode the Base64-encoded digital signature back to raw bytes  
 signature\_bytes = base64.b64decode(signature\_b64)  
  
 try:  
 # Use the sender's public key to verify the digital signature  
 public\_key.verify(  
 signature\_bytes, # The received digital signature  
 contract\_bytes, # The original contract  
 padding.PSS( # Use the same padding scheme  
 mgf=padding.MGF1(hashes.SHA3\_512()),  
 salt\_length=padding.PSS.MAX\_LENGTH  
 ),  
 hashes.SHA3\_512()  
 )  
 return "[+]Signature is VALID. Contract integrity verified."  
 except Exception as e:  
 # If verification fails, the content or signature may have been tampered with  
 return f"[!]Signature is INVALID. Possible tampering detected!\nError: {e}"

## Screenshots of Running Program Output

A black screen with a black rectangle

AI-generated content may be incorrect.

Figure : running program output of communication protocol

# Considerations for UK Laws

## UK eIDAS Regulation

The electronic identification, authentication and trust services (eIDAS) Regulation mentions and includes digital signatures authenticity (Information Commissioners Office, 2024). It verifies that qualified electronic signatures (QES) hold the same legal proof as handwritten signatures in court for legal and compliance reasons. HRSCEP complies with this regulation by using the RSA-4906 protocol which can be certified under a UK-trusted certificate authority. In court the eIDAS takes more precedence over the Electronic Communication Act (ECA) 2000 as per post-Brexit UK laws (Information Commissioners Office, 2024).

## Electronic Communications Act 2000

The Electronic Communication Act (ECA) 2000 is used to confirm that electronic signatures are legally valid for agreements and transactions in the UK. The communication exchange protocol being recommended complies with this act as it uses RSA-4096 and SHA-3 for digital signatures which meet the UK’s legal requirements for authenticity and integrity for contracts. the use of mTLS and PKI also gives the communication exchange protocol edge because it means that digital contracts that are signed are verified and can be admissible in UK courts. (Gov.UK, 2011)

## Data Protection Act 2018

The Data Protection Act (DPA) 2018 requires organizations to protect personal data and ensure transparency in how data is being processed and stored. The proposed communication exchange protocol is compliant with the UK’s DPA 2018 by using AES-256-bit encryption to ensure that sensitive information both at rest and during transmission is encrypted, role-based access control is implemented to ensure and reduce the risk of insider threats and that only authorized individuals with the right roles can access contracts and data they are meant to. Tamper-proof logging is also implemented to provide a secure audit logging and audit trail that would meet UK Data Protection Act 2018 requirements on accountability and data access tracking. The blended use of multiple secure cryptographic tools creates a strong emphasis on confidentiality, integrity and availability ensuring the proposed communication protocol is compliant with the DPA 2018 (UK Government, 2018).

# Conclusion

The proposed H&R Secure Contract Exchange Protocol (HRSCEP) offers a set of comprehensive security and cryptographic solutions for engaging in secure communication and contract exchange between the buyers, the seller’s solicitors. The protocol ensures that confidentiality, integrity, availability, authentication, authorization, accounting, and non-repudiation are maintained making HRSCEP a robust communications protocol against cyber threats and attacks as well as being legally compliant under DPA 2018, eIDAS, and the Electronic Communications Act 2000.

# References

Clemm, A., Callas, J., & Kelsey, J. (2025). *RFC 5848: Signed Syslog Messages*. IETF Datatracker. <https://datatracker.ietf.org/doc/html/rfc5848>

Cloudflare. (2024). *What is mTLS? | Mutual TLS*. Cloudflare.com. <https://www.cloudflare.com/learning/access-management/what-is-mutual-tls/>

Computer Security Division, I. T. L. (2017, January 4). *SHA-3 Project - Hash Functions | CSRC*. CSRC | NIST. <https://csrc.nist.gov/projects/hash-functions/sha-3-project>

cwd. (2011, December 10). *Are there any disadvantages to using a 4096-bit encrypted SSL certificate?* Stack Overflow. <https://stackoverflow.com/questions/8453529/are-there-any-disadvantages-to-using-a-4096-bit-encrypted-ssl-certificate>

Gov.UK. (2011). *Electronic Communications Act 2000*. Legislation.gov.uk. <https://www.legislation.gov.uk/ukpga/2000/7/contents>

Information Commissioners Office. (2024, November 20). *Guide to eIDAS*. Ico.org.uk; ICO. <https://ico.org.uk/for-organisations/guide-to-eidas/>

Malayek, S. (2024, March 9). *TLS 1.2 Vulnerability*. Software.land. <https://software.land/tls-1.2-vulnerability/>

UK Government. (2018). *Data Protection Act 2018*. Legislation.gov.uk. <https://www.legislation.gov.uk/ukpga/2018/12/contents>

# Appendix A

# ================================  
# HRSCEP Secure Contract Exchange  
# ================================  
# GLOBAL IMPORTS FOR PROGRAM  
from cryptography.hazmat.primitives.asymmetric import rsa, padding  
from cryptography.hazmat.primitives import hashes  
import base64  
import PyPDF2  
  
# --------------------------------  
# Step 1: Key Pair Generation (Buyer Side)  
# --------------------------------  
print("[Step 1] Generating RSA-4096 Key Pair for Buyer...")  
  
# Buyer generates a private RSA key with a size of 4096 bits  
buyer\_private\_key = rsa.generate\_private\_key(  
 public\_exponent=65537,  
 key\_size=4096  
)  
  
# The corresponding public key is derived from the private key  
buyer\_public\_key = buyer\_private\_key.public\_key()  
print("[+] RSA Key Pair Generated.\n")  
  
# --------------------------------  
# Step 2: Extract Contract Text from PDF  
# --------------------------------  
print("[Step 2] Extracting contract text from PDF...")  
  
  
# Function to extract readable text from each page in a PDF file  
def extract\_text\_from\_pdf(pdf\_path):  
 with open(pdf\_path, "rb") as file:  
 reader = PyPDF2.PdfReader(file) # Read the PDF  
 # Loop through all pages and extract text, then join with line breaks  
 text = "\n".join([page.extract\_text() for page in reader.pages if page.extract\_text()])  
 return text  
  
  
# Path to the contract PDF  
pdf\_file\_path = "contract\_6051CEM\_Test.pdf"  
contract\_text = extract\_text\_from\_pdf(pdf\_file\_path)  
  
# Check if text was extracted successfully  
if not contract\_text:  
 raise ValueError("[!] Error: No text found. Make sure the PDF is not empty or scanned.")  
print("[+] Contract text extracted.\n")  
  
# --------------------------------  
# Step 3: Buyer Signs the Contract (Digital Signature)  
# --------------------------------  
print("[Step 3] Buyer signing the contract using RSA-4096 + SHA3-512...")  
  
  
# Function to sign contract text using buyer's private key  
def sign\_contract(contract, private\_key):  
 # Convert the string contract into bytes (required by cryptography)  
 contract\_bytes = contract.encode("utf-8")  
  
 # Create a digital signature using PSS padding and SHA3-512 hash  
 signature = private\_key.sign(  
 contract\_bytes,  
 padding.PSS(  
 mgf=padding.MGF1(hashes.SHA3\_512()), # Mask Generation Function using SHA3-512  
 salt\_length=padding.PSS.MAX\_LENGTH  
 ),  
 hashes.SHA3\_512() # Secure hashing algorithm  
 )  
  
 # Encode the signature to base64 string for safe transmission/storage  
 return base64.b64encode(signature).decode()  
  
  
# Generate the digital signature for the contract text  
digital\_signature = sign\_contract(contract\_text, buyer\_private\_key)  
print("[+] Digital Signature created.\n")  
  
# --------------------------------  
# Step 4: Verify Signature (H&R or Seller’s Solicitor)  
# --------------------------------  
print("[Step 4] H&R or Seller's Solicitor verifies digital signature...")  
  
  
# Function to verify the integrity and authenticity of the signed contract  
def verify\_signature(contract, signature\_b64, public\_key):  
 # Encode the contract into bytes  
 contract\_bytes = contract.encode("utf-8")  
  
 # Decode the Base64-encoded digital signature back to raw bytes  
 signature\_bytes = base64.b64decode(signature\_b64)  
  
 try:  
 # Use the sender's public key to verify the digital signature  
 public\_key.verify(  
 signature\_bytes, # The received digital signature  
 contract\_bytes, # The original contract  
 padding.PSS( # Use the same padding scheme  
 mgf=padding.MGF1(hashes.SHA3\_512()),  
 salt\_length=padding.PSS.MAX\_LENGTH  
 ),  
 hashes.SHA3\_512()  
 )  
 return "[+]Signature is VALID. Contract integrity verified."  
 except Exception as e:  
 # If verification fails, the content or signature may have been tampered with  
 return f"[!]Signature is INVALID. Possible tampering detected!\nError: {e}"  
  
  
# Perform the verification process  
verification\_result = verify\_signature(contract\_text, digital\_signature, buyer\_public\_key)  
print(verification\_result + "\n")  
  
# --------------------------------  
# Step 5: Display Output for Auditing / Logging  
# --------------------------------  
print("======== HRSCEP CONTRACT SIGNING LOG ========")  
  
# Display the base64-encoded digital signature (truncated for readability)  
print("[+]Digital Signature (Base64 Encoded):")  
print(digital\_signature) # Truncated for display  
  
# Display a preview of the contract contents  
print("---------------------------------------------")  
print("[+]Contract Summary (First 300 characters):")  
print("---------------------------------------------")  
print(contract\_text[:300] + "\n")  
print("===============================================")  
# Display the signature verification result  
print("[+]Verification Result:")  
print(verification\_result)  
print("=============================================\n")