White Paper for Wacom:
Cryptography in the STU-541 Tablet

Matthew Dodd

matthew@cryptocraft.co.uk

Cryptocraft Ltd.
Chapel Cottage
Broadchalke
Salisbury
Wiltshire
SP5 5EN

Issue 0.2

June 20, 2017

1. Introduction

This White Paper describes the new cryptographic security features designed into the STU-541 tablet and the benefits they bring. It is intended to be useful to anyone wanting to learn more about STU-541 cryptographic security, including those without a technical background, but it is also intended to provide enough information for security specialists to understand the robustness of the design and implementation of cryptography in the STU-541 unit.

But first, why is cryptography needed at all? The primary reason is to protect the tablet-host communication link. In brief, it addresses the following security objectives:

1. Information on the link should be confidential from unauthorised parties
2. Information on the link should be secure against modification by unauthorised parties
3. The host should be assured that it is communicating with an authorised tablet (and not, for example, with a man-in-the-middle)
4. The tablet should be assured that it is communicating with an authorised host (and not, for example, a man-in-the-middle)

In the following section we discuss why it might be important to achieve these objectives.

2. The Need for Cryptography: Possible Attack Scenarios

Wacom tablets may be used for signing important or confidential documents, so that data they handle may be sufficiently valuable to encourage malicious attack. Various attacks may be possible when the tablets are handled by users outside the acquiring organisation, or by a malicious insider.

One attack would be to replace the USB cable connecting tablet and host system with another cable which looks legitimate but contains active circuitry. This circuitry could record information on the wire, or even perform an active attack, perhaps replacing signature data with previously recorded signature data.

This demonstrates the need to protect the confidentiality of information on the USB link, and to protect against it being modified or replaced: security objectives 1 and 2 in section 1.

In another attack, a terminal could be replaced by a similar looking one intended to behave maliciously – perhaps to record transaction data.

Hence it is desirable that the host system has proof that it is connecting to a valid terminal, with a particular identity: this is security objective 3.

Finally, there may be situations in which organisations want to be assured that their tablets are only used for valid transactions with their system. For example, a malicious insider may connect a tablet to another system – his own laptop perhaps – in order to trick a customer into signing a modified document.

This final scenario is covered by security objective 4. Of the four objectives, this is the least likely to be important to an organisation, but may be desirable in some demanding applications.

3. Cryptography in STU-541

STU-541 tablets achieve the security objectives of section 1 by using an industry standard for cryptographic protection of point-to-point links, Transport Layer Security (TLS). Earlier versions of this protocol were called SSL. TLS is widely deployed, particularly for securing TCP links across the internet (for secure online shopping, for example).

We will describe TLS in general terms in sections 4 and 5 below, and give details specific to the STU-541 implementation in section 6 but we highlight now the benefits of adopting TLS:

- The protocol is fully standardised. The STU-541 uses the most recent published version of the standard, version 1.2 (version 1.3 still has draft status). TLS 1.2 is defined by the IETF in RFC 5246, and updated by subsequent standards.
- Being such a widely used standard, the protocol has been intensively studied by security experts, and problems resolved in newer versions (if configured properly). We discuss this further in section **Error! Reference source not found.** below.
- Open source implementations of the protocol are widely available and have also been subject to security evaluation. The STU-541 uses wolfSSL, as described in section 6.2 below.
- Each STU-541 is configured in the factory with a public key pair, which will be used with a default strong cipher suite. Again, we discuss this further below.
- Customers with special security requirements can choose to use their own choice of cryptographic algorithms and parameters used to protect the link.

4. An Overview of Public Key Cryptography

First, let's recall some concepts behind public key cryptography, which underpins how TLS works.

The idea behind public key encryption/decryption is that it's possible to construct, together, a complementary pair of encryption and decryption mappings which are the inverse of one another and can each be efficiently computed on any input, but so that, if just the encryption mapping is revealed publically, it's hard to work out how to perform the corresponding decryption efficiently. The point is that this gives a way for a party to allow another to send them an encrypted message: he (or she) generates a complementary pair of methods for encrypting and decrypting, and publishes the encrypting algorithm so that others can send him confidential messages. The first published system of this kind was RSA, named with the initials of its inventors.

A related concept is that of digital signatures. Here a party generates a complementary pair of algorithms, now for creating and verifying signatures on messages. In this case the party publishes a method for verifying the signature on any message, but publishing this doesn't reveal an efficient method for creating a valid signature on any message. Thus the first party, and no other, can sign messages which can be verified by anyone else.

A problem with the technique of public key encryption we've described so far is that the party doing the encrypting needs to be confident that he's using the intended recipient's

encryption algorithm. Without this confidence he might be subject to a man-in the-middle (MITM) attack – the MITM pretends to each party that he's the other, and in this way is able to see the unencrypted message.

PKI (public key infrastructure) is a solution to this problem in which a new party is introduced, the certification authority (CA), who is trusted by sender and receiver. The CA can now sign a message containing both a party's public key and a text string that clearly identifies him – the public key, associated text string and CA's signature together form a public key *certificate*. Using a public key certificate removes the problems we mentioned above with public key encryption without certificates – now the party doing the encrypting can be confident he's using the public key of the intended recipient.

We should add here that there is another type of public key system which allows two parties to agree on a common secret not known to anyone monitoring their communications. In this case, each party computes two related quantities, one secret and one public. Each retains his secret, but transmits the corresponding public quantity to the other party – and again there is a problem here about being confident about the provenance of a received public quantity. The scheme then allows each party to make a calculation with their secret and the other party's public value, in such a way that both parties arrive at a common value known only to them. The most widely used scheme of this type is Diffie-Hellman key exchange, and it can be used of a variety of algebraic system including ones based on elliptic curve calculations.

5. An Overview of TLS

TLS uses certificated public key cryptography to encrypt (or otherwise agree) a short random or random-like message. This short message, now known only to the two communicating parties, is then used to generate key material for much faster encryption and authentication algorithms of a mutually agreed cipher suite which are used to protect the communication session.

5.1 A Brief Description of TLS

In this section we provide a little more detail about how the TLS protocol (Dierks and Rescorla 2014) works.

The STU-541 takes the role of TLS 'server', and the host connects to it as 'client'. TLS traffic is exchanged between client and server in records, each made up of a record type, protocol version, record length and record payload.

At the start of a new secure session, a key agreement handshake takes place, with the exchange of handshake messages, each sent in a record of type 'handshake'. The handshake protocol acts to establish all the necessary parameters for encrypting and authenticating data to be exchanged.

A full key agreement handshake works as follows. First the host sends the tablet a **Client Hello** message, containing: the version of TLS the client is using, a 'client random' value consisting of the current time (coded as 4 bytes) together with a random value (28 bytes)

produced by the client, a string used to identify this TLS session, a list of cipher suites the client supports in decreasing order of preference, a list of compression algorithms the client supports in order of preference, and optionally some further capability and options data in an extension field. Notice that a **cipher suite** defines three things:

- the method to be used for key exchange
- the data encryption algorithm; and
- the integrity protection algorithm.

The tablet will respond to this with a **Server Hello** message, containing: the version of TLS to be used for this session, a 'server random' of similar format to the 'client random', and the cipher suite, compression algorithm and (optionally) extensions to be used for the session. The tablet follows this with a **Server Certificate** message, containing its signed certificate. Next, it sends a **Server Key Exchange** message whenever the key agreement algorithm requires a message from the tablet. If customer certificates are being used and require the host to authenticate itself to the tablet, the tablet will now send a **Certificate Request** message. Finally the tablet will send a **Server Hello Done** message, to indicate that this group of messages from the server is complete. Now the host, as TLS client, responds. It sends a **Client Certificate** if this has been requested. Then it always sends a **Client Key Exchange** message, containing data that will allow key agreement to take place – the agreed key is called the premaster secret. Finally, if client certificates are being used, it sends a **Certificate Verify** message, containing a signature on the handshake data so far, in order to convince the tablet that the client knows the private key corresponding to the public key in the certificate, and hence is its legitimate holder.

From the premaster secret and the time/random information they each sent, tablet and host apply a one-way calculation to compute a master secret, and then delete their copies of the premaster secret. And from the master secret, both tablet and host compute two encryption keys, for encrypting the two directions of communication, two authentication keys, for integrity protection of each packet of encrypted data sent in the two directions of communication, and two implicit nonce values used in some authentication schemes. All these parameters become active for one direction of communication when the transmitting party sends a **ChangeCipherSpec** message, in a record of type 'change_cipher_spec'.

The first record sent by a transmitting party after a ChangeCipherSpec message is a **Finished** message (in a record of type 'handshake'). This contains the result of applying the hash function agreed in the handshake to compute a MAC (Message Authentication Code – a form of cryptographic checksum) of all the handshake data using the master secret as key. This gives a check that the master secret has been agreed correctly and the handshake data had not been altered by a man-in-the-middle. Furthermore, a Finished message is encrypted and authenticated under the newly established algorithms and keys.

Once a Finished message has been sent, application data can follow, which will also be encrypted and authenticated under the new algorithms and keys. Protected application data is sent in records of type 'application_data'.

5.2 Benefits of TLS Revisited

We have already highlighted the benefits of using TLS in section 3, but revisit this topic in the light of our discussion about TLS.

We've seen now that TLS uses public key cryptography to

- authenticate the tablet to the host, that is, to prove to the host that the tablet has a key pair associated with its serial number authenticate the tablet to the host
- optionally to authenticate host to tablet
- to agree a secret which use used to provide link encryption and authentication.

Moreover the TLS handshake allows the algorithms used to be negotiated. This means that the STU-541 can offer TLS protection as standard, with carefully chosen strong algorithms. But then customers with specific security needs can customise this process to suit their

requirement. We discuss the details of how this can be done in section 6.

6. The Implementation of TLS in the STU-541

In this section we give details of the use of TLS in the STU-541. We describe the security features present in each STU-541 tablet, and describe how it is implemented in order that customers can assess the robustness of this implementation. We also describe how organisations with special security demands can customise the protection according to their requirements.

6.1 Default Protection

A STU-541 tablet protects the USB link using TLS 1.2. A client must connect using TLS 1.2 in its ClientHello message – fall back to earlier versions of TLS is not permitted.

Each STU-541 tablet is shipped with its own factory default certificated key pair for the cipher suite TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256.

The public key passed in the Server Certificate message is the RSA public key used to verify the RSA signature. The Common Name (CN) field of this public key is the electronic serial number of that particular tablet, and Wacom has signed the public key via a certificate chain ending in a standard Cybertrust root key.

6.2 TLS Implementation in the STU-541

As well as the advantages of using a standard security protocol, TLS, that we have already discussed, the STU-541 also uses an open source implementation of TLS, wolfSSL. This can be freely downloaded from the wolfSSL website (wolfSSL Inc. 2016), and so can documentation about the implementation and its use (wolfSSL Inc. 2016).

The advantage of using an open source implementation is that it has been subject to wide review, reducing the possibility of serious implementation error. Evidence of the care taken

over the implementation, and its scrutiny, is a FIPS 140-2 level 1 accreditation of a version of the codebase (NIST 2016).

6.3 Assessment of the Security of TLS on the STU-541

Prior to the STU-541, Wacom tablets incorporated a proprietary scheme for security on the USB link, encrypting some data passing over this link. This scheme used 2048-bit RSA for key exchange, and 256-bit AES for data encryption. Note that, according to table 7.3 of the Ecrypt II report (ECRYPT 2012), 2048-bit RSA is equivalent in strength to a symmetric system of key size about 103 bits. So in fact there is no real benefit in using 256-bit AES rather than 128-bit AES in this system.

By comparison, in the STU-541, all data on the USB link is both encrypted and authenticated, and public key cryptography is used to certify the public-key key agreement protocol, in order to provide proper protection against man-in-the-middle attacks. With the factory default key pair, elliptic curve cryptography is used for key agreement and 128-bit AES in GCM mode is used for data encryption and authentication. As we mentioned in section 6.1, factory default protection in the STU-541 system provides perfect forward secrecy.

7. Summary

The STU-541 uses the industry-standard TLS cryptographic protocol to provide confidentiality, integrity and authentication on its USB link. Each unit is configured in factory to provide key exchange, encryption and authentication and the 128-bit level of security, authenticated by 2048-bit RSA, and using elliptic curve cryptography for secure and efficient key agreement. Care has been taken in the implementation and default factory settings to ensure that published attacks on TLS don't apply to this configuration. The STU-541 incorporates a respected open source implementation, configured to take advantage of hardware features for random bit generation and hardware acceleration of authenticated encryption. Customers with special requirements can use the flexibility of TLS to tailor the factory setting to their own requirements, and optionally to use certificates to authenticate host connections to the tablet.

8. References

Barker, Elaine B., and John M. Kelsey. 2015. 'Recommendation for Random Number Generation Using Deterministic Random Bit Generators'. NIST SP 800-90Ar1. National Institute of Standards and Technology. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf.

Certicom Research. 2000. 'SEC 2: Recommended Elliptic Curve Domain Parameters, Version 1.0'. http://www.secg.org/SEC2-Ver-1.0.pdf.

Dierks, T., and E. Rescorla. 2014. 'RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2'. Accessed July 15. http://tools.ietf.org/html/rfc5246.

ECRYPT. 2012. 'ECRYPT II Yearly Report on Algorithms and Keysizes (2011-2012)'. http://www.ecrypt.eu.org/documents/D.SPA.20.pdf.

Internet Engineering Task Force (IETF). 2016. ‘RFC 7457 - Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS)’. Accessed October 18. https://tools.ietf.org/html/rfc7457.

NIST. 2016. ‘Validated FIPS 140-1 and FIPS 140-2 Cryptographic Modules’. Accessed October 17. http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm#2425.

Rukhin, Andrew, Juan Soto, James Nechvatal, Elaine Barker, Stefan Leigh, Mark Levenson, David Banks, Alan Heckert, James Dray, and San Vo. 2010. ‘NIST Special Publication 800-22 Revision 1a: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications’. National Institute of Standards and Technology. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf.

wolfSSL Inc. 2016. ‘wolfSSL User Manual’. https://www.wolfssl.com/documentation/wolfSSL-Manual.pdf.

———. 2016. ‘wolfSSL - Embedded SSL Library for Applications, Devices, IoT, and the Cloud’. Accessed October 14. https://www.wolfssl.com.