Qwyit™
Perfect Security

TECHNOLOGY PRESENTATION

Today's Security Overview

"**It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong.**" – Richard P. Feynman

- Expert security methods don't agree with real world implementation
  - Terrifying Cybercrime Statistics in 2021: $**1.5*Trillion*/yr in losses**
- Only One Conclusion [whether the methods *aren't* used (don't fit, too complex, too slow, insecure, etc.) or they fail *when* used]:

Digital security needs *new methods*!

Today's Security - Examples

- 2017 Equifax data breach – 140+ *million* people attacked *all at the same time, in a single crime*
- 2020 US Government data breach – Millions of computers hacked, *with a single unauthorized code insertion*

- Resulting in 'The Dr. Seuss Security Conundrum'
  - "Sometimes the questions are complex (how to secure today's networks), but the answer is simple":

We need *Perfect Security methods*

Introducing ***Perfect Security***

- ***Perfect Security*** is real world privacy of every bit in all digital landscapes for all participants at all times
  - Cryptography: ***Perfect Security*** is a special case of information-theoretic security: unbreakable, never computationally vulnerable

- ***Perfect Security*** is a singular, complete solution that works everywhere, in everything, all the time

***No current Perfect Security* implementations**

Perfect Security – Required Capabilities

- Cryptography has a *Perfect Secrecy* encryption method (Shannon defined/proven)
- That method has an *Ideal System* definition (Shannon specified)
- A proper *Ideal System* implementation delivers ***Perfect Security***
  - **Mutual, Continuous Authentication** – Every system event, for all participants, all the time
  - **Perfect Secrecy encryption** – Every system bit, for all participants, all the time

Perfect Security: Every event authentic, every bit perfectly secret

Perfect Security – Required Properties

- A universal ***Perfect Security*** method where every Event and Bit are mutually, continuously authentic and perfectly encrypted, *must be*
  - **FAST** – Operate within protocol latency; storage processing, transmission benchmarks, in any and every OSI layer: ***Perform everywhere***
  - **EFFICIENT** – Embed in every H/W device, S/W control, etc.; no performance and bandwidth degradation: ***Fit in everything***
  - **FLEXIBLE** – Work in any type of implementation, network, trust model: ***On all the time***
  - **FUTURE SAFE** – not computationally bound: ***No cryptanalytic solution***

Perfect Security *is the only solution that solves real world implementation cybercrime – and* ***Qwyit Built It***

[Review the FPGA Demo]

Introducing Qwyit® - Security Engineering

- Qwyit™ Authentication and Data Security protocol – the only ***Perfect Security*** implementation

- *Mutual, continuous authentication of every system event*
  - QwyitKey™ – Trusted 3rd Party token creation/mgmt/distribution
- *Provably secure, unique-key encryption of every system bit*
  - QCy™ cipher – Shannon's Perfect Secrecy Finite Key Ideal System

QwyitKey™ - Replace PKI

QwyitKey™ Registration - Anonymous Set Up (QAASU)

QwyitKey Client

Initial operation: Make QAASU Request (online at qwyit.com, etc.)

Step 5 (ASUC5) Check public AID=created AID

Step 1 Client Request (ASUC1) →

← *Step 2 QAAS Reply (ASUQ2)*

Step 3 Confirmation (ASUC3) →

QwyitKey Authentication Service/Server (QAAS)

Upon ASU request receipt, generates Qwyit credentials (token)

Step 4 (ASUQ4) Decrypt ASUC3

Authentication Request

Client (Sender)
Auth Request

Step 4 Decrypt ARQ3 (ARC4)
Ready for secure msgs

Step 1 Client Request (ARC1)
Request for recipient-only secure auth bundle →

← *Step 3 QAAS Reply (ARQ3)*
Reply w/bundle

QwyitKey Authentication Service/Server QAAS

Step 2 Decrypt and create Authentication Bundle (ARQ2)
Decrypt the request, create Auth bundle

Purchasing an SSL/TLS Certificate From a Certificate Authority

If you want to secure your website with an SSL/TLS certificate, you can purchase one from a Certificate Authority. You can also purchase one directly through Plesk, as well as protect your website with a free self-signed SSL/TLS certificate, or with a certificate you already own.

To purchase a certificate, you need to generate a Certificate Signing Request (CSR for short) first. Go to Websites & Domains and click SSL/TLS Certificates > Add SSL/TLS Certificate. Fill out the fields marked with the red asterisk symbol (*), such as the certificate name (you will use it to identify the certificate in the list of all certificates), your personal information, the name of the domain the certificate will be protecting, and so on. Make sure that all information you provide is correct, as mistakes may make the certificate unsuitable and make it necessary to pay for a new one.

Note: If you want to purchase a wildcard SSL/TLS certificate, your domain name must start with an asterisk symbol (*). For example, a certificate generated for *.example.com can be used to secure any subdomain of example.com.

When you have finished, click Request. This will result in the CSR and the private key being generated and placed in your repository.

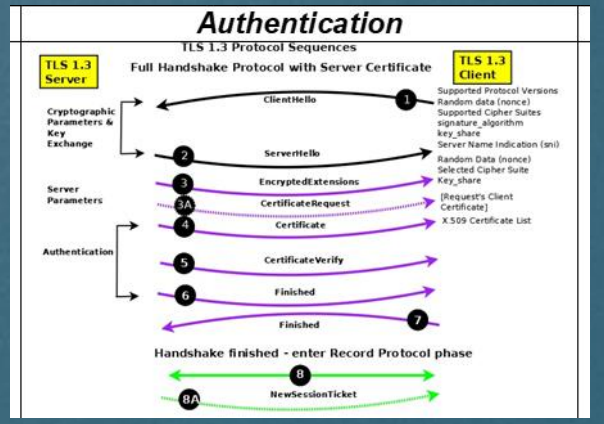
Now that the CSR has been generated, you need to provide it to the Certificate Authority of your choice to purchase a certificate from them. To retrieve the CSR, go to Websites & Domains > SSL/TLS Certificates and click the name of the certificate you have just generated. Scroll down to the CSR section and copy the text starting with -----BEGIN CERTIFICATE REQUEST----- and ending with -----END CERTIFICATE REQUEST----- (including those lines with all the dashes) to the clipboard. You will need to provide the CSR to the Certification Authority when purchasing your certificate. The exact procedure differs from one Certificate Authority to another, so contact the Certificate Authority for assistance should you encounter any difficulties. Once you complete the purchase, you will be given the certificate in the form of either a *.crt file, a *.pem file, or in text form.

To secure your website with the certificate you have purchased, you need to upload it first. Go to Websites & Domains > SSL/TLS Certificates and click the name of the certificate, then upload it as described below:

If you received the certificate in the form of a *.crt or a *.pem file, scroll down to the Upload the certificate files section and upload the file. Click Upload Certificate when you have finished.
If you received the certificate as text, scroll down to the Upload the certificate as text section and paste the certificate into the corresponding field. Click Upload Certificate when you have finished.
This will result in the certificate being placed in your repository. You can see a list of all SSL/TLS certificates in your repository by going to Websites & Domains > SSL/TLS Certificates.

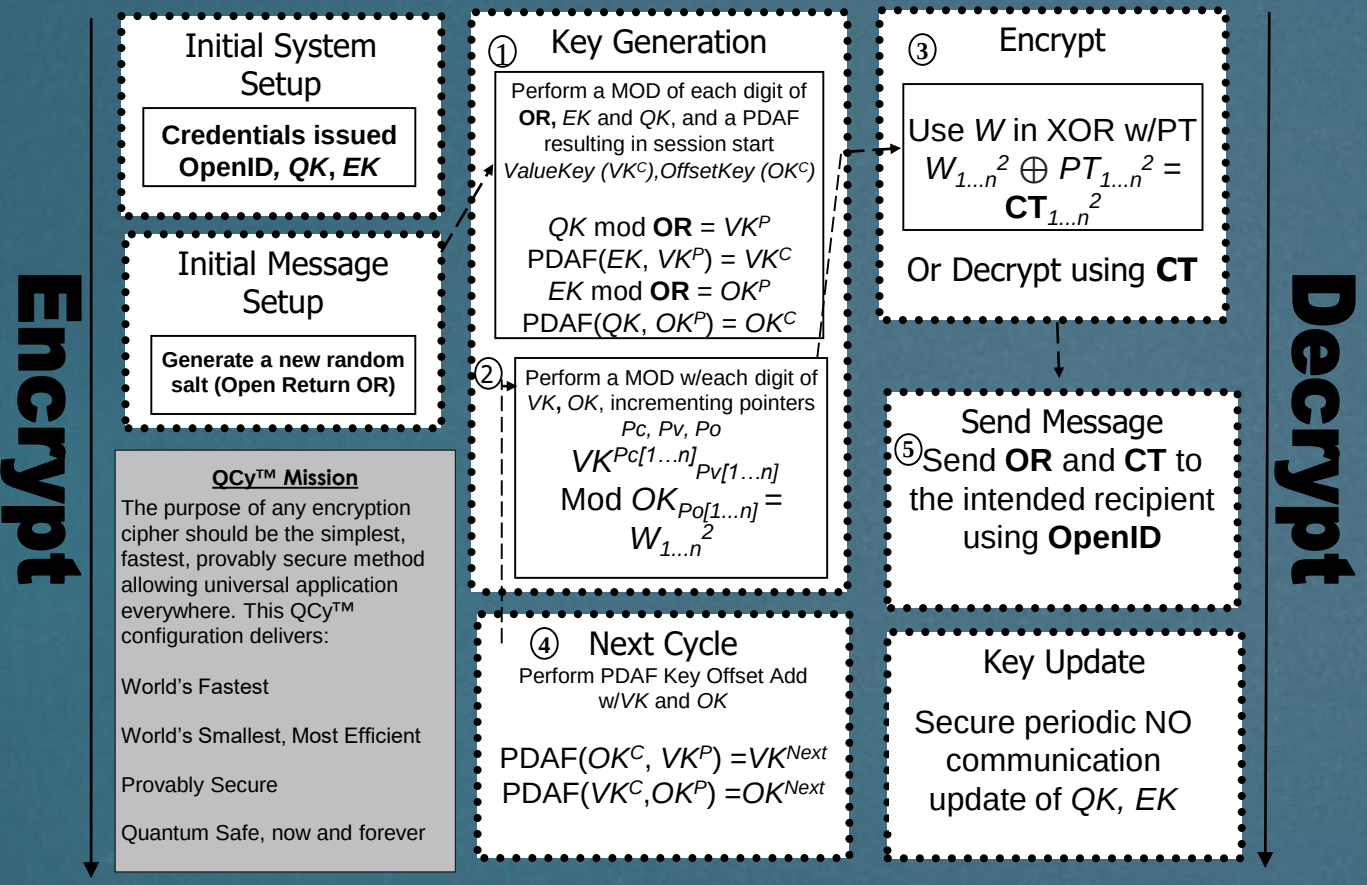
Now that the certificate has been uploaded, you need to install it. Go to Websites & Domains and click Hosting Settings. Select the SSL/TLS support checkbox, select the certificate you have just uploaded from the Certificate menu, then click OK.

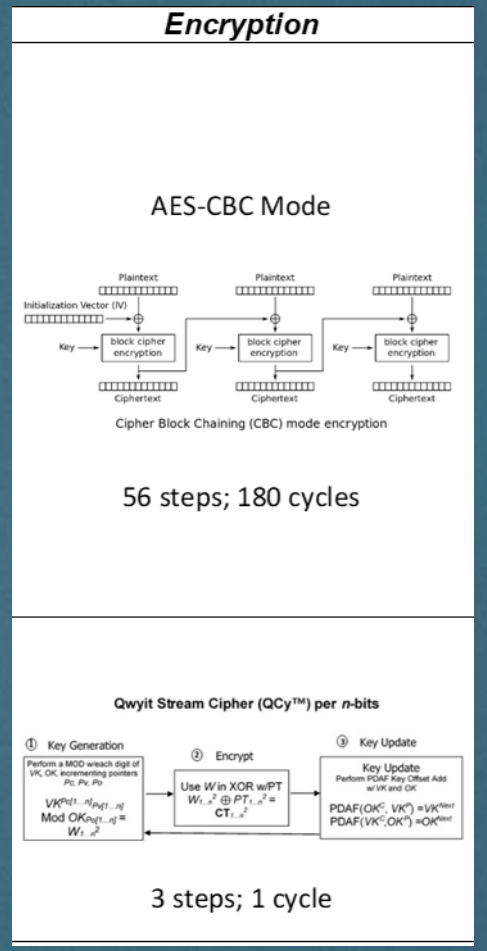
PKI – TLS 1.3 Handshake



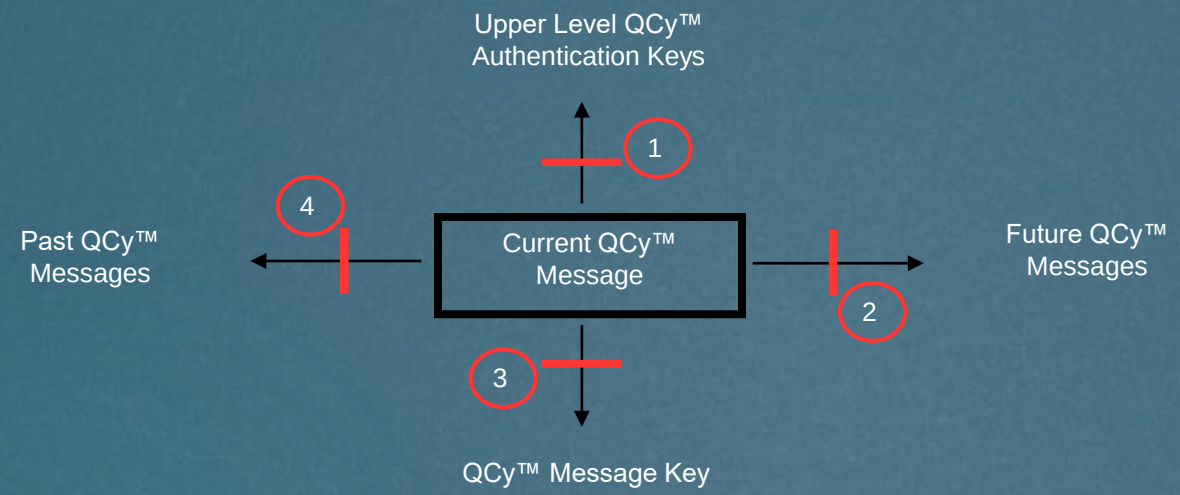
QCy™ - Replace AES

Qwyit Stream Cipher and Key Exchange/Update (QCy™)





QCy™ Perfect Security Cross



System Assumptions:

- Upper Level QCy™ Authentication Keys are securely pre-shared
  - By participant-managed, independent trust QwyitKey™ system, or other
- QCy™ Auth Keys create unique, new message keys for every message

Perfect Security IF:

1. Broken Current Message does NOT reveal Authentication Keys (One-way math gate)
2. Broken Current Message does NOT reveal Future Messages (One-way math gate)
3. Current Message is Provably Secure (Math proved, Shannon Secure)
   - Known Plaintext reveals key, but 1, 2 and 4 still hold – as does any remaining msg
4. Broken Current Message does NOT reveal Past Messages (One-way math gate)

QCy ™ is Perfect Cross Secure because the PDAF stops all attacks in all directions

QCy™ is Perfect Cross Secure:

1 – Continuously defeated by the underdetermined, irreversible PDAF Offset Key Add mode

2 – Even w/previous message knowledge, including *all* values except the Master Keys, new message breaks are defeated by the PDAF/OR reseed

3 – With only some PT knowledge and corresponding *W* section, the *VK/OK* keys all remain underdetermined for other sections, and maintain message key integrity

4 – Same as 2, PDAF/OR reseed

Qwyit™ ***Perfect Security*** is Ready Now

Commercially Ready

- Protocols – Complete Authentication and Encryption
- Validation – Several independent Expert reviews
- Prototype/Demo – FPGAs (Arria V and Achronix ACE)
- Complete Universal development kits and capability
  - H/W (QwyitChip™), S/W (QwyitSDK™), QwyitKey™ key token management platform

Market Applications

- IoT, AI, autonomous cars, mobile, storage, health, Gov, media, etc.

Perfect Security Summary

- ***Perfect Security*** exists when every system event is authentic and every content bit is perfectly secret
- **Mutual, Continuous Authentication** and **Perfect Secrecy encryption** accomplishes ***Perfect Cross Security***, eliminating all cybercrime
- Qwyit® has designed, published, validated and demonstrated the Qwyit™ protocol, using QwyitKey™ authentication and QCy™ encryption to ***deliver Perfect Security***
- ***Fast, Efficient, Flexible and Future Safe*** are proven in our demonstrated prototype benchmarks

Let's proliferate Qwyit™ Everywhere, in Everything, On All the Time

Further Information

Contact Qwyit LLC

Paul McGough, Founder and CTO
PaulM@Qwyit.com
www.qwyit.com info@qwyit.com

All truth passes through three stages. First, it is ridiculed. Second, it is violently opposed. Third, it is accepted as being self-evident. - Arthur Schopenhaue