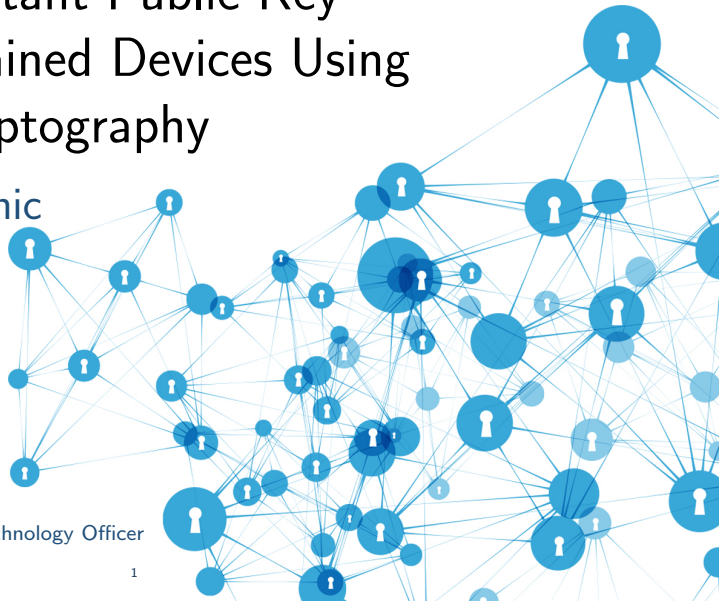
Fast, Quantum-Resistant Public-Key Solutions for Constrained Devices Using Group Theoretic Cryptography

International Cryptographic Module Conference

May 18, 2017



SECURE RF
Securing the Internet of Things®

Derek Atkins, Chief Technology Officer

Real World Problems

How do you secure devices that have minimal processing power or energy?

- AES? Requires explicit key management

  Solution: use a Public Key Algorithm
- RSA, ECC, and DH? all require too much time/CPU/power
- Software implementations are too slow
- Hardware implementations require too much silicon

Need a Lightweight Solution

Enter Group Theoretic Cryptography

SECURE RF
Securing the Internet of Things®

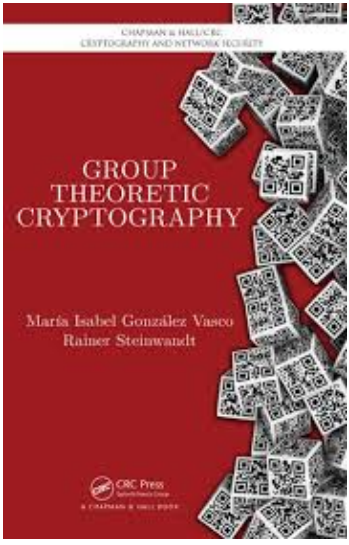
So What Is Group Theoretic Cryptography?

Talk Outline

- Enter Group Theoretic Cryptography
- Math Primer and E-Multiplication
- Ironwood Key Agreement Protocol
- Walnut Digital Signature Algorithm
- Conclusions

Enter Group Theoretic Cryptography (GTC)

- Hard problems have been studied over 100 years
- GTC dates to the 1970s
- Computational complexity scales linearly with security (instead of quadratically like RSA, ECC, and DH)



SECURE RF
Securing the Internet of Things®

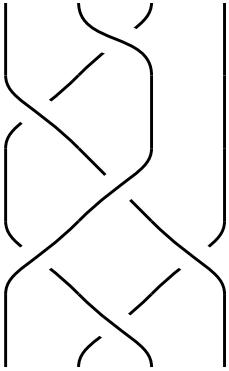
GTC Deconstructed

- Every public-key method is based on several math foundations; GTC is no different.
- GTC leverages structured groups, matrices, permutations, and arithmetic over finite fields.
- The structured group used for GTC is the Braid Group.
- Note that there have been other uses of the Braid Group for cryptography (some of which have been broken); GTC is different than those.
- GTC is not "Braid Group Cryptography"

The Artin Braid Group

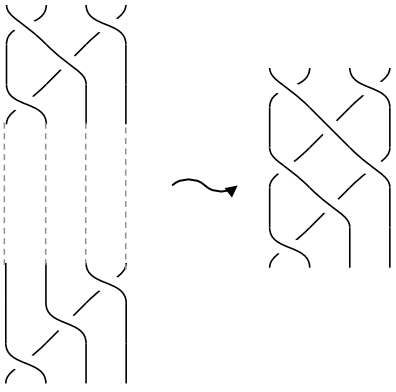
Braids

Introduced by Emil Artin in 1921, a braid is a configuration of strands of the form:



Multiplying Braids

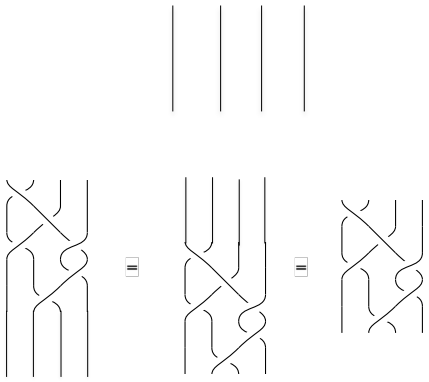
Two braids can be concatenated to yield a third braid:



SECURE RF
Securing the Internet of Things®

Identity Element

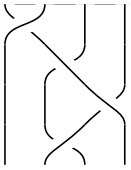
The braid without any crossings functions as an identity element:

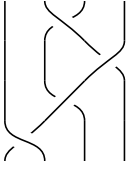


SECURE RF
Securing the Internet of Things®

Inverses

The inverse of a braid can be found by inverting the crossings and multiplying in the reverse order: the inverse of the braid β

$\beta =$ 

$\beta^{-1} =$ 

SECURE RF
Securing the Internet of Things®

Braid Generators

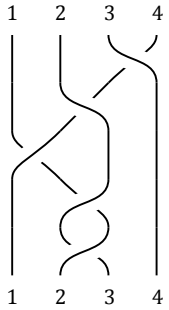
Every braid β on N strands can be viewed as a product of a sequence of braids with a single crossing, denoted $b_1, \ldots, b_{N-1}$, and their inverses. For example, when $N = 4$ we have

$b_1 =$  $b_2 =$  $b_3 =$ 

$b_1^{-1} =$  $b_2^{-1} =$  $b_3^{-1} =$ 

Associated Permutations

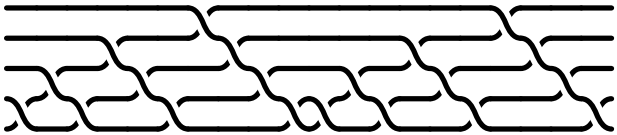
Every braid β has a natural permutation σ_β associated with it: for $i = 1, 2, \ldots, N$ the value of $\sigma_\beta(i)$ is the endpoint of the strand originating at the point i. Below, $1 \mapsto 2,\ 2 \mapsto 3,\ 3 \mapsto 4,\ 4 \mapsto 1$.



Pure Braids

Braids with trivial permutations

You can find braids that are not identity braids but still have a trivial permutation:



Note how each strand ends in the same position where it started.

Colored Burau Representation of B_N

Each $b_i^{\pm 1}$ is associated with the ordered pair $\left(CB(b_i)^{\pm 1}, \sigma_i\right)$ where σ_i is the transposition $(i, i+1)$,

$$CB(b_i) = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & t_i & -t_i & 1 & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}, CB(b_i^{-1}) = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & 1 & -\frac{1}{t_{i+1}} & \frac{1}{t_{i+1}} & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}.$$

By letting permutations act on the left on the matrices $CB(b_i)$ (by permuting the variable entries), the ordered pairs $(CB(b_i)^{\pm 1}, \sigma_i)$ form a semi-direct product which satisfy the braid relations. This gives a representation of B_N.

E-Multiplication (denoted $\star$)

A one-way function, a building block for all SecureRF group-theoretic cryptographic constructions.

E-Multiplication System Data

- B_N = the braid group on N strands
- F_q = a finite field of q elements.
- A set of T-values $\{\tau_1, \tau_2, \ldots, \tau_N\} \subset F_q^{\times}$.
- $m \in GL(n, F_q), \sigma \in S_N$.

E-Multiplication Continued

E-Multiplication by one Artin generator

$$(m,\sigma)\star\Big(CB(b_i),\ (i,i+1)\Big) = \left(m\cdot\begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & \tau_{\sigma(i)} & -\tau_{\sigma(i)} & 1 & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix},\ \sigma\cdot(i,i+1)\right)$$

By iterating this computation we can compute the E-Multiplication of (m,σ) with an arbitrary braid element (finite product of Artin generators and their inverses).

E-Multiplication Example

Assume we are working in B_4F_7 with T-values 2 4 6 3. To start we have a matrix and permutation:

$$\begin{pmatrix} 1 & 4 & 3 & 5 \\ 2 & 5 & 2 & 6 \\ 3 & 6 & 2 & 1 \\ 2 & 4 & 2 & 5 \end{pmatrix}, (2, 4, 3, 1)$$

Next we want to apply b_2. We apply the current permutation and T-values to the CB matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ t_2 & -t_2 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 \\ 3 & 4 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Then we matrix-multiply with the finite field $\mathbb{F}_7$ and apply the braid twist to wind up with the following matrix and permutation:

$$\begin{pmatrix} 6 & 2 & 0 & 5 \\ 3 & 6 & 0 & 6 \\ 0 & 3 & 1 & 1 \\ 0 & 2 & 6 & 5 \end{pmatrix}, (2, 3, 4, 1)$$

SECURE RF
Securing the Internet of Things®

Laurent Polynomial Entry

Short random word of length 10 in $B_4 \rtimes S_4$ – What E-Multiplication erases!

$$
-\frac{1}{t[3]}\left(1-t[1]+t[1]\,t[2]-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}+\right.
$$

$$
\frac{1}{t[1]}
$$

$$
\left(\frac{1}{t[2]}\right.
$$

$$
\left(-t[1]\left(1-t[1]+t[1]\,t[2]-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}\right)+\right.
$$

$$
t[1]\left(1-t[1]+t[1]\,t[2]-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}+\right.
$$

$$
t[3]\left(-\frac{(1-t[1])\,t[2]}{t[3]}+\frac{-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}}{t[4]}\right)-
$$

$$
\left.\left.\left.\frac{-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}}{t[4]}\right)\right)-\right.
$$

$$
t[1]\left(1-t[1]+t[1]\,t[2]-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}+\right.
$$

$$
t[3]\left(-\frac{(1-t[1])\,t[2]}{t[3]}+\frac{-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}}{t[4]}\right)-
$$

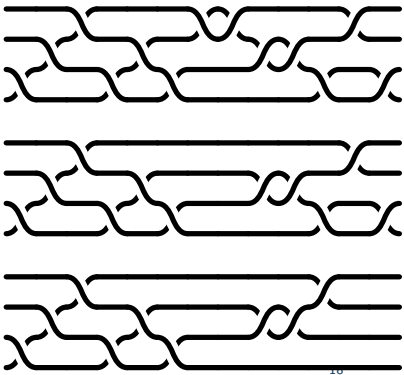
$$
\left.\left.\left.\frac{-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}}{t[4]}\right)\right)-\frac{-\frac{1-t[1]}{t[3]}+\frac{(1-t[1])\,t[2]}{t[3]}}{t[4]}\right),
$$

SECURE RF
Securing the Internet of Things®

Shortest Word Problem

The (true) length of a braid is the minimal number of crossings needed to represent it. Finding this is a known hard (NP-Hard!) problem.

A given braid has many different representations in terms of crossings, because we consider two braids to be the same if one can be manipulated into the other without moving the ends or cutting.



Conjugacy Search Problem

The *conjugacy search problem* (CSP):

- Assume a braid of the form $w\ a\ w^{-1}$ (aka a conjugate) where a is known
- The Problem: find w.

Unfortunately, this is not a hard problem in many cases. There are algorithms that exist to solve this problem efficiently.

The good news is that none of these algorithms depend on the CSP.

Group Theoretic Cryptographic Constructions

Many methods. All based on E-Multiplication:

- Ironwood™ Key Agreement Protocol
  Enables two endpoints to generate a shared secret over an open channel.
- Walnut™ Digital Signature Algorithm
  Allows one device to generate a document that is verified by another. Very fast verification
- Kayawood™ Key Agreement Protocol
- Hickory™ Hash (Cryptographic Hash Function)

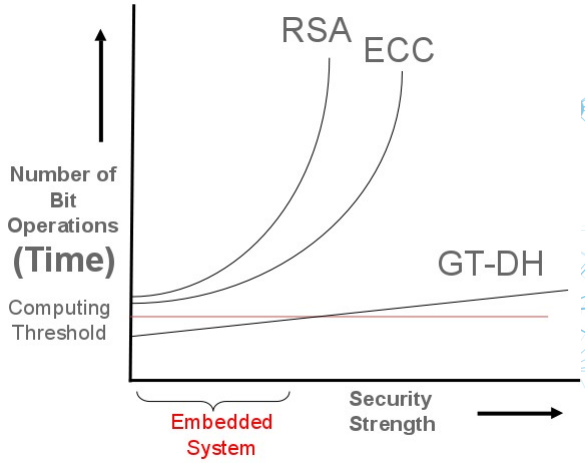
Today we will focus on Ironwood KAP and WalnutDSA

SECURE RF
Securing the Internet of Things®

Group Theoretic Cryptography

Breakthrough Performance over Number Theoretic Cryptography

- Diffie-Hellman type Key Agreement
- DSA-like Digital Signature
- Based on Infinite Groups
- "Linear-in-Time" Security Strength
- Safe against known Quantum Attacks



SECURE RF
Securing the Internet of Things®

The Ironwood^{TM} Key Agreement Protocol

Public (system wide) Information:

- B_N (The braid group in N strands)
- F_q (A finite field with q elements)
- A matrix $m_0 \in GL(N, F_q)$

Private Information (created by TTP):

- A set of T-values $\{\tau_1, \tau_2, \ldots, \tau_N\} \subset F_q^{\times}$.
- Two sets of rewritten commuting conjugates:

$$A = \left\{ z\alpha_1 z^{-1},\ z\alpha_2 z^{-1},\quad \ldots,\quad z\alpha_k z^{-1} \right\},$$
$$B = \left\{ z\beta_1 z^{-1},\ z\beta_2 z^{-1},\quad \ldots,\quad z\beta_k z^{-1} \right\}.$$

where some α_i are purebraids, and z is destroyed after use.

Generating Home Device (HD) Keys

Home Device private/public key pair creation

Step 1: Generate two non-singular matrices:

$$C = \sum_{k=0}^{N-1} c_k m_0^k, \qquad C' = \sum_{k=0}^{N-1} c'_k m_0^k, \qquad (\text{with } c_k, c'_k \in \mathbf{F}_q).$$

Step 2: Generate two braids, β, β' with the same permutation σ.

Step 3: Using the T-values, compute the following two E-Multiplications:

$$(C, \mathrm{Id}) \star (\beta, \sigma) := (CM, \sigma), \qquad (C', \mathrm{Id}) \star (\beta', \sigma) := (C'M', \sigma).$$

Step 4: Compute the HD public key:

$$Pub_{HD} = (CM(C'M')^{-1})^{-1} = C'M'M^{-1}C^{-1}$$

HD Keys: What Does This Mean?

- C and C' are private matrices
  Fixed Size (based on N and q)
  Need to be kept secure
- β and β' are private braids
  Variable Size
  Also need to be kept secure
- The public key (Pub_{HD}) is a also matrix, which gets shared with other devices.
  It can be encoded into a certificate for identity verification.
- T-values are also stored on the device and must remain secure.

SECURE RF
Securing the Internet of Things®

Generating the Other Device (OD) Keys (Tokens)

Other Device private/public data creation

Step 1: Generate a non-singular matrix:

$$C_i = \sum_{k=0}^{N-1} c_{k,i} m_0^k, \qquad \left(\text{with } c_{k,i} \in \mathbf{F}_q\right),$$

Step 2: Generate a random braidword β_i using the Other Device conjugates.

Step 3: Compute the public key: $\text{Pub}_i := (C_i, \text{Id}) \star (\beta_i, \sigma_i) = (M_i, \sigma_i)$, where *Id* is the identity permutation.

Note that this operation must occur on the TTP, and the resulting data must be securely provisioned onto the device. Also note that this device does not need access to the T-values.

OD Keys: What Does This Mean?

Data stored on e.g. a smartphone

- C_i is a private matrix
  Fixed Size (based on N and q)
  Needs to be kept secure
  (Same size at the matrix on the HD)
- The public key (Pub_i) is a matrix plus permutation, which gets shared.
  It is signed by the CA.

Ironwood KAP Shared Secret

After keys are generated, the Ironwood KAP proceeds as follows:

Step 1: Devices exchange public keys (Pub_i and Pub_{HD})

Step 2: Home Device computes the following two E-Multiplications:

$$(CM_i, \sigma_i) \star (\beta, \sigma) := (Y,\ \sigma_i\sigma), \qquad (C'M_i, \sigma_i) \star (\beta', \sigma) := (Y',\ \sigma_i\sigma).$$

Step 3: Home Device computes the column vectors:

$s = (N/2)^{\text{th}}$ column of the matrix Y, $\qquad s' = (N/2)^{\text{th}}$ column of the matrix Y'.

Step 4: Home Device sends s to the other device.

Step 5: Other Device computes the matrix and vector multiplications:

$$s' = C_i \left(C'M'M^{-1}C^{-1}\right) C_i^{-1} \cdot s = C_i Pub_{HD} C_i^{-1},$$

where s' is the shared secret

SECURE RF
Securing the Internet of Things®

Shared Secret: What Does That Mean?

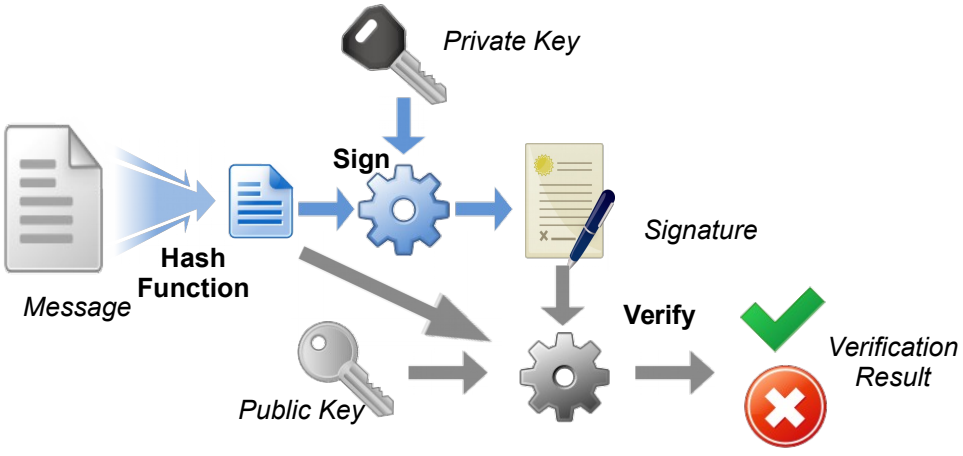
- The devices share their public keys (Pub_i and Pub_{HD})
- Certificates are validated
- The HD computes two column vectors, s and s'
- It sends s to the other device; s' is the shared secret
- The other device computes the shared secret (s') using Pub_{HD}, C_i, and s

Walnut Digital Signature Algorithm™

- Quantum-resistant public-key digital signatures
- Based on SecureRF's Group Theoretic Cryptography
- Leverages *E*-Multiplication for extremely fast verification
- Security is based on the hard problems of solving a novel equation over the braid group and reversing *E*-Multiplication

SECURE RF
Securing the Internet of Things®

WalnutDSA™ Architecture



Generating WalnutDSA Keys

- Choose BnFq (e.g. B8F32 for 2^{128} security level (sl))
- Generate Random T-values (not 0 or 1)
- Choose exactly two (2) T-values (positions a,b) as 1
- Choose $Priv_S$ as a random braid of length at least $n\ sl/\left(4\log_2((n-1)(n-2))\right)$
- Compute and Distribute Public Key (Pub_S) as:
  - T-values
  - E-Multiplication result: $(1,1) \star Priv_S$

WalnutDSA Signature Generation

- Take a 256-bit message (e.g. M=Hash(Input), for 2^{128} security)
- Encode as a braid: $\mathcal{E}(M)$
- Choose a random number $1 < i < n$
- Generate *Cloaking Elements* v and v_1 as purebraids
  - Built by conjugating a square of a braid generator ($w\, b_i^2\, w^{-1}$), where
  - The permutation of w takes element i to a and $i+1$ to b for v (v_1 is constructed in an analogous manner)
  - The purpose of the cloaking elements are to obscure $Priv_S$ from the CSP. They stabilize the E-Multiplication action.
- Signature is: $Priv_S^{-1}\, v\, \mathcal{E}(M)\, Priv_S\, v_1$ – rewritten to hide the form

More on WalnutDSA Cloaking Elements

- Recall that CSP requires a braid of the form $z\,a\,z^{-1}$
- A *Cloaking Element* is a special braid that disappears during E-Multiplication but protects a braid from the CSP by preventing the necessary form
- Formed by a braid $\beta_{CE} = w\,b_i^2\,w^{-1}$
- The result is that $(1,1) \star \beta_{CE} = (1,1)$
- Based on the actual form of w and choices of a, b, and i, a cloaking element can also result in $(1,\sigma) \star \beta_{CE} = (1,\sigma)$

Note: Cloaking Elements are a novel idea by SecureRF.

WalnutDSA Signature Validation

- Take a 256-bit message (e.g. M=Hash(Input))
- Encode as a braid: $\mathcal{E}(M)$
- Compute the E-Multiplication: $(Mat_M, Perm1) = (1,1) \star \mathcal{E}(M)$
- Multiply Mat_M with Pub_S: $Mat_1 = Mat_M \cdot Pub_S$
- Compute the E-Multiplication: $(Mat_2, Perm2) = Pub_S \star Sig$
- Verify that the matrices are equal:
  $Mat_1 =? Mat_2$

WalnutDSA: Did you notice?

- Structurally similar to DSA, ECDSA (versus RSA signatures)
- No TTP Required
- No Public Conjugates
- No Public System-wide Data

Performance of WalnutDSA Verification

$B_8F_{32}, 2^{128}$ Security level (equivalent to ECC P256)

Platform	Clock	WalnutDSA			ECDSA			Gain
	MHz	ROM[1]	RAM[1]	Time[2]	ROM[1]	RAM[1]	Time[2]	(Time)
MSP430	8	3244	236	46	20-30K [3]	2-5K	1000-3000	21-63x
8051	24.5	3370	312	35.3				
ARM M3	48	2952	272	5.7	7168 [4]	540	233	40.8x
FPGA	50			0.05			2.08 [5]	41.6x

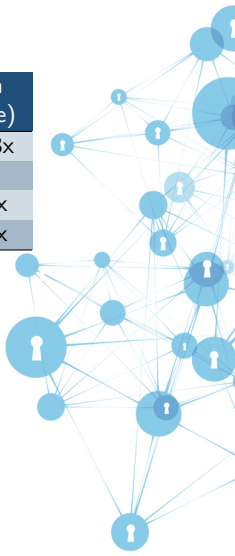
[1] ROM/RAM in Bytes

[2] Time is in milliseconds.

[3] C.P.L. Gouvêa and J. López, *Software implementation of Pairing-Based Cryptography on sensor networks using the MSP430 micro controller*, Progress in Cryptology, Indocrypt 2009

[4] Wenger, Unterluggauer, and Werner in 8/16/32 *Shades of Elliptic Curve Cryptography on Embedded Processors in Progress in Cryptology*, Indocrypt 2013

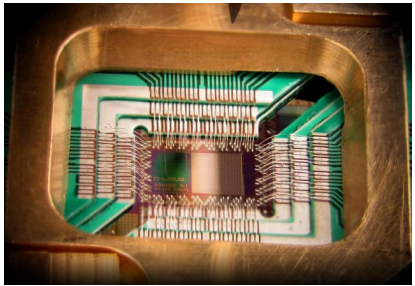
[5] Jian Huang, Hao Li, and Phil Sweany, *An FPGA Implementation of Elliptic Curve Cryptography for Future Secure Web Transaction*, 2007.



SECURE RF
Securing the Internet of Things®

"The National Security Agency is advising US agencies and businesses to prepare for a time in the not-too-distant future when the cryptography protecting virtually all e-mail, medical and financial records, and online transactions is rendered obsolete by quantum computing"

Source: Ars Technica, August 21, 2015



D-Wave System Chip with Quantum Properties

SecureRF's Cryptography Methods are quantum-resistant to all known attacks

SECURE RF
Securing the Internet of Things®

Quantum Resistance

- Two important quantum methods: Shor's Algorithm and Grover's Search Algorithm
- Shor: Breaks ECC, RSA, and DH by quickly factoring or solving the discrete log problem
  - Requires the method's math be Finite, Cyclic, and Commutative
  - GTC is neither Cyclic nor Commutative, and the underlying group is Infinite, so Shor does not apply
- Grover's Search Algorithm reduces the security level (e.g. AES-128 becomes 64-bit secure)
  - Doubling the security of GTC requires doubling the key size which only doubles the runtime

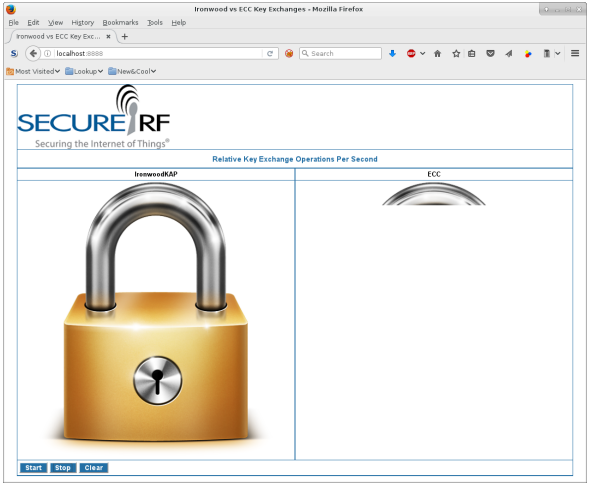
Commercial Use Cases

- Anti-counterfeiting
  Example: an industrial partner protecting replacement parts
- Credentialing
  Example: a faster, quantum-resistant replacement for a CAC Card
- Fast, Quantum-Resistant Secure Boot
  Example: a RISC-V processor that validates its firmware before it runs
- Mutual Authentication of IoT Devices
- Chips that can prove their identity without secure databases

Demonstration

Ironwood versus ECDH speedtest on a Microsemi Smartfusion2

- 182μs versus 370ms per authentication
- ARM: 100MHz, Fabric: 50Mhz



Conclusions

- GTC enables novel Public Key methods suitable for low-power, passive, and otherwise constrained devices
- Significantly faster than ECC, RSA, and DH
- Performance scales linearly (as opposed to quadratically) as security increases
- The underlying core technology (E-Multiplication) can also be used to form block cipher, hash, and prng
- Algorithms are quantum resistant
- Has a small footprint for hardware and software implementations

We believe GTC has a big future in the IoT universe.

Thank You!

Any Questions?

SecureRF Corporation
100 Beard Sawmill Rd, Suite 350
Shelton, CT 06484
(203) 227-3151

Derek Atkins, CTO (datkins@securerf.com)