
# Walnut Digital Signature Algorithm™: A lightweight, quantum-resistant signature scheme for use in passive, low-power, and IoT devices

Derek Atkins, SecureRF Corporation

*Historically "Lightweight Cryptography" has focused on symmetric schemes, yet asymmetric methods can also work effectively in these environments. Specifically, the Walnut Digital Signature Algorithm™ (WalnutDSA™) provides a public-key signature scheme that verifies signatures significantly faster than ECC in both software and hardware, even in small, constrained environments and is resistant to all known attacks, both conventional and quantum. Specifically, WalnutDSA can validate a signature anywhere from 32 to 178 times faster than ECC using less ROM/RAM or fewer gates. This presentation previews the Walnut Digital Signature Algorithm, performance results, and shows how it can be applied to real-world uses.*

SecureRF Corporation
100 Beard Sawmill Road
Suite 350
Shelton, CT 06484

203-227-3151
info@SecureRF.com
www.SecureRF.com

# 1. Introduction

As Moore's Law continues, smaller and smaller devices are gaining more and more computational power. Where in the previous decades end systems were home personal computers, laptops, and even cloud servers, these days systems are getting smaller and more constrained. Growing into the Internet of Things, no longer are systems as powerful as before, as companies spend pennies to add computational resources to devices as small as light bulbs. However as these devices get smaller, the need to secure them grows exponentially.

As more devices get connected the need to identify, authenticate, and protect those devices grows. Real-world devices (e.g. a door lock) needs to authenticate the source of command and control messages, to ensure that a command to unlock the door originates from an authorized source and is not spoofed by an attacker trying to gain entry. However these devices are so constrained that existing technologies either cannot address the space or power constraints, or their performance is so slow that the devices are unusable for real-world activities.

Enter "Lightweight Cryptography." Lightweight Cryptography is used to define the study of algorithms that are efficient in extreme conditions such as limited CPU, RAM, ROM, or power. It often refers to only symmetric schemes, but SecureRF believes it can (and should) apply to suitable public-key algorithms as well.

As ubiquitous as the Internet is today, it was not until the 1990's that the necessary security protocols were put in place that enabled users to function securely while online. For example, the online revolution in the financial sector, e-commerce, and medical records, all rely on the security breakthrough provided by public-key cryptography. Public-key cryptosystems fall into two categories: the Diffie-Hellman (DH) protocol published by Whitfield Diffie and Martin Hellman in 1976, and the RSA protocol, as publicly described by Ron Rivest, Adi Shamir, and Leonard Adleman in 1978. The deployments of RSA, DH, and a more recent Diffie-Hellman like method, Elliptic Curve Cryptography (ECC), have brought solutions to the current Internet.

In many respects, we are at a similar evolutionary point with devices found in the Internet of Things (IoT) that are used for connectivity, credentials, the Smart Grid, and industrial controls. The almost 40-year-old solutions that enabled the Internet revolution are not suitable for microcontrollers; wireless sensors and devices utilizing Near Field Communication (NFC), radio frequency identification (RFID) or Bluetooth Low Energy (BLE); embedded systems and other low-resource environments being implemented as part of the IoT. A new generation of public-key infrastructure needs to emerge.

An issue with today's commonly implemented RSA and Diffie-Hellman type public-key protocols, including ECC, concerns the computational footprint they entail. While memory and energy usage is not a primary concern for most environments requiring cryptographic security, these issues, along with runtime, lie at the heart of any small computing device security discussion. Every one of these cryptographic systems at its core utilizes multiplication of large numbers. As a result, the computing resources required to achieve security grow rapidly as the level of security is increased. Even ECC, with its lower resource usage than RSA or DH, provides inadequate performance in constrained devices, to the point where developers are not even considering public key solutions.

This presentation proposal describes our work in progress, a new public key method called the Walnut Digital Signature Algorithm (WalnutDSA). We propose to briefly describe the method, analyze the performance, and show how its efficiencies can provide improved performance in many constrained

environments. In a forthcoming paper by Anshel-Atkins-Goldfeld-Gunnells, a rigorous description and security analysis of WalnutDSA will be presented.

# 2. The Walnut Digital Signature Algorithm

WalnutDSA is a Group Theoretic public-key signature scheme with its execution and security based in three distinct areas of mathematics: the theory of braids, the theory of matrices with polynomial entries (expressions of finite length constructed from variables), and modular arithmetic in small finite fields. At its core is a highly specialized one-way function known as E-Multiplication™, which brings together these mathematical tools and enables the system to provide high-speed security without overwhelming the memory and power available. This core function is quantum resistant and highly resistant to reverse engineering due to the connections with mathematically intractable problems arising from the symbiotic combination of these disparate fields of mathematics.

E-Multiplication was originally defined in 2006 in a paper published by The American Mathematical Society in the peer-reviewed book "Algebraic Methods in Cryptography" in the section titled *Key agreement, the Algebraic Eraser™ and Lightweight Cryptography*. A version of this paper is available on SecureRF's website (http://www.securerf.com/wp-content/uploads/2014/03/SecureRF-Technical-White-Paper-06-with-Appendix-A-B.pdf).

Using E-Multiplication the original authors constructed a key agreement protocol (called the Algebraic Eraser Key Agreement Protocol, AEKAP) which proved to run extremely fast even in the smallest of environments. This speed is due to the efficiency of E-Multiplication, even in 16- and 8-bit environments. Indeed, AEKAP was presented at the NIST Lightweight Cryptography Workshop in 2015.

Since its introduction in 2006, the key agreement construction has withstood several attacks, but none of the attacks were against E-Multiplication or the underlying hard problems in group theory. One attack against AEKAP (KTT: Kalka-Teicher-Tsaban, *Short Expressions of Permutations as Products and Cryptanalysis of the Algebraic Eraser*) showed that randomly chosen keys are weak, but keys chosen properly are not susceptible to the attack, as explained in *Defeating the Kalka-Teicher-Tsaban Linear Algebra Attack on the Algebraic Eraser* by Goldfeld and Gunnells. Another attack (MU: Myasnikov-Ushavok, *Cryptanalysis of the Anshel-Anshel-Goldfed-Lemieu Key Agreement Protocol*) showed that key material that is too short can be a weakness, but if suggested key lengths for implementation are longer then their attack fails 100% of the time as shown by Gunnells in *On the Cryptanalysis of the Generalized Simultaneous Conjugacy Search Problem and the Security of the Algebraic Eraser*. More recently Ben-Zvi, Blackburn, and Tsaban (BBT) released an attack (*A Practical Cryptanalysis of the Algebraic Eraser*) that builds on the earlier, defeated, KTT attack and showed that with access to all the key material, both public keys, and after a long pre-computation they could recover the shared secret in eight hours. However their attack is predicated on a conjecture that they can always find short expressions in certain permutations. In January, 2016 a paper titled *Defeating the Ben-Zvi, Blackburn, and Tsaban Attack on the Algebraic Eraser* showed that it is possible to generate key material where their conjecture doesn't hold.

All these attacks were purely against the AEKAP construction. E-Multiplication is still considered a strong one-way function. Indeed, a Hash function based on E-Multiplication was published in 2016, and Blackburn was the editor of the journal where it was published.

Digital signatures based on hard problems in group theory are relatively new. Ko, Choi, Cho, and Lee proposed, in 2002, a digital signature based on a variation of the conjugacy problem in non-commutative groups. Wang and Hu in 2009 introduced a digital signature whose security is based on the hardness of the root

problem in braid groups. However several attacks suggest that these schemes may not be practical over braid groups in low resource environments.

Enter WalnutDSA, a digital signature construction based on E-Multiplication. In WalnutDSA the Private Key is a braid, as is a signature. The public key is a matrix and permutation along with a set of finite field elements (called T-Values). No TTP is required. And finally, validating a signature is as simple as performing two sets of E-Multiplication operations, a matrix multiplication, and then comparing two matrices. As has been shown time and again, these operations are extremely fast, and actual performance numbers will be shown in the next section.

None of the attacks against AEKAP apply to the WalnutDSA construction. There are no conjugates for a KTT- or BBT-style attack to work. The signatures are long enough than an MU-style attack won't work (and indeed, the shortest word problem is known to be NP-Hard in the braid group). WalnutDSA works completely differently.

## 2.1. WalnutDSA Design

WalnutDSA combines braids, matrices, and operations over finite fields. This section provides an overview of the WalnutDSA construction. Before beginning one must first choose the braid group and finite field to use. For these exercises we chose $B_8$ (the braid group over a braid with 8 strands) and $F_{32}$ (a finite field with 32 elements). We target a security level of $2^{128}$against brute force attacks, but can increase that.

Unlike the AEKAP construction, Alice generates her Public/Private key pair by herself. She chooses a set of unique T-values (an array of N-8 elements in $F_{32}$) and then generates a private braid, $Priv_S$, of a specific form. The braid can be chosen to meet the specific security criteria, such that the chance of choosing the braid at random is at the desired security level. The higher the desired security, the longer $Priv_S$ becomes.

To generate her public key Alice merely uses E-Multiplication. Specifically the Public Key, $Pub_S = (1,1)\star Priv_S$, where (1,1) is the Identity Matrix and Identity Permutation and $\star$ denotes the E-Multiplication operation.

A WalnutDSA Signature is also a braid of a specific form. To sign a message Alice first generates the hash of the message: M = Hash(Message). Then Alice needs to encode that message into a braid word, E(M). Finally, Alice pieces the signature together from the private key, the encoded message, and two braid words called *cloaking elements*.

A cloaking element is a braid word of a special form that hides the contents of a braid but disappears during E-Multiplication. They are chosen specifically to fit the security requirements.

To validate the signature, Bob receives the message, hashes it to compute M, and then encodes M into a braid word using the same encoding mechanism to acquire E(M). Next Bob uses E-Multiplication to compute $(1,1)\star E(M)$ and multiplies that by the matrix part of $Pub_S$. Next Bob computes the E-Multiplication $Pub_S\star Sig$. Finally, Bob compares the two resulting matrices. If they are equal then the signature verified. If the matrices differ then the signature validation failed.

A full description of the method and a complete security analysis will be available in a forthcoming paper by Anshel-Atkins-Goldfeld-Gunnells.

## 2.2. Quantum Resistance

Researchers at MIT have successfully built a 5-atom quantum computer which was able to run Shor's algorithm and factor a small number. The question is not if a quantum computer will get built, but when one will get built that is sufficiently large to break existing RSA, ECC, and DH cryptosystems.

When looking at quantum resistance there are two main algorithms that need to be examined: Shor's algorithm that can be used to factor numbers (breaking RSA) and solve discrete logarithms (breaking ECC and DH), and Grover's algorithm that enables very rapid searching.

Shor's algorithm provides a solution to the Hidden Subgroup Problem. In order to solve this problem the overlying group must be finite, abelian, and cyclic. In other words the group must be limited in size, the members must commute, and once it wraps the sequence repeats. This is certainly the case in ECC, RSA, and DH.

The braid group is infinite and the internal operation of braid multiplication is non-commutative. Braids can go on forever; there is no limit on the length. When you multiply braids, order matters. And even when you reduce E-Multiplication to its matrix form (which does condense down to a finite space), it remains non-cyclic and non-commutative as well. As a result, Shor's algorithm cannot be used against any E-Multiplication based construction like WalnutDSA.

The second important quantum method is Grover's search algorithm which provides a quantum method that significantly reduces the security of any system by speeding up key/value searches. It applies to everything: hash algorithms, block ciphers, public key operations – any input/output problem that can be tested. The only protection against Grover is increasing the security size. Specifically, Grover cuts the security in half, so if you had a $2^{128}$ security, with Grover you now have $2^{64}$. The only way to "defeat" Grover is to increase the security level more to "bring you back" to the desired security level.

This is where the linear nature of E-Multiplication shines. In order to increase the security level of WalnutDSA to, e.g. $2^{256}$, all that is required is doubling the size of the keys and messages, which results in only double the amount of work to validate a signature! Compare this to ECC, where increasing from P256 to P521 more than quadruples the amount of effort required!

# 3. Implementation Experiences

WalnutDSA has been implemented in Software (C, and platform-specific Assembly) and a hardware (VHDL and Verilog) implementation is in the works. The following sections detail several implementations of WalnutDSA and how it compares to ECC.

Before we started we chose N=8, q=32 and a security level of $2^{128}$against brute force attacks. Using these parameters we generated a set of 50 keypairs and for each key we generated 50 random 256-bit messages (simulating the output of a hash) and then signed the message (resulting in a total of 2500 signatures). Because the private keys and signatures are variable length, this exercise provided data to estimate the average sizes we would expect to see.

The Public Key is a fixed size. It is an NxN matrix over q elements, a permutation, and the set of T-values. This requires $(N^2-N+1)*\log_2(q) + N*\log_2(N) + N*\log_2(q)$ bits. For our parameters this results in 285 + 24 + 40 = 349 bits (44 bytes).

The Private Key is a braid in $B_8$. After generating 50 key pairs the keys ranged in length from 64 to 117 Artin generators. Each generator requires 4 bits, so this ranges from 256 to 468 bits (32 to 59 bytes). The average length was 89.5 generators (358 bits, or 45 bytes) with a standard deviation of 10.78 generators.

The Signature is also a braid in $B_8$ of variable length, depending on the private key, cloaking elements, and encoded message. After generating 2500 signatures, they ranged in size (before any rewriting) from 810 to 1128 Artin generators, with an average of 966.0 and a standard deviation of 51.67. While this is interesting, the more important length is after rewriting. For rewriting we used a combination of Birman-Ko-Lee canonical form (BKL) and Dehornoy's algorithm. This resulted in signatures of length 642 to 1692 Artin generators, with an average length of 1066.6 and a standard deviation of 143.7. With 4 bits per generator this means the average signature was 4266 bits (534 bytes).

The following sections details our implementation experience on several different constrained platforms. For testing purposes we chose a signature close to average length, 537 bytes, to give an average validation time.

## 3.1. TI MSP430

The Texas Instruments (TI) MSP430 microcontroller is used extensively for sensors and low power (or even passive) devices. We used an MSP430F5172 for our testing, and ran it at 8MHz. Because of the size of the fields we used, we were able to use a lookup table for the finite field operations. This required 1KB of storage for the multiplication table.

Compilation proceeded using the GCC compiler version 4.9.1 (20140707) using optimization level -O3. With that configuration the WalnutDSA implementation consumed a total of 3244 bytes of ROM, and running the code required only 236 bytes of RAM. The verification completed in 370944 cycles, which at 8Mhz equates to only 46mS. Compare this an estimate of ECDSA of over 1-3 seconds.

Others have researched ECC on the MSP430; In particular Software *implementation of Pairing-Based Cryptography on sensor networks using the MSP430 micro controller*, by C.P.L. Gouvêa and J. López (published in Progress in Cryptology –- Indocrypt 2009) shows results that required upwards of 20-30KB of ROM, 2-5KB of RAM, and required anywhere from 1000 to 3000ms to validate a signature.

## 3.2. ARM Cortex M3

The ARM family of processors is probably the most widely used 32-bit CPUs in the IoT space. While the ARM Cortex M0 is the smallest of the set, we focused on the ARM Cortex M3 (using an NXP LPC1768), a very popular choice and easily available for development. This board runs at 48MHz.

GCC was also used to compile on the ARM platform, using version 4.9.3 (20150303) also using optimization level -O3. On the ARM platform we also used lookup tables for the finite field operations. This resulted in a code size of 2952 bytes of ROM, and running the verification required 272 bytes of RAM. The signature verification completed in a total of 275563 cycles, which is only 5.7mS.

Now let us compare AE to ECC. Wenger, Unterluggauer, and Werner in *8/16/32 Shades of Elliptic Curve Cryptography on Embedded Processors* in Progress in Cryptology, Indocrypt 2013, showed a full assembly-language implementation of ECC in 7168 bytes of ROM that required 540 bytes of RAM but still required 233ms to perform a point multiplication (on a Cortex M0). We also created an implementation using WolfSSL in C; this used 9780 bytes of ROM, 7456 bytes of RAM, and required 889ms to do the same. Moreover,

employees at ARM reported at the IETF-92 in March 2015 (H. Tschofenig, M. Pégourié-Gonnard, *Crypto Performance on ARM Cortex-M Processors*) that they measured an ECDSA verify on the LPC1768 in 458ms.

## 3.3. 8051

The 8051 8-bit microcontroller is another popular platform used in low power sensors, including many passive NFC targets. We chose an implementation of the 8051 from Silicon Industries. This specific chip runs at 24.5 MHz.

This platform is unique in the way it handles RAM, including a specific "relocatable" section. Code on the 8051 was built using the Keil V9.54 system with the small memory model and optimization set to OPTIMIZE(11,SPEED). On this platform we implemented WalnutDSA using a combination of C and 8051 Assembly. We compiled WalnutDSA into 3370 bytes of ROM. Running WalnutDSA required a total of 312 bytes of RAM, split into 251 bytes of "xdata", 3 bytes of "data", and 58 bytes of "relocatable data." With this implementation WalnutDSA required 864101 cycles, which implies a runtime of 35.3mS.

## 3.3. Field Programmable Gate Arrays

More impressive than software is the improvement of WalnutDSA over ECDSA in hardware. For our test cases we focused on Field Programmable Gate Arrays (FPGAs) as these are popular devices and allow easy development and testing. The devices tested typically have a fabric clock speed of 50Mhz, and they can vary in size significantly.

Often an FPGA will have an embedded processor that runs an OS, and then can leverage the rest of the FPGA to optimize operations for improved performance. Of course it takes time to pass data back and forth between the processor and the hardware fabric, and that is taken into account. In the case of testing WalnutDSA, we needed to pass 161 words into the fabric. The amount of time this takes it dependent on the actual FPGA in use.

We tested on two different FPGA systems, a Microsemi Smartfusion 2 and an Altera Cyclone V SoC.. The major variation in execution times was due to the data transfer time, but in total we could perform a signature validation in under 5000-10000 cycles. This implies (at 50MHz) an execution time of only 100-200uS!

When compared to ECC, validations take in the range of tens to hundreds of milliseconds. Moreover, increasing the speed of ECC can be accomplished by parallelizing the implementation. However doubling the gate count does not reduce the runtime by half. Indeed, the implementation hits diminishing returns, doubling the size to reduce the runtime by 10%.

## 3.4. RISC-V

RISC-V is a Reduced Instruction Set architecture that originated from a project out of University of California, Berkeley. It's an open source Instruction Set for embedded CPUs that allows manufactures to develop cores without paying a royalty to companies that own the IP. Because of the open nature of the RISC-V ISA, it's possible to add custom instructions to implement hardware speed optimizations.

SecureRF worked with a partner to implement a WalnutDSA verifier on a Z-Scale inspired RISC-V core implemented in a Microsemi Smartfusion 2+ board. On this platform we also implemented an ECDSA verify routine from Micro-ECC to compare the two. This CPU ran at 50MHz. In software, Micro-ECC required

9076 bytes of ROM and verified a signature in 680ms, whereas WalnutDSA compiled into 4504 bytes and executed in 10.7ms (a 63x speed improvement).

Adding only a 2.3% hardware overhead we implemented an enhancement to allow multiple Galois Field operations to occur simultaneously. This not only reduced the code size to 3052 bytes (a 25% reduction), but sped up the verification to 3.8ms, resulting in a 178x speed improvement over Micro-ECC.

# 4. WalnutDSA Standards Initiatives

SecureRF is active in numerous standards organizations and will be attempting to add WalnutDSA to various standards. Specifically, SecureRF is active in JTC-1/SC-31/WG-4 and JTC-1/SC-27/WG-2. However due to certain requirements we must wait additional time before we are allowed to introduce WalnutDSA. SecureRF is also active in the IETF and plans to propose WalnutDSA there as well.

# 5. About SecureRF

The inventors of SecureRF's public-key cryptosystem are co-founders Dr. Michael Anshel, Dr. Dorian Goldfeld and Dr. Iris Anshel.

Dr. Michael Anshel is a security thought-leader and world-class mathematician with expertise in the field of cryptography. Dr. Anshel has authored and co-authored numerous papers in the area of public-key cryptography, is the co-inventor of four patents in the area of cryptography, zeta-one-way functions, and braid group and has received numerous fellowships and honors. He is a Professor Emeritus in the Department of Computer Science at The City College of New York.

Dr. Goldfeld is a world-class mathematician who has published over 100 papers and books and lectured internationally on a wide range of cryptographic topics and methods including applications of elliptic curves, quadratic fields, zeta functions, public-key cryptography, and group theoretic approaches to public-key cryptography. In 2009 he was inducted as a Fellow of the prestigious American Academy of Arts & Sciences. He is the co-inventor of five patents in the areas of multistream encryption systems, high speed cryptography, cryptographically secure algebraic key establishment protocols based on monoids, and cryptographic hash functions. He has been a professor in the Faculty of Mathematics at Columbia University since 1985.

Dr. Iris Anshel, SecureRF's Chief Scientist, is an accomplished mathematician and cryptographer. In addition to extensive research and publications, Dr. Anshel has experience in the commercialization of security technology. As a co-founder of Arithmetica Inc she was responsible for documenting methods for commercial deployment of new cryptography protocols including the AAG Braid Group Cryptosystem and supported sales and business development activity. She is co-inventor on three patents in the areas of cryptographically secure algebraic key establishment protocols based on monoids, and cryptographic hash functions.

Derek Atkins is the Chief Technology Officer at SecureRF, bringing over two decades of security protocol and implementation experience to the team. He has been a significant influence in the Internet Engineering Task Force (IETF), chairing several working groups and participating in multiple directorates including the Security Area Directorate. His software engineering experience includes numerous open- and closed-source projects including an implementation of the Pretty Good Privacy (PGP) protocol that has continued to last for over 15 years across five different company ownerships. His experience and focus ensures SecureRF delivers the optimum experience to its users and customers in both speed and security.

SecureRF Corporation – Securing the Internet of Things® – provides security solutions for embedded systems and wireless sensor technologies used in non-traditional payment systems, secure supply chain

management, cold chain management, and anti-counterfeiting applications in the pharmaceutical, fashion, spirits, defense, and homeland security sectors. The company's technology is based on a breakthrough in public-key cryptography that is computationally efficient, yet highly secure and available as a software development kit, Verilog/VHDL, or as a core for FPGAs and ASICs. SecureRF also offers the LIME Tag™ - a range of highly secure NFC, UHF and Bluetooth LE sensor tags along with its anti-counterfeiting solution – Veridify™.