Spring 2014

IMPROVING SMART GRID SECURITY USING MERKLE TREES

Melesio Calderón Muñoz
San Jose State University

IMPROVING SMART GRID SECURITY USING MERKLE TREES

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree Master of Science

by

Melesio Calderón Muñoz

May 2014

The Designated Masters Writing Project Committee Approves the Project Titled

IMPROVING SMART GRID SECURITY USING MERKLE TREES

by

Melesio C. Muñoz

APPROVED FOR THE DEPARTMENT OF COMMUNICATION STUDIES SAN JOSÉ STATE UNIVERSITY

May 2014

Dr. Melody Moh	Department of Computer Science	Date
Dr. Teng Moh	Department of Computer Science	Date
Dr. Mark Stamp	Department of Computer Science	Date

***Abstract*—Presently nations worldwide are starting to convert their aging electrical power infrastructures into modern, dynamic power grids. Smart Grid offers much in the way of efficiencies and robustness to the electrical power grid, however its heavy reliance on communication networks will leave it more vulnerable to attack than present day grids. This paper looks at the threat to public key cryptography systems from a fully realized quantum computer and how this could impact the Smart Grid. We argue for the use of Merkle Trees in place of public key cryptography for authentication of devices in wireless mesh networks that are used in Smart Grid applications.**

I. INTRODUCTION

The electrical power grid infrastructure forms the functional foundation of our modern societies. The grid has served humanity well to now, but in the near future our electric power grids will reach a limit. If left as they are power grids will not keep pace with our demands of them. By 2050 worldwide consumption of electricity is expected to triple [1]. Governments and organizations have begun developing plans to implement electrical power grids with improved functionality, reliability and efficiency. These new grids will have "advanced decentralized, digital, infrastructures with two-way capabilities for communicating information, controlling equipment and distributing energy" [2]. Generally referred to as the Smart Grid, this infrastructure will be better able to incorporate new forms of energy generation, as well as be self-healing, and more robust. However, these new systems will be complex, and will have the potential for numerous vulnerabilities.

Smart Grid will combine several well established yet, distinct industries, namely the electrical power, information technology and communications industries. These industries have different priorities and goals. For instance, in the electrical industry the highest priority is human safety. In the information technology industry the highest priority is confidentiality, integrity and availability of information. For the Smart Grid cyber security measures must not get in the way of the safe and reliable power system operations [3].

Each device in the new grid will likely have its own IP address and will use protocols like TCP/IP for communication. Thus they will be vulnerable to similar security threats that face present day communication networks [4], however the stakes will be much higher. That is, if present day social networking websites are brought down, or even if financial records are compromised, unpleasant situations could arise. However, if a hospital loses power, then human lives are endangered. The Smart Grid will take a generation to create and will bring great benefits to the power grid, but also tremendous risks in allowing so many devices to communicate with each other.

This paper looks at one aspect of Smart Grid security, namely the use of public key cryptographic systems in electrical power grid equipment in the context of the potential future threat from a quantum computer attack. This paper argues for the use of Merkle (Hash) Trees as opposed to public key cryptography systems in the Smart Grid, specifically when used to authenticate wireless devices.

A. PROBLEM STATEMENT

In recent years wireless mesh networks (WMN) have received a lot of attention and have becoming increasingly popular topologies due to their cost effectiveness and robustness. With a WMN topology it is possible to cover the same area as with typical WiFi, but with less routers. This makes them economically pragmatic to use. It seems likely WMN will be widely used in the Smart Grid.

Authentication enables a node to ensure the identity of the peer node it wants to communicate with. This must be done in a network to keep it secure when a new device is added. Public key cryptography is a means by which this can be accomplished. The security of the public key system rests on a difficult math problem, which we will discuss later. This problem is difficult today, however, if a new type of computer, called the quantum computer, is fully realized then this entire system is broken. This is something we know today.

Merkle trees can also be used for authentication, but their security rests on the use of cryptographically secure hash functions, which we understand to be resistant to a quantum computer attack. Merkle trees were developed in the late 1970's by Ralph Merkle. A Merkle tree is a complete binary tree constructed from a set of secret leaf tokens, where each internal node of the tree is a hash of its left and right child. The leaves consist of a set of m randomly generated secret tokens. Since it is a complete binary tree $m = 2^h$ where h is the height of the tree. The root is public and the result of recursive applications of the one-way hash function on the tree, starting at the leaves [5].

For this work we implement a Merkle tree authentication scheme and incorporate it into the ns-3 Network Simulator. We then compare its performance to that of a publicly available version of a well known public key cryptographic system, namely RSA. This comparison will be based on time to compute, use of memory and number of exchanges required. Our goal is to show that Merkle trees are a reasonable alternative to public key in terms of computational time and functionality yet since they are resistant to a future attack from a quantum computer, they are preferred.

By its nature electrical equipment can difficult, disruptive, and expensive to replace or modify. By using Merkle trees for authentication now, we can avoid future problem.

II. Background

In this section we look at an example that demonstrates some of the weaknesses of our current power grid. We also look at some of the early steps that have been taken to implement the Smart Grid and security issues that have already arisen.

A. The Northeast Blackout of 2003

In August 2003 the U.S. Midwest and Northeast as well as Ontario, Canada experienced a blackout that left fifty million people without of power for days, with some parts of Ontario experiencing rolling blackouts for more than a week. This was the largest power outage in North American history [6].

In cases of natural disaster like storms or earthquakes, visibly destroyed infrastructure makes it easy to understand why power was lost and why it takes so long to restore. Unlike those types of events, the 2003 blackout was triggered by a simple overgrowth of trees in Ohio. 14 August 2003 was a hot day, although not unreasonably hot for this region of the U.S. in the summer. The hot weather created a large load on the grid; this was mostly the result of air conditioner usage. In addition several key power generators were off-line, causing voltage in this area to be depressed. In this part of the Midwest there are many utility companies. FirstEnergy, which is composed of seven utility companies, was the domain where the fault occurred. North American Electrical Regulatory Council (NERC) rules mandate that utilities must inform their neighbors when they are short on power. FirstEnergy did not do this.

Power lines normally expand and contract due to the effects of load and weather. On this day in the Cleveland-Akron, Ohio area sagging high voltage transmission lines came in contact with overgrown tree branches causing a ground fault [7]. At 3:05pm EDT those lines tripped. The failure of these transmission lines subsequently caused lower voltage lines to surge and trip. These

failures then caused adjacent high voltage transmission lines to surge and trip, which at 4:06 EDT caused an uncontrolled cascading failure [8]. Millions of people, in cities like Cleveland, New York City, Baltimore and Toronto, were left without power.

Although the most immediate cause of the failure was the transmission lines coming into contact with untrimmed trees, there were four underlying causes as documented in official U.S.-Canadian report on the incident:

> ***Reason #1*:** FirstEnergy's failure to assess and understand the inadequacies of its system, particularly with respect to the voltage instability and vulnerability of the Cleveland-Akron area. FirstEnergy was not operating with the appropriate voltage criteria.
>
> ***Reason #2:*** Inadequate situation awareness at FirstEnergy as the situation began to unfold.
>
> ***Reason #3:*** FirstEnergy's failure to manage tree growth in its transmission right-of-ways.
>
> ***Reason #4:*** Failure of the interconnected grid's reliability organization to provide effective real-time diagnostic support [9].

It is not unreasonable to conclude that the effects of this event could have been dampened, or even prevented, had an *advanced decentralized, digital infrastructure with two-way capabilities for communicating information, controlling equipment and distributing energy* been in place.

B. Implementing the Smart Grid

In the U.S. the Energy Independence and Security Act of 2007 established that the National Institute of Standards and Technology (NIST) has the "primary responsibility to coordinate development of a framework that includes protocols and model standards for information management to achieve interoperability of smart grid devices and systems" [10]. The NIST is a division of the U.S. Department of Commerce. NIST issues standards and guidelines it hopes will be adopted by all computer systems, thereby promoting interoperability that will in turn promote economic development [11]. In recent years NIST has generated many important documents related to smart grid technology and concerns; including the *NIST Framework and Roadmap for Smart Grid Interoperability Standards* and the *NIST Guidelines for Smart Grid Cyber Security*. Version 2 of the *Framework* (as it is referred to) was released in 2012. The *Guidelines* version 1.0 was released in 2010. Future releases of both documents are expected.

These documents offer tools for organizations involved with research and implementation of Smart Grid technologies. They are a starting point, a guide and a means to evaluate the framework of the Smart Grid. These documents are not intended to be a report that prescribes solutions to issues [12]. Academic, governmental and industrial institutions are the target audience of these documents, and they are encouraged to "identify research and develop topics based on gaps in technical areas related to the functionality, reliability, security and scalability requirements of the Smart Grid" [13].

In the U.S. most of the electrical power infrastructure was built up in the 1950s [14]. Electrical equipment is usually installed with the intention that it will be in service for many years, even decades. To do otherwise would not be efficient or acceptable. Computer and communication technology advance at a much more rapid pace. As a result, technology on the grid tends to lag.

Many functions in the grid today continue to use communications technologies similar to those that were used in the 1980s and 90s for personal computer dial-up connections [15]. Considering the expense, potential for disruption, and difficult to reach locations of some of this equipment, it seems clear why it is not updated with the latest trends in the computer world; the electrical power grid does not abide by Moore's Law.

In the Smart Grid components like power generators, switchgear, control systems and the like will now be networked together much like Internet nodes, sending and receiving messages related to the health and operation of the system. The Smart Grid will not be comprised of a single communication technology. Rather, it will utilize different technologies and protocols at different levels. It will be a network of networks [16]. "Networks are defined by connections between multiple locations or organizational units and are composed of many differing devices using similar protocols and procedures to facilitate a secure exchange of information" [17]. To ensure the reliability of the grid, data integrity must be maintained. Failure to do so could then leave the system open to compromising of devices, man-in-the-middle attacks, denial of service attacks, introduction of malicious code and so on.

C. WIRELESS MESH NETWORKS

In a wireless mesh network each node is a peer, i.e., there is no special base station node. Messages can then be forwarded through each peer to get to their final destination. This makes them more reliable since each node only needs to transmit as far as the next node, and each node is usually connected to several other nodes [18]. Additional robustness comes from the fact that there can be multiple routes from source to sink. The capability of a mesh network to create and modify routes also dynamically increases the reliability of the wireless connection [19]. A drawback with respect to the lack of a central infrastructure is that each node will be more complicated and as well as more expensive (both in terms of monetary cost and power required to run them). For devices that are battery powered this could be an important issue [20].

There are four main constraints on WMN, namely computing power, battery life, mobility and bandwidth [21]. The bandwidth on a WMN can be increased by adding more nodes. Typically a WMN is configured in three levels. The first level is the *Mesh Clients* that connects to the network via *Mesh Routers* at the second level. The *Mesh Routers* then connect to *Mesh Gateways,* at the third level, that provide access to other networks like the Internet [22].

WMN are extremely vulnerable to attacks due to their dynamically changing topology, absences of conventional security infrastructure, and wireless nature [23]. Wireless communications of the IEEE 802.11 and 802.15 varieties are known to be very easy to "sniff", i.e., eavesdrop on communications. Free network sniffers like *Kismet* and *WireShark* allow you to see network packets that are behind network communications. Sniffing packets is completely passive, i.e., this activity is not directly viewable by the target system. Passive attacks can compromise confidentiality. WMN are also vulnerable to active attacks, where the activity is directly viewable by the target system. These attacks could result in loss of availability, integrity, authentication and non-repudiation [24]. All wireless networks are inherently insecure because they are broadcast and have no real boundaries [25].

A basic network of the Smart Grid is constructed hierarchically: Home Area Networks (HAN), Neighbor Area Networks (NAN), and Wide Area Networks (WAN). NAN receives data from multiple HAN's and provides a backbone for data to be transferred to electrical providers [26]. Deployment of the Advanced Metering Infrastructure (AMI) has already started with the

installation of smart meters in the field. Smart meters provide detailed information on power usage at service locations, and usually form wireless networks. This gives us an early look at issues that could develop during this conversion to the Smart Grid.

D. THE ZIGBEE STANDARD FOR WIRELESS NETWORKS

The ZigBee standard defines a set of communications protocols for low-data-rate short-range wireless networking [27]. ZigBee devices operate in 868MHz, 915MHz and 2.4GHz frequency bands and have a maximum data rate of 250 K bits per second [28]. The ZigBee standard is comprised of four networking protocol layers. The lowest layer is the physical layer (PHY) and directly controls and communicates with the controls and radio transceiver. The next layer up is the Medium Access Control (MAC) layer with provides an interface between the PHY layer and network layer (NWK). The MAC layer is responsible for generating beacons and synchronizing devices. The NWK layer is next and is responsible for managing network formation and routing. Finally, the top layer is the applications layer (APL) which hosts the application object. The first two layers, PHY and MAC, layers are defined by the IEEE 802.15.4 standard [29]. ZigBee beholds to the 802.15.4 standard, but then goes beyond that to implement the top two layers. So if a device is ZigBee it beholds to 802.15.4, but if a device is 802.15.4 that does not mean it is ZigBee.

ZigBee is well suited for controls applications because it is a low power, low data communications protocol which can support a mesh topology. WiFi is complicated and its transceivers are expensive. Bluetooth uses a lot of power and is also complicated. The goal of ZigBee is to be simple and inexpensive [30]. ZigBee is meant for devices that do not interact very often, but when they are needed they need to power up and work quickly, then power down. In this way they preserve battery life, which is a concern for some of these applications. ZigBee is becoming increasingly popular since it is well suited for home automation, e.g., temperature sensors, lighting, smoke detectors, irrigation controllers, utility meters, et cetera [31].

Certicom is a cryptograph company that provides authentication services to ZigBee networks. They hold many patents related to the elliptic key cryptography (ECC) system. ECC is a public key cryptosystem based on the discrete log problem. Certicom uses ECC for their authentication [32]. If the quantum computer were available tomorrow all smart meters using this system would become insecure.

Today we see new products on the market for wireless lighting controls. Digital addressable controls have been on the market for some time, but wireless is new. They eliminate the cost and labor associated with running conduit and wire between a light switch or occupancy sensor, and a light fixture. Traditionally, this is how basic lighting control has been done. Now with these new systems pipe and wire is not required for controls. These wireless systems also allow for more fine grain control of lighting. These systems are expensive and not without their flaws, but they are now entering the market and finding favor among high tech companies that want to be in the vanguard of innovation. Many of the systems currently on the market are not fully ZigBee, but instead use the IEEE 802.15.4 standard, and then implement their own layers above that.

E. PSEUDO RANDOM NUMBER GENERATOR FLAWS IN SMART METERS

In 2010 security flaws were discovered in a radio chip set used in some smart meters. This provides a good case study of the vulnerabilities that the new Smart Grid will face. Smart meters form a part of what is known as the Advanced Metering Infrastructure (AMI). These have already been deployed in many countries at this point.

Smart meters tend to be wireless and in this case they were using the ZigBee standard for their communications. The flaws were discovered in the Texas Instruments radio chip set CC2530. The CC2530 is a 2.4GHz system-on-chip featuring an IEEE 802.15.4 compliant RF transceiver, which is suitable for ZigBee applications. The CC2530 broadcasts at a frequency band of 2400-2483.5, which is used worldwide and has up to 16 channels. 2.4GHz is not uncommon in many wireless devices such as WiFi networks, cordless phones, Bluetooth devices, et cetera [33].

The problems occurred in the CC2530 Z-Stack (ZigBee protocol stack) version 2.2.2-2.3.0. The CC2530 specification state:

> *The random-number generator uses a 16-bit LFSR to generate pseudorandom number, which can be read by the CPU or used directly by the command strobe processor. It can be seeded with random data from noise in the radio ADC* [34].

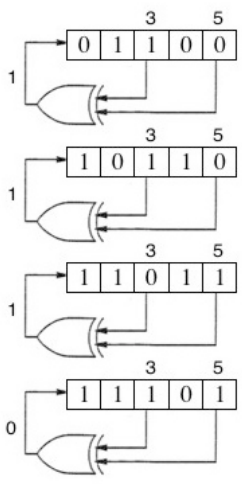
The first problem was the use of a 16-bit linear feedback shift register (LFSR) to generate random numbers. With 16-bits you will have 2^{16} or 65,531 internal possible states, i.e., random numbers. These random numbers were being used for encryption of data. A requirement for cryptographically random numbers is that they must be statistically random and unpredictable [35]. With 65,531 internal states, this falls well short of the requirements needed to maintain a secure system. We will discuss this below. The resulting random numbers were then used by the Certicom ECC crypto library to create session keys and digital signatures [36]. As we will see later an attacker only needs a subset of the output of the pseudo-random number generator (PRNG) to recreate the LFSR that generated them. With this information an attacker will be able to know ahead of time what new number is going to be generated, i.e., the numbers will no longer appear random.

The second flaw had to do with the ADC (Analog to Digital Converter) noted in the specifications above. Computers by their nature are not a good place to find randomness, however PRNGs can be very good in providing the required amount. To start, the generator requires a good (random) seed value, hence the use of the ADC. Unfortunately, for unclear reasons the seeds obtained from the ADC are not evenly distributed, in fact they cluster around only a handful of values [37].

This type of problem has been recognized in the NIST *Guidelines*; "Smart grid devices may have limited sources of entropy that can serve as a good source of true randomness"[38]. Entropy is a measure of the amount of randomness in a distribution [39]. The solution to this may require additional hardware or some regime of seeding a PRNG on a device before distribution, but that is a mater beyond the scope of this paper. The rest of this section will focus on the PRNG flaw.

"A PRNG is a single point of failure for many real world cryptosystems" [40]. That is, to say if the PRNG fails to produce a good random set of numbers, the entire crypto system is considered broken.

Fig. 1



A LFSR is a shift register whose input is a linear function of its previous states. Certain values of the shift register, called taps, are XORed together and then feed back into the register, the register then shifts 1 bit to the right. "An n-bit LFSR can be in one of 2^n-*1* internal states." The case where the LFSR is all zero is excluded because that would result in the output being zeros forever [41].

Figure 1 gives the first four stages of a 5-bit linear feed back register. The taps are at 3 and 5.

The goal of the attacker would be discover the function being implemented by the LFSR. Given the binary sequence being produced by the LFSR, the Berlekamp-Massey Algorithm can be used to

determine the smallest LFSR that can generate the sequence [42]. Using Berlekamp-Massey, this 16-bit LFSR is trivial to discover.

Texas Instruments corrected the 16-bit LFSR flaw in version 2.3.0 of its Z-stack firmware. It is unclear how many meters went into the field with this flaw, and how many of those have been corrected. Pacific Gas & Electric reports that 5 million smart meters have been installed in their dominion alone [43].

Smart meters have not been without their controversy. Largely this has stemmed from higher power bills, concern for radio frequency signals, but it has also involved the idea of privacy. Data will now be available that will record very detailed and personal accounts of individuals lifestyles, habits and other information much coveted by the curious, as well as the malicious. However, this information is of integral importance to the creation of a cleaner, more robust, power grid for the future. These issues only add weight to the fundamental importance of information security on the Smart Grid.

III. EXISTING SOLUTIONS

Authentication is a means by which trust is established in a network. RSA is a type of public key cryptosystem that can be used in this application. In this section we look at the strengths of RSA and how it is vulnerable to a quantum computer attack.

A. AUTHENTICATION

Usually the authentication mechanism in a non-WMN uses a centralized system, which administers access based on lists and certificates. Since WMN are decentralized, the use of such servers is not always possible [44]. To this end public key cryptography can be used for authentication. The NIST *Guidelines* suggests the use of public key cryptography on the Smart Grid, particularly as a basis for authentication operations [45].

In symmetric key cryptography, the same key is used to encrypt as to decrypt. These ciphers are fast and reliable, yet are problematic in regards to secure distribution of keys. That is, with the same key at both ends, if either key is compromised the whole system is broken. Public key, sometimes called asymmetric cryptography, uses two different keys, i.e., a public-private key pair. This has the advantage that if either key is compromised the system is not broken. Public key was developed in the 1960s and 70s by government and academic researchers. Considering that symmetric ciphers go back thousands of years, public key is very new [46]. Public key crypto systems are very robust and strong and if implemented correctly and offer some of the best protection available today. At its core the strength of the public key system rests on very difficult to solve math problems; namely, the factorization problem or the discrete logarithms problem. These topics will be explored further below.

B. AUTHENTICATION USING PUBLIC KEY

Public key can be used to authenticate two devices in the following manner. For clarity call one device *Alice* and the other *Bob*.

Alice sends a message to *Bob* claiming to be *Alice*. *Bob* needs more proof than this, so *Bob* encrypts a message *R* using *Alice*'s public key. Since the public key is public, anybody can encrypt a message, but only the holder of the private key can decrypt the message. Alice receives the message *R*, decrypts it then sends it back to Bob. Since she is the only holder of her private key, she has authenticated herself [47].

In the graphic $\{R\}_A$ represents encryption of message *R* using *Alice*'s public key.

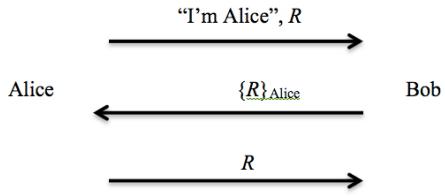


Fig. 2

C. RSA

RSA is a public key crypto system that is based on the factoring problem. At present the fastest computers and most efficient algorithms "cannot feasibly factor an arbitrary 1024-bit number" [48].

To create a public-private key pair two large prime numbers *p* and *q* are multiplied together to generate *N*. Then a value *e* is chosen at random that is the relative prime (i.e., their greatest common divisor is 1) of the product

$$(p-1)(q-1),$$

Finally we find the multiplicative inverse of

$$e \bmod (p-1)(q-1)$$

and call it *d*.

Now we have all the components we need to satisfy

$$ed = 1 \bmod (p-1)(q-1).$$

At this point *p* and *q* can be forgotten.

The RSA key pairs then are:

Public Key: (N, e)
Private Key: d

Now let *M* represent our plaintext message and let *C* be our ciphertext. To encrypt we calculate the following:

$$M^e \bmod N = C$$

To Decipher we calculate:

$$C^d \bmod N = M$$

Here is a simple example [49] :

$$p = 47,\ q = 71$$

Then

$$N = p\,q = 3337$$

$$(p - 1)(q - 1) = 46 * 70 = 3220$$

Choose a random number *e* (must be relative prime to 3220) say 79. Then

$$d = 79^{-1} \bmod 3220 = 1019$$

e and *N* are published and *d* is kept secret.

Then to encrypt a message say:

$$M = 688232$$

Break into blocks:

$$M^1 = 688$$
$$M^2 = 232$$

Then:

$$688^{79} \bmod 3337 = 1570 = C^1$$
$$232^{79} \bmod 3337 = 2756 = C^2$$

Then your cipher text is *1570 2756.*

To decipher:

$$1570^{1019} \bmod 3337 = 688 = M^1$$

$$2756^{1019} \bmod 3337 = 232 = M^2$$

It is not know for certain at this point if factoring is "difficult" [50]. That is to say, the best factoring algorithm asymptotically is the number field sieve, which is an exponential-time algorithm [51]. The strength of the public key crypto system lies in the fact that solving the factorization problem in a timely manner is beyond the reach of the most powerful computers and most efficient algorithms today.

D. Argument against Public Key in Smart Grid

Public key has a lot of overhead in terms of resources including time and energy. Many devices on these networks can be resource poor. These in themselves are important issues, but the focus of this paper is the threat of the quantum computer.

The concern with information security is not just is it safe today, but will it be safe in 10, 20, 30 years? At one time the German Enigma machine was the state of the art in data encryption, today breaking it is a challenging graduate level homework problem. Hardware installed in an electrical power grid is meant to stay in place for many years and in the Smart Grid they will need to remain secure while in the field. The Smart Grid needs to be built to last and, as best as possible, it needs to be "future-proof" [52]. For this reason investigation into potential security problems related to networks is of great research interest at this stage of development of the Smart Grid.

E. The Threat of Quantum Information Processing

Today's computer architecture is based around the transistor. Invented in 1945 at Bell Labs, within 10 years the solid-state transistor completely replaced its predecessor the vacuum tube [53]. The transistor opened the door to the modern computer age by allowing computers to do more with smaller and smaller components. In recent years the trend has been to increase computing power, by increasing the number of cores on a processor, i.e., increasing computing power by adding more transistors. Multicore processors do indeed seem to be the future for computers in the near term and the potential for greater computational power seems close at hand as a result of the progress of miniaturization, "but this trend cannot continue indefinitely" [54]. Yet for the security of public key there is a greater threat, namely, a realized quantum computer that can break the factorization or discrete log problem in polynomial time.

This paper focuses on the issue with the factorization problem, but several well know and widely used discrete log problems are worth mentioning. Namely Diffie-Hellman, El Gamal, and elliptic curve cryptography (ECC) are all examples of systems that are based on the discrete log problem. "Once quantum computers become a reality, all currently accepted public key encryption systems will be completely insecure." The practical implications of this would be far reaching [55]

The quantum computer is currently in its infancy. As of 2008 the largest calculation such a machine could execute was 3*5 = 15. The underlying principle behind the quantum computer involves Einstein's wave-particle duality. The quantum computer is not bound to the limits of transistors or the binary system architecture. Quantum Information Processing (QIP) uses atoms held in a magnetic field instead of transistors. These units are called qubits. QIP exploits the laws of quantum mechanics, and as a

consequence a single qubit can take infinitely many quantum states. It will be a great challenge to get hundreds and thousands of these atoms to act in unison and function correctly [56]. However, this then "allows for a much more powerful information platform than is possible with conventional components" [57]. It is more than just that a quantum computer would be faster, it is that in the realm of quantum physics a computer can solve the factorization or discrete log problem in polynomial time rather than exponential time [58].

The modern computer's design closely resembles the classic Turing machine. The Turing machine was developed by, the British computer scientist, Alan Turing in the 1930s. The Turning machine can effectively compute any function that is computable [59]. "They can have only finitely many states and they can only read and write one symbol at a time on a one-dimensional tape" [60]. This is how modern computers work, executing instruction after instruction, linearly on a CPU; even multicore processors work like this. The classic Turing machine can be thought of as having a single fixed state at any given time, the quantum machine on the other hand "has an internal wave function, which is a superposition of a combination of the possible basis states." Transforms of the wave function can then alter the entire set of states in a single operation [61].

Currently this is beyond our technological capabilities. Large-scale quantum computer hardware require each qubit to be extremely well isolated from the environment, yet must be precisely controlled using external fields. These problems are far from trivial [62]. However, "no fundamental physical principles are known that prohibit the building of large-scale and reliable quantum computers" [63].

In 1994 Peter Shor of Bell Labs showed that factoring problem and the discrete logarithm problems would not be an obstacle for a quantum computer [64]. Shor's algorithm attacks the problem of finding the period of a function in polynomial time, as opposed to exponential time. It uses quantum parallelism to produce a superposition of all the values of this function in a single step. It then uses a quantum Fourier transform and measuring the yields, gives the period. This is then used to factor [65].

If the quantum computer is realized, the public key cryptosystem is broke. Furthermore, electrical power grid equipment installed in the field with public key systems in their firmware will be vulnerable to attack. Such an event could happen essentially over night, yet replacing electrical power infrastructure in critical and remote locations would be a very slow and expensive process.

IV. Proposed Solution

Public key algorithms use expensive modular arithmetic, exponential operations and are therefore not good fits for mesh clients [66]. An alternative to the use of resource hungry, quantum computer vulnerable public key authentication is a system based on Merkle trees. It is well known that hash based algorithms like MD5 and SHA-2 are computationally less expensive than symmetric key algorithms, which in turn are computationally less expensive that public key algorithms.

Popular cryptographic hash functions like SHA-1 or MD5 work much like block ciphers. That is they take plain texts and split them into fixed sized blocks then iterated by way of a function for some number of rounds [67]. They are considered secure if no collisions have been found; SHA-1 was broken about 10 years ago [68]. They must be fast, and have the effect that small changes to the input result in large changes in the output. This is known as the *avalanche effect* [69].

Merkle trees offer low cost authentication for mesh clients. Compared to public key, they are lightweight and quick to generate and offer the same, and in the case of quantum computer attacks, better security [70].

The strength of the Merkle tree authentication scheme rests on having a secure hash function, and practical cryptographic hash functions do exists. The purpose of a hash function is to produce a "fingerprint" of message, that is, a hash function *s()* is applied to a file *M* and produces *s(M)*, which identifies *M*, but is much smaller [71]. A cryptographic hash function must provide:

1. *Compression:* The input file can be of any size, but the output must always be the same size.

2. *Efficiency*: It must be relatively easy for the computer to compute the output.

3. *One-Way:* Given only *y* of $y = s(x)$, it must be computationally infeasible to compute x.

4. *Weak Collision Resistance:* It is not feasible to modify a message without changing its hash value. That is, given *x* and *s(x)* to find any *y*, with $y \neq x$ and $s(x) = s(y)$ is infeasible.

5. *Strong Collision Resistance:* We cannot find any two inputs that produce the same hash output. That is, it is infeasible to find any *x* and *y*, such that $y \neq x$ and $s(x) = s(y)$.

The last item here refers to how resistant the hash function is to a class of attacks known as the birthday attack.

A. THE BIRTHDAY ATTACK

The birthday paradox is a classic topic in probability, the result being that with only 23 people in a room you have a better than 50% chance of having two people with the same birthday. The paradoxical part of this problem is that at first glance it would appear 23 people are too few. A back of the envelope calculation however shows this is actually reasonable. The number of comparisons required with *n* people in a room will be:

$$n(n-1)/2 \approx n^2$$

Now just working with n^2 we know there are 365 days in a non-leap year and we get the following:

$$n^2 = 365$$
$$n = \sqrt{365} \approx 19$$

Appling this to hash functions, if we have *s(x)* that has an output with *n* bits, then there are 2^n different possible hash values—all values being equally likely. Since

$$\sqrt{2^n} = 2^{n/2},$$

then by the birthday problem we can expect to have a collision after $2^{n/2}$ different inputs. As a consequence to prevent this sort of attack, n must be substantially large that a brute force attack is not reasonable [72].

The goal of the birthday attack on a hash function is not to find a message x such that $s(x) = s(y)$, but rather it is to find two random messages x and y such that $s(x) = s(y)$ [73]. "The strength of a hash function against brute-force attack depends solely on the length of the hash code produced by the algorithm" [74]. This is a key point, if a quantum computer is realized to defend against a known attack the hash code only needs to be increased in length.

B. MERKLE TREES

A Merkle tree is a complete binary tree constructed from a set of secret leaf tokens, where each internal node of the tree is a concatenation then a hash of its left and right child. The leaves consist of a set of m randomly generated secret tokens. Since it is a complete binary tree $m = 2^h$ where h is the height of the tree. The root is public and the result of recursive applications of the one-way hash function on the tree, starting at the leaves [75].

The example below is of an $m = 8$ size Merkle tree. This tree then has 8 one-time authentication tokens to offer.

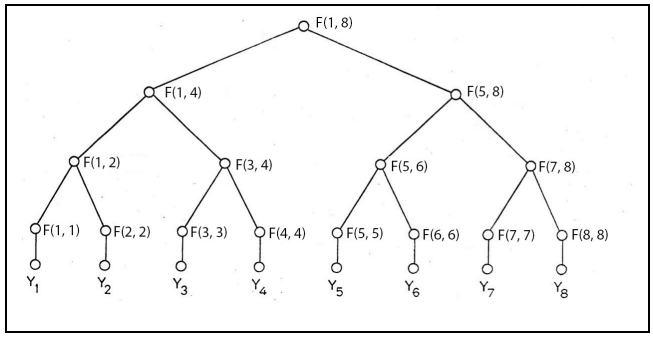


Fig. 3

In a mesh application the client generates the tree, and the root of the tree is made public. Thereafter, the client can prove its identity to any Mesh Router by comparing the published root, against the root that is generated when the hash function and authentication path is provided. It is computationally infeasible to determine the secret token from the published root of the tree [76]. This method is called "tree authentication" [77].

For example, if we wanted to authenticate using leaf Y_5.

Let F be a mapping function that we define by:

$$F(i,j) = s(Y_i)$$

$$F(i, j) = s(F(i,k), F(k+1), j))$$

where $k = (i+j)/2$

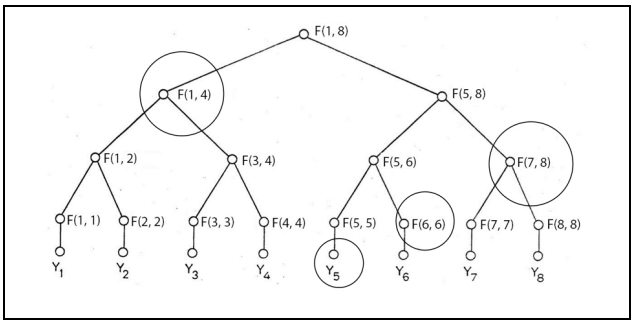
1. $F(1,8)$ is the root and already known by the receiver
2. Requester sends $F(1, 4)$ and $F(5, 8)$ and the receiver computes: $s(F(1, 4), F(5, 8)) = F(1, 8)$
3. Requester sends $F(5, 6)$ and $F(7, 8)$ and the receiver computes: $s(F(5, 6), F(7, 8)) = F(5, 8)$
4. Requester sends $F(5, 5)$ and $F(6, 6)$ and the receiver computes: $s(F(5, 5), F(6, 6)) = F(5, 6)$

5. Requester sends Y_5 and the receiver computes:

$$s(Y_5) = F(5, 5)$$

6. The receiver has now authenticated Y_5

Using this method, $log_2 n$ transmission are required to authenticate, however only half of the transmissions are actually required.



To recap, the client (requester) transmits to the receiver Y_i and the path to the root. The root is already known so no need to transmit that. The client is authenticated by the fact that the receiver is able to regenerate the value of the root based on the function *s()* and the path provided by the client [78].

Fig. 4

V. Complexity Analysis

A. Complexity Analysis of Merkle Trees

1. Computational Time Complexity

A Merkle Tree is a complete binary tree. The number of nodes at height h is 2^h. The height of the tree with n leaves is $log_2 n$. The number of internal nodes in a tree of height h is:

$$1 + 2 + 2^2 + \ldots + 2^{h-1} = \sum_{i=0}^{h-1} 2i$$

$$= (2^h - 1)/(2-1)$$

Therefore, there are $(2^h - 1)$ internal nodes [79].

Then to build a Merkle tree in each node we have an asymptotic upper bound of $O(2^h)$ with additional cost for the hash function. For our experiment we are using a non-cryptographic hash, which is faster than a cryptographically secure function. C++ uses *MurmurHashNeutral2* as its hash function. *MurmurHashNeutral2* uses a "Merkle-Damgard-esque" construction for its hash [80]. This uses a padding scheme on the front end to make sure all input into the compression function is of the same length. The input is broken into blocks, which are then compressed. The compression involves taken the result so far and combines with the next block. Cryptographic hash functions like SHA-1, SHA-2, MD5 all work this way [81]. So we can say that asymptotically *MurmurHashNeutral2* and SHA-1 are $O(ß)$.

The tree is not a sorted tree, so looking up items is not very quick, however that is not a priority. We are concerned with releasing leaves and returning paths for authentication. But obviously, the more leaves, the larger the tree, the longer to build, the longer to traverse. Our time to authenticate then is bounded by the height of the tree and the hash function, i.e., $O(h)$.

Merkle trees, as well as RSA, we are concerned with the time to build and the time to authenticate. Cryptographic hash functions work like block ciphers in that text is take, broken into blocks, then processed in some number of rounds. The amount of time required to process the input text is proportional to its size, the size chosen for this model is 32-bits, in real life this size would have be much bigger to be secure. The total time to build the tree is proportional to the size of the Merkle tree.

2. Memory Complexity

For our tests we start with a small tree of depth 4. That gives $2^4 = 16$ leaves (secret leaf tokens) and $2^4 - 1 = 15$ internal nodes (non-leaf nodes). Since our key is length 32, every node has a 32-bit key. These keys are concatenated and hashed and that new value is also kept at 32-bits.

MEMORY COMPLEXITY GROWTH				
		KEY LENGTH (BITS)		
		32	64	128
DEPTH	4	992	1,984	3,968
	8	16,352	32,704	65,408
	16	4,194,272	8,388,544	16,777,088

Chart 1

So in our example we would have 31 nodes with 32 bits in each node for keys, then we have 992 bits for keys, per node. Additional memory will be required for links and other data structure information. The amount of memory a Merkle tree will require then is proportional to the size of the tree, and the length of the key. We can see in Chart 1 how these interact.

Then our upper bound is based on the size of the tree and the size of the key, so

$$(2^h + 1) * \beta = O(\beta\, 2^h).$$

3. Message Complexity

Once the tree has been build we need to calculate the amount of time to authenticate. Again this is proportional to the size of our tree. In our starting case of depth 4 we would have with 16 leaves that would require $\log_2 16 = 4$ transmissions.
As shown in previous sections the Merkle tree sends an authenticating path back to the request. However instead of sending n transmissions, it can all be done in a single "packet" that contains all values in that path, organized in a manner that can be recomputed. This is done in our simulation with an array. So instead of $log_2 n$ transmission required there are $log_2 n$ entries into the packet array, obviously the payloads in these packets will be bigger than if they were sent one at a time. Still this approach helps to improve communication overhead and delays, in that there are simply less communications required. So we will have three message exchanges. However we should be concerned with the size of our key length, if too long that might not be advantageous to send such a large package. Chart 2, shows how this would increase depending on the size of the keys. We have an upper bound of $O(h\ \beta)$ for message complexity.

MESSAGE COMPLEXITY GROWTH				
		KEY LENGTH (BITS)		
		32	64	128
DEPTH	4	128	256	512
	8	256	512	1,024
	16	512	1, 024	2,048

Chart 2

B. Complexity Analysis of RSA

1. Computational Time Complexity

Public and private key generation by RSA relies on modular exponentiation. This is when an operation is raising one number to a power modulo another number. This is resource heavy—time, energy and processor resources. Assume the *public key: (N, e)* and *private key: d*, satisfy:

$$lg\ e = O(1),\ lg\ d \leq \beta \text{ and } lg\ N \leq \beta$$

Then applying a public key requires $O(1)$ modular multiplications and uses $O(\beta^2)$ bit operations. Applying a secret key requires $O(\beta)$ modular multiplications, using $O(\beta^3)$ bit operations [82].

2. Memory Complexity

In terms of memory consumption RSA does hold an advantage since it does not grow a large tree, one set of keys and it can authenticate with an unlimited number of devices. For each node they only need to hold their private key. Since public keys are public, the nodes do not need to retain that information. Therefore the amount of memory used in our test is 32 bits, the size of the private key. The upper bound then is $O(\beta)$.

3. Message Complexity

RSA works in three exchanges, like those show in Figure2. The size of the key in our case will be 32-bits. Of course in a real life situation the key would be much longer. The upper bound then is also $O(\beta)$.

C. Summary

Chart 3 compares the complexities per node.

COMPLEXITY COMPARISON			
		MERKLE TREE	**RSA**
COMPLEXITIES	BUILD TIME	$O(2^h) + O(\beta)$	$O(1) + O(\beta^2)$
	AUTH TIME	$O(h)$	$O(\beta) + O(\beta^3)$
	MEMORY	$O(\beta\ 2^h)$	$O(\beta)$
	MESSAGE	$O(h)$	$O(\beta)$

Chart 3

VI. Experiment Setup

In this section we look at our experiment. The mesh network provided with ns-3 required modifications in order to simulate an authentication routine. The simulation was run in Merkle Tree mode and RSA mode. This experiment was run on a MacBook Air running OS X 10.8.5, with a 1.8 GHz Intel Core i5 and 4 GB 1600MHz DDR3 of memory.

A. Wireless Mesh Network

ns-3 is a discrete event network simulator. It is widely used in industry and academia for the purposes of testing and evaluating networks. It is publicly available and comes with a number of different network types already set up. A mesh network is one of them. This mesh network, however, does not come set up with any functionality for authentication of nodes so that was added.

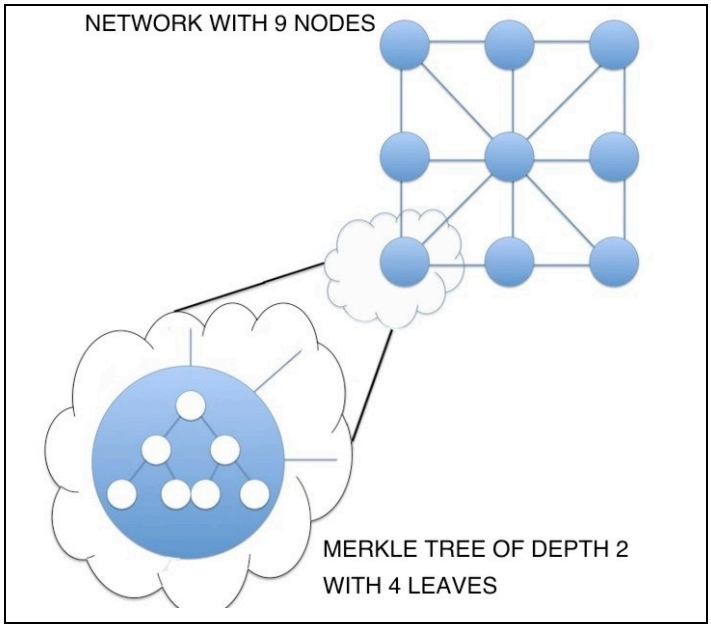


Fig. 5

Conceptually our network looks something like Figure 5, although with more nodes in the network and larger Merkle trees in the nodes. In this figure we see a Merkle tree with 4 leaves that can be used to authenticate, at this point it would have used up 3 and still have one more available.

In industry today we are starting to see utilization of WMN for lighting control systems. Currently these networks are limited to discrete sections of buildings, not entire buildings, and are often limited to no more than 64 devices. For this reason we defined this experiment to have a network of 64 nodes.

B. Merkle Trees

The Merkle Tree algorithm was coded as described in the previous sections. For the hash the *hash()* function available with the *tr1/functional* library of C++ was used. This hash is faster than a cryptographic one.

The Merkle trees were added into the existing ns-3 node structure, which represent devices in the WMN; these are defined in the *network/* section of ns-3. Here we added modules for creation and maintenance of Merkle tree. When a new node is created, a new Merkle tree is created and store in the node. Later when the nodes are being linked into a network, we add a functionality of authentication of nodes. This was done in the *mesh/model/dot11s* section of ns-3.

The initial assumption was that a Merkle tree with 16 leaves (depth of 4) would be sufficient. If we assume a network of 64 devices, then if every node can authenticate with 16 other nodes around it, that should be plenty to create a robust system. Also, since a bigger tree consumes more resources, there is an advantage to having a small tree; deep trees are no more secure than shallow. Our early tests showed that the Merkle scheme was much faster than the RSA scheme. Obviously if Merkle had used SHA-1 or another cryptographic hash function it would have slowed a bit, but still should have easily beaten RSA. However, if we consider RSA has no limit on the number of devices it can authenticate with then RSA does hold an advantage over Merkle tree in that sense.

C. RSA

The RSA model looks similar to the Merkle model shown in Figure 5, however instead of a tree in each node, RSA stores a private key. The public key is made public so there is no need for it to be stored in the node. The RSA software used was obtained from *rsa Project* available at [83]. This functionality was added in the same locations as the Merkle tree. For our experiment we toggled between the two so only one scheme was active during a simulation.

RSA can use the public/private key pair to authenticate itself with any number of other nodes; this is an advantage over Merkle trees. That is the Merkle tree scheme needs to know ahead of time how many nodes it will need to be able to authenticate with. RSA on the other hand can authenticate with as many devices that have access to its public key.

Most electrical systems today are relatively static in the sense that devices do not roaming form location to location. Also these networks tend to be very well defined prior to installation and are not required to dynamically expand by large leaps This makes our network simpler. We can accept that our device will not need to be able to authenticate with a million other devices.

RSA key generation calculations depend on the length of the keys. For the sake of this test we choose 32 bits. We also have the length of our Merkle root at 32 bits. Albeit this would not be secure in a real system, it is gives us good modeling data in a reasonable amount of time. We do test larger keys to see what impact the length has on calculation times.

VII. Experiment Results

So for our comparison we wanted to see how big of a Merkle tree we could build and authenticate with, in the time it takes RSA to build and authenticate. Times were measured during the construction of the node, then again during the linking of the nodes where the authentication process was performed.

A. Build Time

Figure 6 shows show built times.

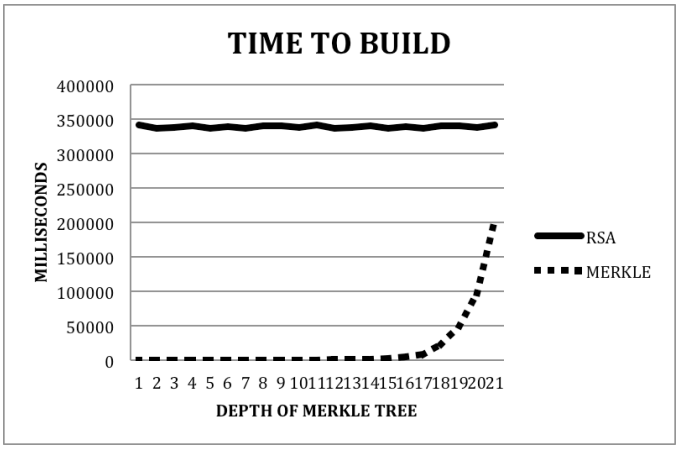


Fig. 6a

Figure 6b give a closer look at the time to build for the Merkle tree only.

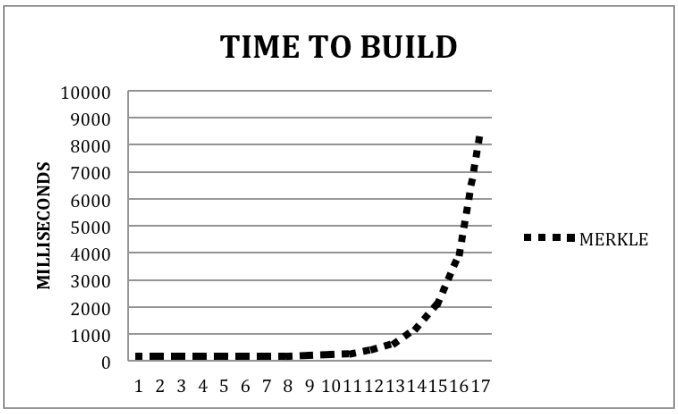


Fig. 6b

We can see from Figure 6 that RSA is very slow to build compared to Merkle, taking 35000 milliseconds to build. When compared to a Merkle tree of shallow depth, we see the Merkle scheme has a clear advantage. We do see the Merkle scheme slow as the tree grows larger. Around a depth of 16 we start to see noticeable slowing in the Merkle scheme. At a depth of 16 each node has 65,536 leaves to authenticate with. It was not possible to see at what depth the Merkle tree equaled RSA's time because the computer that was running the tests started to report memory problems, then seize up. At a depth of 25, which was the case where the computer started having trouble, we were building a Merkle tree with 33,554,432 leaves.

We attempted to build a network with an RSA key of 2048 bits. 2048 bits would be considered a secure length. This, however, proved to be an unattainable goal. That is, the largest reasonable size key that could be built on this system was with a key of 256 bits, which was still short of what would be considered secure. At 256, build time of the public-private key pair took over 22 minutes—for only 1 node. Building a network of 64 nodes all of them taking 22 minutes to build would have taken on the order of 1408 minutes, or about 23.4 hours. That gives us an idea of exactly how computationally expensive RSA is.

B. Authenticate Time

Figure 7a shows the amount of time to authenticate.

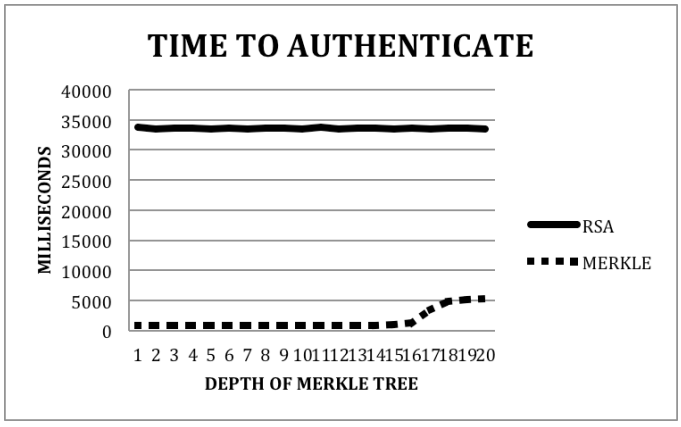


Fig. 7a

Figure 7b takes a closer look at only the Merkle scheme.

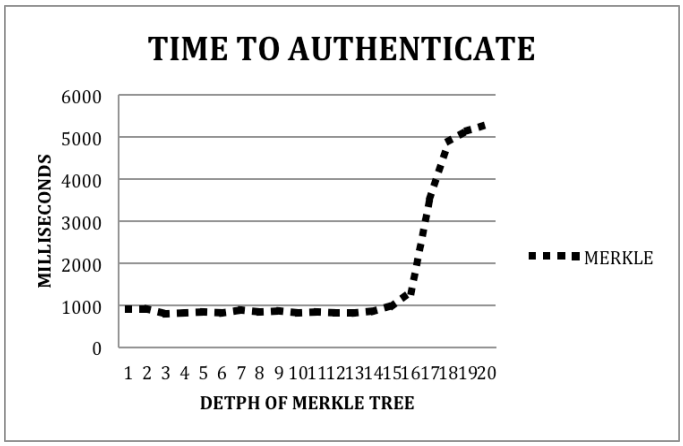


Fig. 7b

For the Merkle trees we can see from the data that the larger the tree the more traversing of the tree we need to do to provide our authentication path. Still Merkle continues to do better than RSA for authentication. In these plots RSA is using a 32-bit key, with the 256-bit key, RSA did well worse than the Merkle tree taking on the order of 3 minutes to authenticate one single node. This number would then be proportional to the size of the network and number of links. What we can see from all of this is that Merkle trees are a viable alternative to the use of public key for authentication.

VIII. Conclusion

This paper is meant to contribute to the discussion on going related to the creation of the Smart Grid. The development of public key cryptography some 40 years ago was revolutionary, and its cleaver solutions to problems have found wide spread usage. What we have shown in this work is that there is an alternative to public key crypto that is better when put in the context of future applications and future threats.

In the decades ahead the electrical power gird we have today will not be able to handle what we require of it due to the demands of our increasingly electricity driven societies. It has been widely reported that the current grid could not handle the load if every internal combustion car was swapped for an electric one [84]. Yet even before we consider cases like that, today we are still prone to expansive power outages that can last days, even weeks, and can affect millions of people. An important factor in the quality of life that we as a species will have in the future will depend on energy; how we get it, how we use it, how we distribute it. The Smart Grid is an important step toward a future with a quality of life better than the one we have today. The process of converting fully to the Smart Grid will take a generation, yet everything that is to come must be built on a solid foundation of information security.

REFERENCES

[1] Kathy Kowalenko, “The Smart Grid: A Primer”, *The Institute, IEEE*, p. 5, December 2010.

[2] NIST 7628, “Guidelines for Smart Grid Cyber Security”, p. 3, September 2010.

[3] U.S. Department of Commerce, “NIST Guidelines for Smart Grid Cyber Security, Volume 1”, p. 4, August 2010.

[4] Ye Yan, Yi Qian, Hamid Sharif, David Tipper, “A Survey on Cyber Security for Smart Grid Communications”, *Communication Surveys and Tutorials, IEEE,* Volume 14, issue 4, Section III, p. 4, January 2012.

[5] Lakshmi Santhanam, Bin Xie, Dharma Agrawal, “Secure and Efficient Authentication in Wireless Mesh Networks using Merkle Trees”, *33rd IEEE Conference on Local Computer Networks*, p. 967, 2008.

[6] U.S.-Canada Power System Outage Task Force, “Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations”, p. 1, April 2004.

[7] U.S.-Canada Power System Outage Task Force, “Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations”, p. 81, April 2004.

[8] U.S.-Canada Power System Outage Task Force, “Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations”, p. 45, April 2004.

[9] U.S.-Canada Power System Outage Task Force, “Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations”, p. 18, April 2004.

[10] NIST 7628, “Guidelines for Smart Grid Cyber Security”, p. 3, September 2010.

[11] Bruce Schneier, *Applied Cryptography*, p. 600, 1996

[12] U.S. Department of Commerce, “NIST Guidelines for Smart Grid Cyber Security, Volume 1”, p. 3, August 2010.

[13] U.S. Department of Commerce, “NIST Guidelines for Smart Grid Cyber Security, Volume 3”, p. viii, August 2010.

[14] Richard DeBasio, Chief Engineer U.S. Department of Energy’s National Renewable Energy Laboratory.

[15] IEEE-USA Board of Directors, “Building a Stronger and Smarter Electrical Energy Infrastructure”, pp. 6-9, February 2010.

[16] Di Wang, Zhifeng Tao, Jinyun Zhang, Abouzeid, A.A., "RPL Based Routing for Advanced Metering Infrastructure in Smart Grid," *2010 IEEE International Conference on Communications Workshops (ICC),* p.1, May 2010.

[17] U.S. Department of Commerce, "NIST Guidelines for Smart Grid Cyber Security, Volume 3", p. 24, August 2010.

[18] Muhammad Shoaib Siddiqui, Choong Seon Huong, "Security Issues in Wireless Mesh Networks", *MUE '07 International Conference on Multimedia and Ubiquitous Engineering*, p. 717, April 2007.

[19] Shahin Farahani, *ZigBee Wireless Networks and Transceivers*, p. 90, 2008.

[20] Larry Peterson, Bruce Davie, *Computer Networks*, p.134, 2012.

[21] Muhammad Shoaib Siddiqui, Choong Seon Huong, "Security Issues in Wireless Mesh Networks", *MUE '07 International Conference on Multimedia and Ubiquitous Engineering*, p. 720, April 2007.

[22] Andre Egners, Ulrike Meyer, "Wireless Mesh Network Security: State of Affairs", 2010 IEEE 35th Conference on Local Computer Networks (LCN), p. 997, October 2010.

[23] Muhammad Shoaib Siddiqui, Choong Seon Huong, "Security Issues in Wireless Mesh Networks", *MUE '07 International Conference on Multimedia and Ubiquitous Engineering*, p. 718, April 2007.

[24] Muhammad Shoaib Siddiqui, Choong Seon Huong, "Security Issues in Wireless Mesh Networks", *MUE '07 International Conference on Multimedia and Ubiquitous Engineering*, p. 719, April 2007.

[25] Tyler Wrightson, *Wireless Network Security*, p.42, 2012.

[26] Ji-Sun Jung, Keun-Woo Lim, Jae-Beom Kim, et al., "Improving IEEE802.11s Wireless Mesh Networks for Reliable Routing in the Smart Grid Infrastructure", *2011 IEEE International Conference on Communications Workshops (ICC),* p. 1, 2011.

[27] ZigBee Specifications 053474r17, Jan 2008

[28] Shahin Farahani, *ZigBee Wireless Networks and Transceivers*, p. 1, 2008.

[29] Shahin Farahani, *ZigBee Wireless Networks and Transceivers*, p. 17-22, 2008.

[30] Josh Wright, "ZigBee Hacking and the Kinetic World", http://www.youtube.com/watch?v=BkVcElfOVyw, 2010.

[31] Shahin Farahani, *ZigBee Wireless Networks and Transceivers*, p. 25, 2008.

[32] https://www.certicom.com/index.php/device-authentication-service/smart-energy-device-certificate-service

[33] http://en.wikipedia.org/wiki/Electromagnetic_interference_at_2.4_GHz

[34] Texas Instruments CC2530 specifications

[35] Stamp, Low, *Applied Cryptanalysis: Breaking Ciphers in the Real World*, p.84, 2007.

[36] http://travisgoodspeed.blogspot.com/2009/12/prng-vulnerability-of-z-stack-zigbee.html

[37] http://travisgoodspeed.blogspot.com/2009/12/prng-vulnerability-of-z-stack-zigbee.html

[38] NISTIR 7628 "Guidelines for Smart Grid Cyber Security" v1.0—Aug 2010, p. 222

[39] Michael Mitzenmacher, Eli Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*, p. 225, 2005.

[40] Kelsey, Schneier, et al., "Cryptanalytic Attacks on Pseudorandom Number Generators", *Fast Software Encryption, Fifth International Workshop Proceedings*, p.169, 1998.

[41] Bruce Schneier, *Applied Cryptography*, p. 413, 1996.

[42] Bruce Schneier, *Applied Cryptography*, p. 380, 1996

[43] Jordan Robertson, *Smart Meter flaws give hackers power*, SFgate, 27 Mar 2010

[44] Muhammad Shoaib Siddiqui, Choong Seon Huong, "Security Issues in Wireless Mesh Networks", *MUE '07 International Conference on Multimedia and Ubiquitous Engineering*, p. 719, April 2007.

[45] U.S. Department of Commerce, "NIST Guidelines for Smart Grid Cyber Security, Volume 1", p. 210, August 2010

[46] Mark Stamp, *Information Security Principles and Practices*, p. 90, 2011.

[47] Mark Stamp, *Information Security Principles and Practices*, p. 324, 2011.

[48] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction to Algorithms*, p.975, 2009.

[49] Bruce Schneier, *Applied Cryptography*, p. 467-468, 1996

[50] Mark Stamp, *Information Security Principles and Practices*, p. 96, 2011.

[51] Peter Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM Journal of Computing 26, p. 15, 1997.

[52] U.S. Department of Commerce, "NIST Guidelines for Smart Grid Cyber Security, Volume 3", p. 66, August 2010.

[53] Andrew Tanenbaum, *Structured Computer Organization*, p.18, 1990.

[54] Jelena Stajic, "The Future of Quantum Information Processing", *Science*, p.1163, March 2013.

[55] Eleanor Rieffel, Wolfgang Polak, *Quantum Computing, A Gentle Introduction*, p. 172, 2011.

[56] Michio Kaku, *Physics of the Impossible*, p. 69, 2008.

[57] Jelena Stajic, "The Future of Quantum Information Processing", *Science*, p.1163, March 2013.

[58] Peter Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM Journal of Computing 26, p. 4, 1997.

[59] Dexter Kozen, *Automata and Computability*, p. 206, 1997.

[60] Kenneth Rosen, *Discrete Mathematics* and its Applications, p. 700, 1995.

[61] Bruce Schneier, *Applied Cryptography*, p. 164, 1996

[62] C. Monroe, J. Kim, "Scaling the Ion Trap Quantum Processor", *Science*, p. 1164, March 2013.

[63] Eleanor Rieffel, Wolfgang Polak, *Quantum Computing, A Gentle Introduction*, p. 3, 2011.

[64] Michio Kaku, *Physics of the Impossible*, p. 68, 2008.

[65] Eleanor Rieffel, Wolfgang Polak, *Quantum Computing, A Gentle Introduction*, p. 163, 2011.

[66] Lakshmi Santhanam, Bin Xie, Dharma Agrawal, "Secure and Efficient Authentication in Wireless Mesh Networks using Merkle Trees", *33rd IEEE Conference on Local Computer Networks*, p. 967, 2008.

[67] Mark Stamp, *Information Security Principles and Practices*, p. 57, 2011.

[68] https://www.schneier.com/blog/archives/2005/02/sha1_broken.html

[69] Mark Stamp, *Information Security Principles and Practices*, p. 133, 2011.

[70] Wikipedia, http://en.wikipedia.org/wiki/Merkle_tree

[71] Mark Stamp, *Information Security Principles and Practices*, p. 126-127, 2011.

[72] Mark Stamp, *Information Security Principles and Practices*, p. 128-130, 2011.

[73] Bruce Schneier, *Applied Cryptography*, p. 429, 1996

[74] William Stallings, *Cryptography and Network Security*, p. 259, 1999.

[75] Lakshmi Santhanam, Bin Xie, Dharma Agrawal, "Secure and Efficient Authentication in Wireless Mesh Networks using Merkle Trees", *33rd IEEE Conference on Local Computer Networks*, p. 967, 2008.

[76] Lakshmi Santhanam, Bin Xie, Dharma Agrawal, "Secure and Efficient Authentication in Wireless Mesh Networks using Merkle Trees", *33rd IEEE Conference on Local Computer Networks*, p. 966, 2008.

[77] Ralph Merkle, "Secrecy Authentication and Public Key Systems", *Information Systems Laboratory, Stanford Electronics Laboratories*, Chapter V, p. 40, June 1979.

[78] Ralph Merkle, "Secrecy Authentication and Public Key Systems", *Information Systems Laboratory, Stanford Electronics Laboratories*, Chapter V, p. 40-43, June 1979.

[79] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction to Algorithms*, p.1179, 2009.

[80] https://sites.google.com/site/murmurhash/

[81] http://en.wikipedia.org/wiki/Merkle-Damgard_construction

[82] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction to Algorithms*, p.958-962, 2009.

[83] https://code.google.com/p/rsa/

[84] www.aolnews.com/2010/10/17/can-the-nations-grid-handle-a-surge-of-electric-cars