I am Joe's Fridge: Scalable Identity in the Internet of Things

Prashant Anantharaman*, Kartik Palani†, David Nicol†, and Sean W. Smith*

*Department of Computer Science, Dartmouth College, Hanover, USA
†Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, USA
Email : {pa,sws}@cs.dartmouth.edu,{palani2,dmnicol}@illinois.edu

***Abstract*—Most visions of the emerging Internet of Things (IoT) distribute massive numbers of devices throughout physical infrastructure. In most visions, these devices need to talk to each other. For this to work, the devices need to be able to authenticate each other. Presenting design issues and empirical results, this paper explores the cryptographic infrastructure necessary to capture this identity ontology.**

I. Introduction

In standard visions of the Internet of Things, there are a massive number of things talking to each other. For this talking to be meaningful, the listeners need to know who the talkers are.

Was it really my smart phone app that just asked my front door to unlock and the heat to turn on (e.g. [13])? Was it really my cars ECU that just told my car's brakes to engage—or was it an impostor (e.g., [15])? Was it really Google Calendar that just asked my smart refrigerator for my password—or was it an impostor (e.g., [20])? Was it really my washing machine that just told my utility company that the machine is using a less power-hungry washing algorithm? Thanks to the permeable nature of networked communication, impersonation is always a concern.

For a representative example, consider the smart home—an exciting environment where consumer IoT meets the smart grid. However, this interaction between the two elicits the need for stronger security protocols that can protect the safety critical grid infrastructure.

Smart Home Appliances are widely distributed, and given their intimate connection to reality, their impersonation can sometimes turn out to be very lucrative (consider ransomware for the home)—or at the same time have devastating consequences.

With over 50 million AMI meters installed across the US, the smart meter is one of the most pervasive smart devices along with the smart phone. In the vision of the consumer-side smart grid, every house will have a smart meter that communicates with other smart appliances in the house to make users more energy aware and energy efficient.

- Meters could have knowledge of occupancy of a particular home, and turn off particular appliances to save electricity.
- The smart meter would receive real-time pricing information, and could then relay pricing signals to the appliances.
- The smart meter could communicate demand response signals to these appliances as well. Examples would include switching off air conditioning for 20 minutes or reducing the heat by 5 °F over the next hour to help ease stress on the grid.
- Smart meters communicate with certain gateway devices like Bidgely [8] to help users get more detailed bills.
- A smart appliance would receive software updates from its manufacturer.
- A smart appliance would want to send repair diagnostics to the manufacturer, to aid in quick fixing of the appliance.
- A smart meter could help coordinate charging of an electric vehicle with other local usage—and perhaps even use the battery to store power.

This communication link between the smart meters and the smart devices in a home increases the attack surface of the grid—and can extend the reach of such attacks. Effective identification and authentication on these communication links is an important step toward mitigating this increased risk.

The embedded systems in the IoT will likely have constrained computational power and memory, and (for systems not connected to the wall) may have constrained electrical power as well. The communication channels between them may also be constrained. Thus, we need to consider the engineering impacts of the supporting cryptographic technologies. For example, in prior work [36], we found that adding cryptographic authentication to the BGP routing protocol—only only a few tens of thousands of entities—kills performance. In [31], we considered some issues for the smart grid alone. Also, as we will see in Section IV, popular but complex cryptographic algorithms may take unacceptably long to run on these constrained devices, forcing the need for more lightweight cryptography.

Towards this end, our contributions are as follows:

- We provide an elaborate discussion of the namespace and cryptographic complexity in the consumer-side smart grid, as an example of the IoT.
- This work discusses two possible solutions to the rel-

atively less explored problem of namespace and cryptographic complexities, which could serve as a starting point for future work in this direction.

II. Namespace and Cryptographic Complexities

For secure interaction, devices in the smart grid need to know to whom they are talking. Unfortunately, we have seen in the incipient smart grid and IoT a rush to deployment which leaves glaring holes here. Several exploits have been demonstrated including home cameras sharing private images with anyone [30]; insurance dongles in cars accepting software updates from anyone [14]; a person maliciously changing the temperature in his ex-spouse's bedroom [23]; home alarm systems allowing anyone to intercept and alter alarm messages [10].

Making things even more complicated is that a globally unique name may not suffice for the listener. What attributes do listeners need to know? Who is in a position to witness these attributes? (When does it become Joe's washing machine?) When will the binding change? (What if Joe sells the car or moves out of the apartment?) How would this communication of naming and attribute data happen?

Attributes. For example, in the consumer side of smart grid, consider a basic smart appliance. If it's talking to an external party, the external party might need to know what general type of appliance it is, what its make and model are, who owns it, and the physical place it resides. The appliance may need to know who the external party is: the manufacturer? A duly authorized repair person? The utility currently providing electricity to that household? Google Calendar?

In scenarios where appliances talk to peer appliances, they need to know if this is really an appliance of a certain type, and also that the peering relationship exists (we are in the same household or perhaps even the same room together).

In scenarios where a smartphone or laptop controls devices, this controller needs to know it's talking to the right devices, and vice-versa.

In electric vehicle scenarios, a car and its charging infrastructure need to authenticate each other; the charging infrastructure needs to know whose car it is, for billing, for scheduling when the charge needs to be complete, and possibly to know whether the car battery can be used as back-up for the grid.

Again, in all these situations, a simple global name does not suffice to tell the relying party what they need to know.

Lifetimes. In some sense, we can formalize an attribute as tuple (P, O, Δ): property P holds for the object O (initially, the object at the other end of the communication), for time Δ (initially, from right now). Looking at the above discussion, we can see the need for many types of attributes in a smart home.

Some are basic identity: this P will always bind to this O. This will always be this specific TPMS, or this specific ECU, or this refrigerator made by GE. Someone present at the birth of this device would be in a position to assert the binding of P to O, and the binding would hold for the life of the device.

However, other attributes depend on a more dynamic "ontology of association." How did this device come to be in this household? How did these two devices come to be in the same household? Does an appliance (or a human) change households? With what utility did the user contract? Has the tire been removed—and perhaps even sold to someone else? The likely witnesses here may even more distributed and varied—and the time interval in which the P, O binding holds may be much less than the lifetime of the device.

This latter case opens up another can of worms: how does a relying party know this witness is in a position to make this assertion?

III. Proposed Architectures

To explore scalability, we will thus consider two kinds of identity for each entity:

- Core identity — This would tell us that an appliance is of a particular type (e.g., washing machine of type x).
- Association attribute — This would tell us who or what an appliance is associated with (e.g., washing machine in Bob's apartment).

We will need cryptographic tools and a trust calculus so that entities can prove they have possess such identities, entities generating these assertions about other entities' identities may themselves have supporting assertions testifying that they can do this, and relying parties will have some set of rules about how they can infer a conclusion given a set of assertions and some core axioms.

A natural approach (and one often suggested for the smart grid) is to use public key infrastructure (PKI). Each entity would have matching public key and private key, and an entity can issue a digitally signed *certificate* asserting something about the public key of another entity. In a basic scheme, a single *certificate authority (CA)* issues certificates, any relying party who knows the CA's public key (*trust root*) can verify a certificate, and a certificate holder proves it matches the certificate by doing something with the corresponding private key. This approach can also extend to chains of certificates (with appropriate inference rules for what these chains mean).

An interesting alternative approach is to use *Macaroons* [5]. A macaroon consists of a public part—a random nonce and a set of additional data elements called *caveats*—and a private part—the HMAC value generated with a symmetric key on the public part. The caveats enable complex assertions like "trust this as long as it satisfies these caveats". These caveats form the public part of the macaroon. Instead of a certificate, entities have the public part of the macaroon; instead of key pairs, entities have the private part of their macaroons. Macaroons can chain together; for example, an entity E_1 with public macaroon M_1 and secret K_1 can use K_1 to generate a macaroon M_2, K_2 for entity E_2, and so on. A relying party which knows the secret key at the root of a macaroon chain can derive the private parts of all the macaroons along the chain—and thus share a secret with holder of the final macaroon.

Figure 1 sketches these two approaches. We now explore using applying each one to the smart grid identity problem.

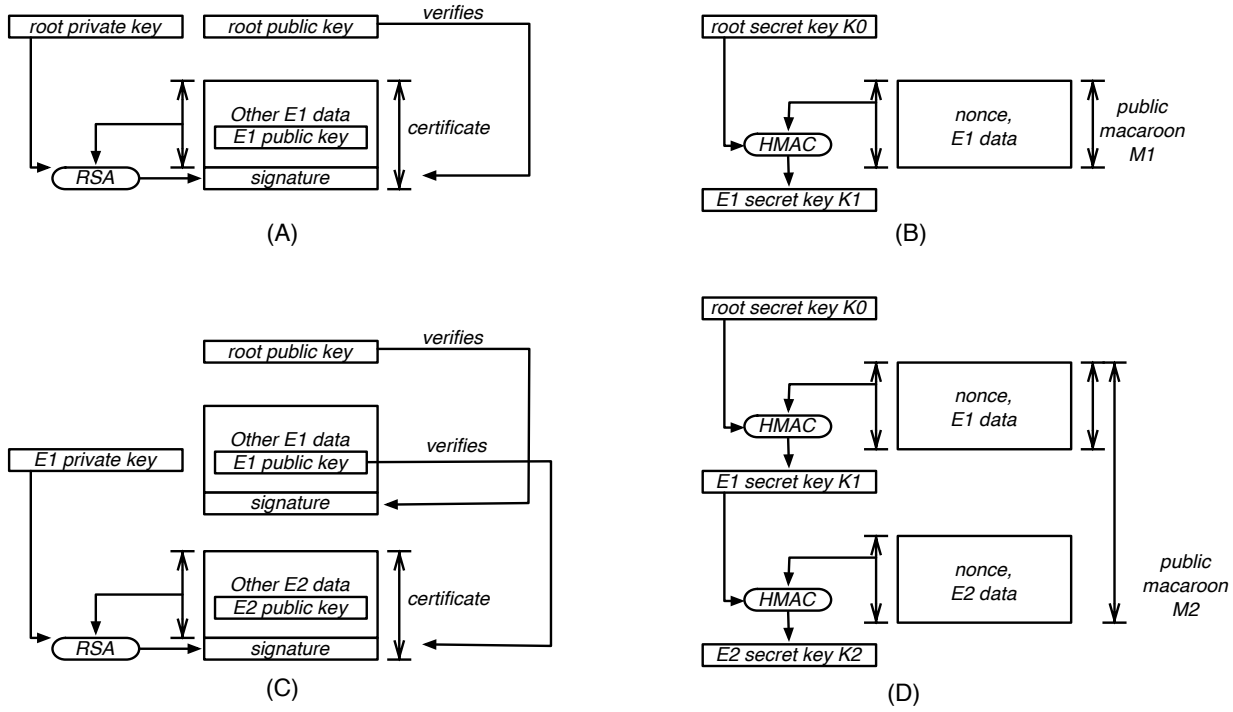

Fig. 1. (A) With PKI, a entity exhibits a certificate to prove its identity and uses its private key to authenticate itself to any relying party knowing the trust root public key. (B) With Macaroons, a entity exhibits a macaroon to prove its identity and uses this shared secret to authenticate itself to a relying party knowing the trust root secret key. (C,D): Both approaches can be chained.

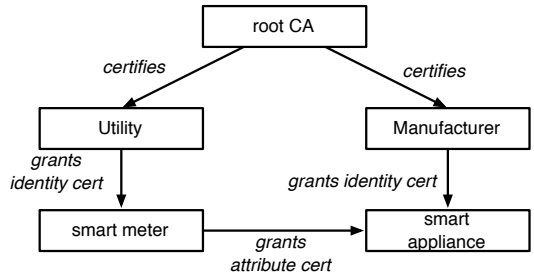

Fig. 2. Using identity and attribute PKI for smart appliances

A. Using PKI with Attribute Certificates

With PKI, we can have a trust root certify a CA at the utility and at each appliance manufacturer. The appliance manufacturer would issue an identity certificate to each appliance; the utility would issue an identity certificate to each meter. Each meter could then issue an X.509 *attribute certificate* [24], [7] to its co-located appliance, establishing the association. Figure 2 shows this trust flow.

1) Construction: **Device Registration.** When a device shows up in a house, it would present its identity certificate to the smart meter and prove knowledge of its private key. The smart meter checks the validity of the certificate, and then grants an attribute certificate which specifies that the device is associated with this specific meter.

2) Revocation: Every certificate that is issued has its validity limited by an expiry date. However, there are cases where there might be a need to *revoke* a certificate prior to the expiration date. Most revocations happen due to change in affiliation of the key holder, cessation of operation or private-key compromise. For trust to hold in PKI, it is important to make all nodes aware of these revocations, otherwise, relying parties may falsely conclude a binding exists when it does not. The most commonly used revocation schemes are the *Certificate Revocation List* (CRL), where the CA periodically publishes a list of revoked certificates to its clients, and the *Online Certificate Status Protocol* (OCSP), where an online check is made in real time with a trusted entity. The continually growing size of CRLs has made them hard to read at the client side and this will only get worse with a large population the size of IoT and Smart Grid as revocations get more frequent. OCSP, on the other hand, suffers from the fact that the trusted server is many a times unavailable due to large number of requests. Other schemes like NOVOMODO [22] and Certificate Revocation Trees [17] exist but they don't scale even for the Internet size population. Many certificate revocation schemes have been proposed, however, in the Internet today, most major web browsers fail to check for revocation status. When the heartbleed vulnerability was discovered, more than 80,000 certificates needed to be revoked. Imagine finding a vulnerability like heartbleed in a population the size of the Smart Grid and not checking for revocation.

3) Cryptographic Considerations: Public key cryptography is largely susceptible to brute force factoring attacks on keypair moduli. However, for moduli of at least 2048 bits (for RSA),

the amount of computation required to perform brute force and crack such large keys is not in the reach of attackers. (Unfortunately, constrained devices have been shown to have crackable private keys due to being generated with predictable randomness [16]—an orthogonal problem.)

We make use of RSA, DSA [18] and the Elliptic Curve Ed25519 [4] in our experiments.

B. Using Macaroons

With Macaroons, we can have a trust root create an *introduction macaroon* M_1, K_1 introducing a given utility's meters to a given appliance manufacturer. The utility would embed the introduction M_1, K_1 with each meter. The manufacturer would use K_1 to generate a *manufacturer's identity macaroon* M_2, K_2 for each appliance, establishing what kind of appliance it is. When an appliance arrives in a house and presents M_2 to the meter, they can set up a mutually authenticated channel using the shared secret K_2. (Using of HMACs instead of digital signatures requires the manufacturer know this introduction key K_1—however, the worst it can enable the manufacturer to compromise introduction of their own appliances).

If the utility also has a root secret R_U and uses that to give each meter a *utility's identity macaroon* M_3, K_3, then the meter can then issue its own macaroons. For example, after having authenticated appliance via the introduction and ManIdent macaroons, the meter can issue the appliance a M_4, K_4 specifying both the type and location of the appliance; the meter can also issue a pairwise introduction macaroon to pairs of appliances at the house so they can establish authenticated sessions between themselves.

Figure 3 shows these trust flows.

The lifetimes of these macaroons would correspond to the lifetime of the bindings to which they testify.

1) Construction: **Shared secret secure channel.** Since there is a macaroon HMAC value that is shared between two appliances, or an appliance and the smart meter, or the smart meter or any appliance and the manufacturer themselves, we can follow the NIST guidelines [9], and use a Key Derivation Function (KDF). This KDF could be used to generate a key for the purpose of the symmetric key channel. We identify the following methods could be used for this purpose - bcrypt (72 bytes) [27], scrypt (64 bytes) [25] and argon2 (68 bytes) [6]. The standard AES encryption uses 256 bit keys (32 bytes).

The key generated by the key derivation function will have to be truncated to 256 bits to be used in AES encryption or Simon/Speck Ciphers (which have been optimised for constrained devices [2]), which are both symmetric encryption schemes.

Integrity Check. In order to check the integrity of the messages being sent over the secure channel, we use an HMAC using the MD5 algorithm, and the macaroon secret K as the initial password to generate the HMAC.

2) Cryptographic Considerations: The parties generating the initial macaroon, which are namely the utility provider, and a central authority, generating keys K and L, need to keep this value a secret, and only they can verify the correctness of the macaroon completely.

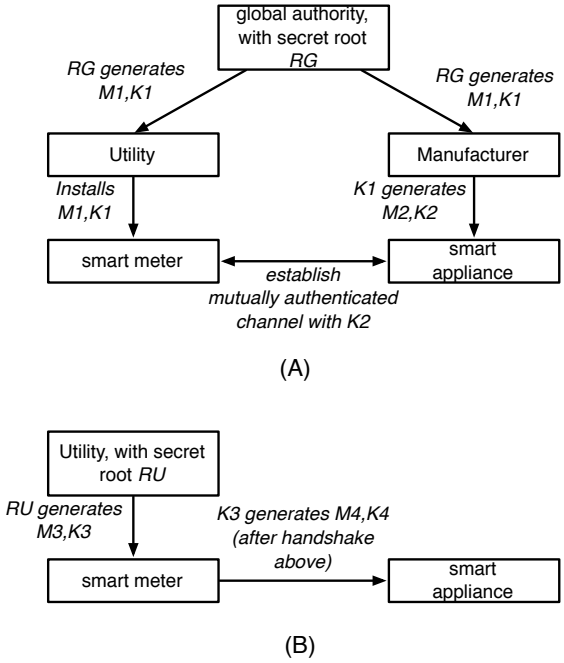

Fig. 3. (A) To use Macaroons to let a smart meter authenticate a particular type of smart appliance, a global root creates M1,K1 to let a manufacturers' devices introduce themselves to a utility's meters. (B) If the utility had also given the meter its own identity Macaroon beforehand, the meter can use that to to then issue a combined identity and association Macaroon to the appliance.

The security of the algorithm largely depends on the security of the HMAC algorithms itself. The strongest attacks against HMAC are based on the frequency of collisions, "birthday attack" and the "timing attack" could be used for improperly secured systems. The birthday attack is impractical for reasonable hash functions [19]. Even the MD5 and SHA-1 hashes, with known lack of collision resistance, don't show any vulnerabilities when used as a message authentication code [34][3].

3) Revocation: Short lived certificates, as proposed in [28], are issued with lifespans over a few days unlike conventional X.509 certificates that are valid for a year or two and need to be revoked by the CA. The bandwidth required to reissue a certificate is significantly lesser than the requirement of disseminating Certificate Revocation Lists. Short lived certificates form the basis of revocation in macaroons.

Macaroons contain a validity caveat in them, which is implemented using epoch counters, beyond which the macaroon would no longer be valid.

The utility provider minting these macaroons, signs the macaroon with the validity caveat set to a lifespan of a few days. This macaroon is then granted to the smart meters. The smart meter could further attenuate the macaroons and transfer them to other appliances in the house.

Once the macaroon expires, the validity caveat makes the macaroon unusable. Thus, a device presenting an expired

macaroon is no longer trusted.

Unlike Certificate Revocation Lists, where the utility, the smart meter and the appliances would need to check against a blacklist of invalid certificates, there is no need to maintain such state in a macaroon based implementation. This makes it a memory and network friendly replacement.

In our next section we discuss the performance of each of the above algorithms. We also discuss the performance of attribute certificate based algorithms and the macaroons in constrained devices.

IV. Results and Discussion

In this section we discuss the performance of the smart meter in various scenarios.

The major difference between the Macaroons and the PKI-Attribute Certificate based scenario is the fact that the PKI based Attribute Certificates make use of asymmetric cryptography, where as Macaroons make use of HMAC. Intuitively, we would think that a HMAC based scheme should take much lesser time than an asymmetric scheme. We will discuss the performance further in this section.

We are concerned about testing the methods $createAttrCert$ and $verifyAttrCert$ methods of the PKI based model and the $createMacaroon$ and $verifyMacaroon$ methods of the Macaroon model.

The $createAttrCert$ method which computes a hash of the certificate contents, and then creates a signature using RSA, DSA or Elliptic Curve Ed25519. The $verifyAttrCert$ algorithm verifies the signature and the hash for its correctness.

The $createMacaroon$ method mints a macaroon by performing a sequence of nested HMACs using one of the many hash functions widely available and the $verifyMacaroon$ performs a similar operation and verifies the final HMAC value of the macaroon by performing the same operation.

The certificates and macaroons are minted at the initial server which has a good computational capability, and at a constrained device such as a smart meter. In the past our project colleagues have developed a smart meter research platform [11] in order to obtain realistic results for experiments pertaining to the Advanced Metering Infrastructure. The platform is an embedded system that uses a metering IC for power readings and a front-end microcontroller (TI MSP430) which runs applications for the smart meter. The platform also has a zigbee rf module for communication between other such smart meters and to the local data collector unit. The micro-controllers and system design used to build the platform are commonly used in commercial meters that are deployed worldwide. Figure 4 shows the smart meter research platform. We have modified the firmware on the smart meter research platform to incorporate the macaroons model. We also run our tests on a *Raspberry Pi 2* and a *GNU/Linux Server : Processor of Intel(R) Xeon(R) CPU 5160 @ 3.00GHz and 1GB of RAM*.

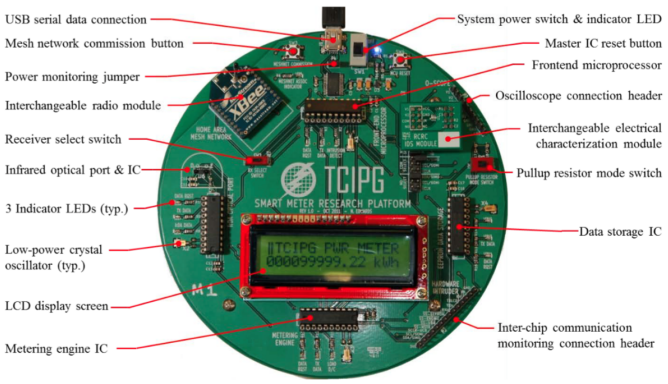

Fig. 4. Our Smart Meter Research Platform [11].

We make use of the *NaCl: Networking and Cryptography* and the *PyCrypto* libraries in python in order to perform these experiments.

We first test our PKI based scheme by running the $createAttrCert$ and $verifyAttrCert$ methods on the above mentioned server and raspberry pi. In Table I and Table II we can see that the time taken for the tasks of creating and verifying attribute certificates. We vary the key-size and the algorithm in order to check for the time taken to generate the attribute certificates. A granting authority would have to generate a keypair, then sign the certificate. For our experiments, we use a public RSA exponent value of 65537 and vary the private modulus length. Elliptic Curve Ed25519 is a high speed signature algorithm, with fast key generation and small signatures of size 512 bits [4].

TABLE I. Varying RSA modulus length, Elliptic Curve Ed25519 key size and DSA key size for PKI Attribute Certificates on a Server.

Protocol	Key length	$createAttrCert$	$verifyAttrCert$
RSA	1024 bits	40.26 ms	0.10 ms
RSA	2048 bits	253.61 ms	0.40 ms
RSA	4096 bits	1635.65 ms	1.43 ms
DSA	512 bits	19 ms	100 μs
DSA	1024 bits	82 ms	310 μs
Ed25519	256 bits	197 μs	226 μs

TABLE II. Varying RSA modulus length, Elliptic Curve Ed25519 key size and DSA key size for PKI Attribute Certificates on a Raspberry Pi.

Protocol	Key length	$createAttrCert$	$verifyAttrCert$
RSA	1024 bits	4.85 s	1.91 ms
RSA	2048 bits	24.06 s	8.33 ms
RSA	4096 bits	189.07 s	30.91 ms
DSA	512 bits	1.01 s	7.86 ms
DSA	1024 bits	1.34 s	10.36 ms
Ed25519	256 bits	25.79 ms	29.34 ms

We then test our macaroons based model, by testing our methods to create and verify the macaroons. In a macaroon based model, the smart meter would have to add caveats and verify the macaroons it would receive. Hence, we test the macaroon based model on the server and raspberry pi for different cryptographic hash algorithms (MD-5, SHA-1 and SHA-256) for computing the HMACs.

TABLE III. Varying cryptographic hash functions for an implementation of Macaroons on a Server

Hash Algorithm	$createMacaroon$	$verifyMacaroon$
MD5	98 μs	79 μs
SHA-1	100 μs	80 μs
SHA-256	110 μs	85 μs

TABLE IV. Varying cryptographic hash functions for an implementation of Macaroons on constrained devices

Hash Algorithm	$createMacaroon$	$verifyMacaroon$
Raspberry Pi		
MD5	650 μs	473 μs
SHA-1	662 μs	513 μs
SHA-256	761 μs	566 μs
TCIPG research platform		
SHA-1	900 μs	780 μs
SHA-256	1.2 ms	870 μs

From these results, we can see that in the smart grid, it would be time consuming for the smart meter or any constrained device to be doing asymmetric key cryptography often. Macaroons take much less time in comparison. Moreover, the number of macaroons generated is much lesser than the number of attribute certificates, since the macaroons aren't generated by a single server for every smart home appliance. If we have n smart meters, and m smart home appliances, in a macaroon based scenario, the server would generate n macaroons, and each smart meter would generate m macaroons after attenuating them. The results are shown in the Table IV.

In our second set of experiments, we try to ascertain which would be the best shared secret algorithm to set up a secure channel between any two devices sharing a macaroon HMAC value. As discussed in the previous section, the methods available are using a key derivation function, to generate a key from the shared secret. These key derivation functions include the $bcrypt$, $scrypt$ and the $argon2$ algorithms.

The $argon2$ algorithm took — 0.024 s, the $scrypt$ algorithm took — 0.78 s and the $bcrypt$ algorithm took — 24.41 s when tested on a raspberry pi.

V. Related Work

Security in the Smart Grid. In the past, a PKI based scheme for the smart grid has been proposed [21]. PKI was thought as an improvement to a scheme where for each pair would have to have a shared key generated and pre-loaded. A key management scheme combining elliptic curve public key cryptography and symmetric key techniques have also been proposed [35]. The scheme is known to tolerant to attacks such as replay and man-in-the-middle attacks. However, none of these schemes address the issue of *namespace and cryptographic complexities*.

Symmetric Ciphers. Our schemes would rely on the use of symmetric ciphers to communicate after establishing the shared secret. In constrained devices, we would need algorithms that are not only fast, but also have lesser space requirements. AES seems to perform the best in terms of throughput [12]. Other algorithms like Present and Simon and Speck Family of Ciphers [2] fare better when it comes to the space constraints.

Proximity-based approaches. The *resurrection ducking* model in the seminal [33] describes secure transient association of a device with a series of owners in the absence of servers to do the authentication. The work was then extended to peer to peer communication [32]. In [1], we are introduced to the concept of bootstrapping trust between strangers using pre-authentication and location-limited channels. More recent approaches to the problem of bootstrapping trust make use of certificateless encryption schemes like in [29]. Wanda [26] introduces us to a mechanism of using a *wand* to impart information onto devices.

The resurrecting duckling security models, and the work that followed allows users to use physical proximity to assert an ontological association. The questions of how to represent identities and associate cryptographically (what we cover in our paper) is orthogonal.

VI. Conclusion

In the rush to deploy more and more smart devices performing interesting tasks, we must not overlook the plumbing required to be done before deploying them.

In this paper, we explored some of the identity issues for IoT devices. We proposed two possible schemes for reliable communication in the Internet of Things. We noted that a Macaroons-based scheme is expected to scale more reliably for the number of data points in the envisioned smart grid, by putting decentralization and symmetric key ciphers into practice—while still providing a lot of the flexibility of PKI-based schemes.

In scenarios requiring multiple dynamic assertions from multiple witnesses, for what population sizes and revocation patterns will standard X.509 stop working? For which of these latter situations will Macaroons suffice? And when and why would a macaroon based model fail? Will limited-capacity smart appliances have sufficient entropy to generate strong keys (we have not seen a good track record so far)?

Our future work would include answering much of the above questions.

Acknowledgment

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000780. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*. Citeseer, 2002.

[2] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers. The SIMON and SPECK lightweight block ciphers. In *Proceedings of the 52nd Annual Design Automation Conference*, page 175. ACM, 2015.

[3] M. Bellare. New proofs for nmac and hmac: Security without collision resistance. *Journal of Cryptology*, 28(4):844–878, 2015.

[4] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang. High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2(2):77–89, 2012.

[5] A. Birgisson, J. G. Politz, U. Erlingsson, A. Taly, M. Vrable, and M. Lentczner. Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud. In *NDSS*, 2014.

[6] A. Biryukov, D. Dinu, and D. Khovratovich. Fast and Tradeoff-Resilient Memory-Hard Functions for Cryptocurrencies and Password Hashing. *IACR Cryptology ePrint Archive*, 2015:430, 2015.

[7] D. Chadwick, A. Otenko, and E. Ball. Role-based access control with X.509 attribute certificates. *IEEE Internet Computing*, 7(2):62–69, Mar 2003.

[8] P. Chakravarty and A. Gupta. Impact of energy disaggregation on consumer behavior. *Behavior, Energy and Climate Change (BECC) Conference*, 2013.

[9] L. Chen. Recommendation for key derivation using pseudorandom functions. *NIST special publication*, 800:108, 2008.

[10] C. Cimpanu. RSI Videofied Security Alarm Protocol Flawed, Attackers Can Intercept Alarms. *Softpedia*, November 2015.

[11] N. Edwards. Hardware Intrusion Detection for Supply-Chain Threats to Critical Infrastructure Embedded System. *Masters Thesis, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign*, 2012.

[12] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel. A survey of lightweight-cryptography implementations. *IEEE Design & Test of Computers*, 6:522–533, 2007.

[13] E. Fernandes, J. Jung, and A. Prakash. Security Analysis of Emerging Smart Home Applications. In *Proceedings of the 37th IEEE Symposium on Security and Privacy*, May 2016.

[14] T. Fox-Brewster. Hacker Says Attacks On 'Insecure' Progressive Insurance Dongle In 2 Million US Cars Could Spawn Road Carnage. *Forbes*, January 2015.

[15] A. Greenberg. Hackers Remotely Kill a Jeep on the Highway—with Me In It. *Wired*, July 2015.

[16] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pages 205–220, 2012.

[17] P. C. Kocher. On certificate revocation and validation. In *Financial Cryptography*, pages 172–177, 1998.

[18] D. W. Kravitz. Digital signature algorithm, July 27 1993. US Patent 5,231,668.

[19] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, RFC Editor, February 1997.

[20] J. Leyden. Samsung smart fridge leaves Gmail logins open to attack. *The Register*, August 2015.

[21] A. R. Metke and R. L. Ekl. Security technology for smart grid networks. *Smart Grid, IEEE Transactions on*, 1(1):99–107, 2010.

[22] S. Micali. Scalable certificate validation and simplified PKI management. In *1st Annual PKI Research Workshop*, 2002.

[23] C. Neagle. Smart home hacking is easier than you think. *InfoWorld*, April 2015.

[24] T. Nykänen. Attribute certificates in x. 509. *Techn. Ber., Helsinki University of Technology*, 2000.

[25] C. Percival. Stronger key derivation via sequential memory-hard functions. *Self-published*, pages 1–16, 2009.

[26] T. J. Pierson, X. Liang, R. Peterson, and D. Kotz. Wanda: securely introducing mobile devices. InfoCom, 2016.

[27] N. Provos and D. Mazieres. A Future-Adaptable Password Scheme. In *USENIX Annual Technical Conference, FREENIX Track*, pages 81–91, 1999.

[28] R. L. Rivest. Can we eliminate certificate revocation lists? In *Financial Cryptography*, pages 178–183. Springer, 1998.

[29] S.-H. Seo, J. Won, S. Sultana, and E. Bertino. Effective key management in dynamic wireless sensor networks. *IEEE Transactions on Information Forensics and Security*, 10(2):371–383, 2015.

[30] M. Smith. Peeping into 73,000 unsecured security cameras thanks to default passwords. *Network World*, November 2014.

[31] S. W. Smith. Cryptographic scalability challenges in the smart grid (extended abstract). *Innovative Smart Grid Technologies, IEEE PES*, 0:1–3, 2012.

[32] F. Stajano. The resurrecting ducklingwhat next? In *International Workshop on Security Protocols*, pages 204–214. Springer, 2000.

[33] F. Stajano and R. Anderson. The resurrecting duckling: Security issues in ad-hoc wireless networks. 1999. In *Security Protocols, 7th International Workshop Proceedings, Lecture Notes in Computer Science, Springer-Verlag. http://www. cl. cam. ac. uk/fms27/duckling*.

[34] S. Turner and L. Chen. Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms. RFC 6151, RFC Editor, March 2011.

[35] D. Wu and C. Zhou. Fault-tolerant and scalable key management for smart grid. *Smart Grid, IEEE Transactions on*, 2(2):375–381, 2011.

[36] M. Zhao, S. W. Smith, and D. M. Nicol. Aggregated Path Authentication for Efficient BGP Security. In *Proceedings of the 12th ACM Conference on Computer and Communications Security*, pages 128–138, 2005.