IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 June 2022, accepted 16 July 2022, date of publication 21 July 2022, date of current version 26 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3192970

RESEARCH ARTICLE

Comparative Analysis of Energy Costs of Asymmetric vs Symmetric Encryption-Based Security Applications

BASEL HALAK[1], **YILDIRAN YILMAZ**[2], **AND DANIEL SHIU**[3]
[1]School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.
[2]Department of Computer Engineering, Recep Tayyip Erdogan University, 53100 Rize, Turkey
[3]Arqit Ltd., London SW1E 5BY, U.K.

Corresponding author: Basel Halak (basel.halak@soton.ac.uk)

ABSTRACT Public key algorithms are heavily used in many digital applications including key establishment schemes, secure messaging apps, and digital signature schemes in cryptocurrencies. Recent developments in the field of quantum computation have placed these algorithms at risk as they enable the implementation of more effective attacks to derive the secret key. Most notably Shor's algorithm exponentially speeds up solving the factoring, discrete logarithm (DLP), and elliptic-curve discrete logarithm (ECDLP) problems. To address this challenge, NIST has initiated a process to develop and standardize a new quantum-resistant public-key cryptographic algorithm. However, asymmetric encryption schemes are known to be computationally intensive, hence energy demanding. The proliferation of energy-constrained internet of things devices, combined with the need to adopt higher complexity quantum resilient cryptographic algorithms, makes it more challenging to continue to use public-key algorithms for all applications. One approach to address these challenges is to adopt symmetric key systems, which are known to be more energy-efficient and more resilient to quantum computers-based attacks. This work performs a comprehensive comparison of energy costs between asymmetric and symmetric key schemes. This comparison is performed using two methods. The first approach uses the energy cost of data usage (ECDU) metric to evaluate the global energy costs associated with internet data usage. It was found that the annual energy consumed by applications associated with public-key cryptography globally is sufficient to provide electricity for 1000 UK households for a year. The second method uses an experimental technique based on constructing a small-scale network of wireless embedded devices. This is subsequently used to compare two key establishment schemes, symmetric and asymmetric, which allows for comparing the computation and communication costs of each solution in a controlled environment, and more importantly estimating the energy consumed by each device participating in the protocol. Our results show that a 58% saving in global energy costs of public key-based applications can be achieved by adopting symmetric key systems. It was also found that a 20% reduction of the energy consumed by a wireless device during a key agreement protocol, can be achieved if symmetric key encryption is used.

INDEX TERMS Symmetric-key encryption, public-key cryptography, key exchange protocols, digital signatures, energy.

I. INTRODUCTION

Public-key cryptography refers to a set of encryption algorithms that rely on the use of a pair of keys. A private key must be kept secret and a public key that can be revealed to others, wherein knowledge of the public key does not undermine the secrecy of the private key. These algorithms are referred to as asymmetric encryption schemes as the key used for encryption is different from that used for decryption. This contrasts with symmetric encryption schemes wherein the same key is

The associate editor coordinating the review of this manuscript and approving it for publication was Mohamad Afendee Mohamed.

used for encryption and decryption. Historically, public-key algorithms have been developed to solve the key distribution problem for establishing secure communication channels. Nowadays, these algorithms are heavily used in multiple applications including web browsing, social networking apps, and cryptocurrencies. The security of public-key algorithms relies on the difficulty of solving a mathematical problem. The two most widely used algorithms are RSA (Rivest–Shamir–Adleman) [1] and elliptic curves-based systems [2]. The security of the former relies on the difficulty of factorizing larger integer numbers, while the latter is based on a discreet logarithm problem. Recent developments in the field of quantum computation have placed these algorithms at risk as they enable the implementation of more effective attacks to derive the secret key. Most notably Shor's algorithm exponentially speeds up solving the factoring, discrete logarithm (DLP), and elliptic-curve discrete logarithm (ECDLP) problems [3]. To address this challenge, NIST has initiated a process to develop and standardize a new quantum-resistant public-key cryptographic algorithm [4]. Another major challenge facing the use of public cryptographic systems is its relatively large power consumption that may exceed the energy budget in resources constrained devices such as those used on the internet of things applications [5], this challenge is likely to be aggravated by the potential introduction of post-quantum encryption systems [6]. One approach to address these challenges is to adopt symmetric key systems, which are known to be more energy-efficient and more resilient to quantum computers-based attacks [7]–[9]. The authors of [9] have developed a digital signature scheme based on symmetric security primitives, which has promising applications for post-quantum cryptocurrencies [10]. The authors of [7] have recommended that protocol designers should rethink the usage of heavyweight public key constructions compared to symmetric key-based mechanisms to reduce computation resources overheads assuming they have both the same security properties. On the other hand, the authors of [8] have investigated the use of a variant of Kerberos key distribution [11], based on the 128-bit AES encryption, for the establishment of keys in wireless sensor networks. More specifically, they have compared its energy costs with that of an authenticated version of the elliptic curve Diffie-Hellman key exchange. Their results have shown that a 45% reduction in energy can be achieved using symmetric algorithms due to their smaller computation overheads. Although promising, the estimation of this study was based on the total energy of the protocol, whereas in such systems the energy consumed by each node is a more accurate metric for assessing energy efficiency as it has a direct impact on battery life. *The above studies demonstrate it may be beneficial from an energy-saving viewpoint, to use security solutions, which are based on symmetric key algorithms, compared to those based on public-key systems. However, to the best of our knowledge, a comprehensive comparison of the energy efficiency of these schemes that cover all application scenarios has not been done before. As key establishment methods for the internet are being redesigned due to security concerns and applied to more lightweight and low-powered devices, it is appropriate to consider the impact of high computation, high bandwidth, and key establishment methods based on asymmetric cryptography with simpler alternatives.*

This work considers two application scenarios where energy efficiency is particularly important, namely, global energy costs and resource-constrained devices. More specifically this study will answer the following three questions:

1) What are the global energy costs of using public-key algorithms?

2) What would the above figure be if symmetric methods were to be used instead?

3) Will the adoption of symmetric key security schemes reduce the energy requirements of security protocols for resources constrained systems?

To address the first two questions, this work adopts a holistic approach to energy estimation that considers the costs of computation, communication, and other infrastructures associated with the transmission of data across the internet. The challenge, in this case, is that there is no single authoritative data source that can be relied on, therefore the purpose of this part is to obtain an approximate estimate within reasonable bounds rather than aiming for precise calculations, which are sufficient for the intended comparison. The advantage of this technique is that it allows us to develop a better understanding of the hidden, and typically significant costs, of internet data transactions, such as those incurred by data centers [12], which is vital to perform a fair comparison.

To address the third question, a wireless sensor network is constructed using embedded devices. Next, implementations of two agreement schemes, symmetric and asymmetric are developed. This was subsequently used to estimate the average energy consumed by each communicating node for each protocol run.

The contributions of this work are as follows:

1) *Provides a global analysis of aggregate energy costs of public-key cryptography and estimates the much could be saved by a switch to symmetric key methods. Our results show that a 58% saving in global energy costs of public key-based applications can be achieved by adopting symmetric key systems.*
2) *Develop an experimentally based analysis approach to compare energy overheads of key establishment schemes in resources-constrained environments. Our results show that a 20% reduction of the energy consumed by a wireless device during a key agreement protocol, can be achieved if symmetric key encryption is used.*
3) *It outlines outstanding challenges in this area and provides a detailed roadmap for future work*

The remainder of this paper is as follows. Section 2 estimates the global power consumption of the internet and derives the energy cost of data usage. Section 3 develops an estimate of the global energy costs of public-key cryptography schemes. Section 4 explores the potential energy savings if solutions based on symmetric key algorithms are to replace public key-based schemes. Section 5 discusses key findings. Conclusions are drawn in section 6.

II. THE ESTIMATION OF GLOBAL ENERGY COSTS OF INTERNET DATA USAGE

A. THE POWER CONSUMPTION OF THE INTERNET

The expanding use of information and communications technology (ICT) contributes heavily to the rising global demand for energy [13]. This trend is driven by a relentless increase in the number of applications of this technology and its proliferation in all areas of modern life. From emails and social media apps to data analytics and electronic cash. The number of internet users has already reached 4.66 Billion which is approximately 60% of the world population [14]. The question is how much energy is consumed by this technology annually. One of the earliest works in this area is [15] from 2011, which indicated that the power socket of the internet can be up to 143 GW, which is approximately 1-2% of the global demand, similar results have also been obtained in a follow-on study a year later [16]. However, the past decade has witnessed a major increase in demand for ICT technologies. These include a significant increase in the number of internet of things devices reaching 13.8 billion in 2021 [17]. In addition to a rise in the use of social networking apps, for example, the number of users of WhatsApp, the most popular global mobile messenger apps, is more than 2 billion worldwide [18]. And the growing use of electronic cash. A recent study has indicated Bitcoin network, one of many cryptocurrencies currently available, consumes 87.1 TWh of electrical energy annually, equaling a country like Belgium [19]. This section develops an updated estimate for the energy consumption of the internet based on the approach outlined in [16]. The latter study allows us to understand the contributions of different segments of the internet (end devices, transportation, and data centers). To perform this analysis, one needs first to understand the structure of the internet and its various constituents. The internet is a global network of interconnected systems, each of which consists of millions of networks (private, public, academic, business, and government), which are connected by a wide range of communication technologies (e.g., wireless, optical…). The later relies on transmission/reception of standardized data packets. Each node in this network transforms its data into packets that are transmitted individually, possibly through different routes. At the receiving end, the packets are reassembled in the right order based on the protocol specification. The structure of the internet consists of four main parts, end user devices, tier 1 points of presence (POP), including local carriers, internet service providers and small data centers, tier 2 POP (regional carriers and medium data centers, and tier 3 (network access points, national carriers, large data centers). Therefore, when two devices communicate on the internet, data packets will travel from end devices (e.g. a smart phone) through the different layers of the internet (tier 1, 2 and 3) POPs and back again, passing multiple network nodes, each of which requires energy for data procession and related overheads such as cooling and lighting [20]. There is also energy consumed to regenerate transmitted signals by repeaters to compensate for the degradation through the communication links (copper wires, fiber optic…). Therefor the computation of the energy required for each bit of internet traffic should include all equipment and related infrastructures. To achieve this, we will first estimate the wall socket power of the internet, using equation (1)

$$WSP = \sum\nolimits_{i=0}^{K} P_i * D_i * U_i * N_i \tag{1}$$

where:

- K: number of components (PC, servers, data centers…).
- P: Average wall socket for each component.
- D: Duty cycle or the percentage of use dedicated to internet activities.
- N: the estimated global counts for each component.

The wall power socket of each component is estimated based on the available literature. For a personal computer (75 W), smartphone (2 W), or tablet (7 W). For internet of thing devices, the estimates vary greatly depending on the application. Examples include 10mW for low-end sensors [21], (0.5W) for smart bulbs [22], and (3W) for more advanced applications such as smart speakers such as Amazon Echo [23]. An average of (0.5W) per device is considered in this study. The power consumption of servers ranges between (220W) for volume servers, (700 W) for medium-range servers to 10kW for high-end servers [24]. Tier 1 servers are likely to be at the low end of this scale. Tier 2 and tier 3 machines are typically in the mid of this range. In both cases, these are considered not user-facing and not part of the cloud infrastructure, so in this study, an average of (1KW) is used. Cloud servers are present in large data centers that are always on, these are a mixture of medium-range to high-end machines, so in this study, an average of (4KW) is considered. For routers the estimates vary depending on the configurations [25], an average estimate of (5KW) is considered in this work. The figures for Cell power and telecom switches are 3.3KW and 40KW on average [26]. The duty cycle is 50% for end devices and 100% for other parts of network infrastructures (i.e., always on). The power overheads are 100% cloud infrastructures and tier 3 servers [20] and 50% for other servers. The percentage of use dedicated to internet activities varies between different components and from time to time, therefore a maximum and a minimum value are used here to capture these variations. The number of devices in each category is approximated based on previous studies and available data. For example, the estimate for 2021 are as follows tablets 1.28 Billion [27], smartphones 3.8 Billion [28], PC 2 Billion [29], [30] and IoT devices 13.8 Billion [17], WI-FI, 500 millions [31]. Estimates for other parts of the internet infrastructures are calculated using data from previous studies [15], [16] taking

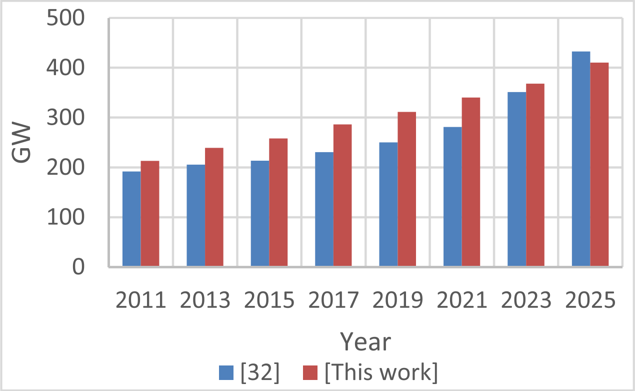

FIGURE 1. Estimation of the power socket of the internet.

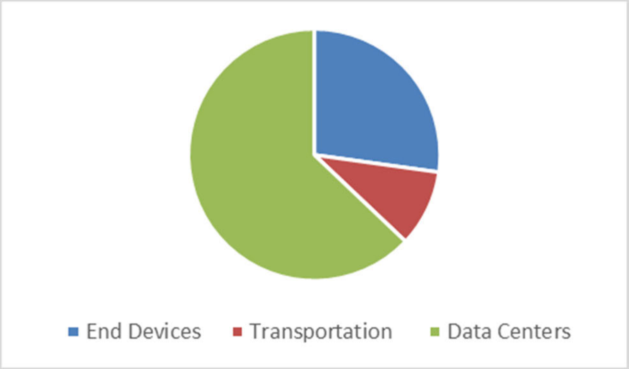

FIGURE 2. Segmented analysis of internet power consumption.

into consideration the 5-% annual growth rates of the internet infrastructure [31]. The results are shown in figure 1 and compared with the estimate from a previous study [32] for validation. A slight variation of the prediction of both studies is expected given the nature of these calculations. This work estimates the annual power consumption of the internet is approximately 12% of the global electricity consumption in 2021 compared to 10% estimated in [32].

Figure 2 shows a segmented analysis of this estimate for the year 2021 based on the classification presented in [16], wherein, end-user devices include personal computers, tablets, smartphones, and internet of things devices. Transportation infrastructures include Wi-Fi, LAN, cell towers, telecom switches, and signal and optical repeaters. Data centers include local and cloud servers and routers. The results show that the percentage of power consumed in each segment of the internet infrastructure is 27%, 10%, and 62%, which corresponds to end devices, transportation, and data centers respectively. This means most of the power consumed for data transactions over the internet is related to data centers and transportation infrastructures. Such costs are not typically included in studies that analyze the energy overheads of security protocols [12]–[33], which are limited to specific configurations. Therefore, such studies are not suitable for evaluating the energy efficiency of different approaches on a global scale. This work uses a metric called energy cost of data usage (ECDU) that takes into consideration these hidden costs, as will be shown in the next section.

B. THE ENERGY COSTS OF DATA USAGE

The energy cost of data usage is calculated as follows

$$ECDU = \frac{Annual\ Energy\ Consumption\ of\ the\ Intenet}{Annual\ Internet\ Data\ Usage} \tag{2}$$

Data usage on the internet is estimated to be 235.7 Exabytes per month [34] in 2021, equivalent to 2800 billion gigabytes (GB) per year. The annual energy consumption of the internet is calculated for the year 2021 based on the predicted power socket analysis presented in section 2.1. Based on these estimations, the ECDU in 2021 is calculated from equation (2) as *0.12 kWh per GB.* This figure will be used in the following sections to quantify the global energy costs of internet data usage for different security schemes.

III. GLOBAL ENERGY COSTS OF PUBLIC KEY CRYPTOGRAPHY APPLICATIONS

Asymmetric encryption algorithms have two major global applications, namely, key establishment schemes and digital signatures. The former is heavily used for secure communications on the World Wide Web, while the latter is the main building block of cryptocurrencies. The *ECDU* metric is going to be used in the following sections to estimate the annual global energy consumption of each of these applications.

A. GLOBAL ENERGY COSTS OF PUBLIC KEY-BASED KEY ESTABLISHMENT SCHEMES

These schemes include Diffie-Hellman's (DH) key agreement and its improvement, Menezes-Qu-Vanstone (MQV). The latter mitigates the risk of a malicious user stealing a private key and using it to masquerade as a third party to the user whose key was compromised. This study considers the two major applications of such methods, namely, secure web browsing and encrypted messaging apps, which allows for developing an estimate of the magnitude of the global energy costs of these techniques, as will be detailed below.

1) SECURE NETWORK COMMUNICATIONS USING TLS

Secure Sockets Layer (SSL) is a protocol based on a public-key algorithm, which was developed by Netscape, for providing data security layered between TCP/IP (the foundation of Internet-based communications) and application protocols (such as HTTPS). The socket refers to the interface between the session and the transport layer. The early versions of SSL however, had several security flows, so it went through several revisions, and it is now deprecated. The first version of its successor the Transport Layer Security (TLS 1.0) was published as part of RFC 2236 in 1999. Since then, it has then gone through several revisions, the last one is TLS 1.3, which was defined in RFC 8446 in 2018. TLS comprises

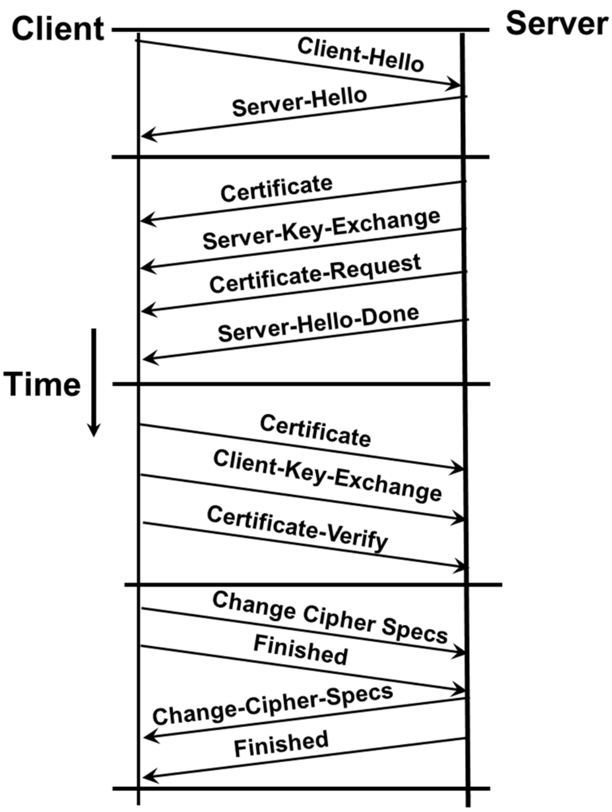

FIGURE 3. Overview of the TLS protocol.

two protocols referred to as Handshake and Record, respectively. TLS comprises two protocols referred to as Handshake and Record, respectively. The Handshake protocol typically employs public-key cryptography to establish a shared secret key between the client and the server, it is also used to provide mutual authentication. The Record protocol uses the secret key established in the handshake protocol to encrypt the communication between the client and the server. To estimate the energy cost of data usage, only the handshake protocol is relevant. The latter consists of four stages as shown in figure 3.

Phase-1 is to establish security capabilities, including session ID, protocol version, and cipher suits, and exchange initial random numbers, it comprises the following steps:

1) The client starts with a ClientHello message, which includes a list of client's preferences (highest TLS version supported by client, client random number including a timestamp, cipher suits supported by the client in order of preferences, and the session ID) nonzero means client wants to use an existing session state for a new connection state (abbreviated handshake); zero means new connection on a new session.
2) The server must respond with a server, hello, otherwise, the session would terminate, this includes (the TLS version proposed by the client is also supported by the server; otherwise, the highest version supported by the server, the server's random number including a timestamp, session ID, Cipher Suite selected from the client's list. At the end of phase 1, the two communicating parties should agree on a protocol version, a cipher suite, and a session id, in addition to the type of compression method to use.

Phase 2 of the protocol is used for server authentication and key exchange, wherein the server may send his certificate and a key exchange request, it may also request the client's certificate. It comprises the following steps:

1) The server sends its certificate that is used for server authentication by the client
2) The server may also send a key exchange request, which includes the server part of a Diffie-Hellman (DH) secret.
3) If client authentication is needed, an optional Certificate Request message is also sent
4) Finally, the server indicates the end of the server hello phase by sending the Server-Hello-Done message.

Phase 3 of the protocol is used for client authentication and key exchange, wherein the client may send his certificate and respond to a key exchange request by the server. The typical steps of phase 3 are as follows:

1) If the server has sent a Certificate Request message, the client must send either its certificate in a Certificate message or a 'no certificate 'alert.
2) The client then sends its part of the DH protocol to complete the key exchange
3) The client will also need to send a Certificate Verify message to authenticate itself, in this message the client should show the ownership of the private key that corresponds to the public key in the client certificate. This is achieved by signing a hash that contains the master secret (derived from the key exchange stage) and handshake messages. It should be noted that the random number exchanged previously are also used in this message to protect against replay attacks.

At the end of phase 3, both client and server calculate keys. Phase 4 of the protocol is the wrap-up stage. It comprises the following steps:

1) The client sends a ChangeCipherSpec message announcing that the new parameters have been loaded.
2) The client sends the *finished* message which is encrypted with the new settings, this massage uses message authentication codes (MAC) based on the agreed-upon master key to verify that the handshake is successful and both parties have the same master keys.
3) The server will do the same as above on its side.
4) The client Finished message is verified by the server and vice versa.

The "*Finished*" message is very important because it helps verify the integrity of all previous messages, which allows the detection of any tampering of protocols messages by adversaries, this makes it feasible to protect against a downgrade attack for example, wherein an adversary modifies the content of the Client-Hello message to change the list of supported TLS version and/or cipher suits, effectively keeping only old versions of the protocol, which may be less secure. This is the end of the handshake protocol, the client and server can now start exchanging encrypted messages.

2) ESTIMATION OF THE GLOBAL ENERGY COSTS OF TLS FOR WEB BROWSING

The TLS handshake protocol is used every time, and a secure connection between a client and server is established through the HTTPS application layer protocol. The annual energy costs of data usage for TLS in the year 2021 are calculated using Equation (3).

$$\begin{aligned} E_{TLS} = & \ ECDU * Number\ of\ web\ connnections * \\ & Percentage\ of\ HTTPS\ connections * \\ & Number\ of\ Bytes\ per\ TLS\ handsahe \end{aligned} \tag{3}$$

a: CALCULATION OF THE NUMBER OF WEB HTTPS CONNECTIONS

The literature does not have specific statistics on the total number of web connections per year. However, some studies estimate the monthly internet traffic for widely used websites such as the work of Joshua Hardwick in [35]. The latter provides detailed estimates of the top hundred most visited websites by search traffic. This information is utilized in this work to develop a prediction model of the annual number of visits for each website based on its rank (e.g. the most visited website is ranked 1, the second most visited is ranked 2, and so on). The next piece of information needed to complete this calculation is the total number of websites globally. The latter is estimated to be 1.8 billion, of which, approximately, 400 million are active [36]. Based on this analysis, it is estimated that in, 2021, 5.12 Tera web connections will be made, globally.

Data from [37] indicate that the default protocol HTTPS is used by 74.2% of all the websites, the study also predicts that is percentage will continue to rise. In addition, the latest analysis by Google shows the percentage of encrypted traffic based on HTTPS across its multiple products is 95%. [38]. Therefore, this study reasonably assumes that at least 90% of web connections, in 2021, are established using TLS.

b: CALCULATIONS OF TLS HANDSHAKE OVERHEADS

The number of bytes used for a TLS handshake varies according to the particular handshake variant. This work considers the most widely used version by web browser connections, anonymous clients, and authenticated servers. In this case, the length of each message in the protocol is as follows. The size of the initial *ClientHello* is approximately 170 bytes, it depends on the number of cipher suites sent by the client, and the number of present TLS ClientHello. The size of the *ServerHello* is around 75 bytes, which also depends on TLS extensions. The size of the *Certificate* message varies greatly between different servers, this message includes the certificate of the server, and all intermediate issuer certificates in the certificate chain, except the root certificate. The size of each certificate is dependent on the parameters and keys used; therefore, it is estimated that an average of 1500 bytes is needed per certificate. The other factor that affects this calculation is the length of the certificate chain up to the root certificate. This is assumed to be 4 certificates in the chain, which gives us about 6k for this message. The size of *ClientKeyExchange* is 130 bytes assuming an RSA server certificate. The size of the *ChangeCipherSpec* is one byte. The *Finished* message is around 12 bytes and varies based on the TLS version used. In addition to the above, one needs to add the overheads associated with the TLS header for each message, in total, this amounts to around 50 bytes. Based on the above, the total number of bytes to establish a new TLS session is around 6500 bytes.

TABLE 1. Comparison of social messaging apps.

Name of The App	Number of Users	Security Features	Protocol
WHATSAPP	2 Billion	End to end Encryption	Signal Protocol
FB Messenger	1.3 Billion	End-to-End Encryption (by 2022)	N/A
WECHAT	1.2 Billion	Client Server Encryption	MMTLS (based on TLS1.3)
Telegram	500 million	End to end Encryption	MTP
Signal	40 million	End to end Encryption	Signal Protocol

It is now possible to compute the annual global energy overhead of internet data usage for TLS using equation (3), this is estimated to be 3.16 million KWH.

c: PRINCIPLES OF KEY EXCHANGE SCHEMES FOR SECURE MESSAGING APPS

Social messaging apps such as WhatsApp, are the second major application for public key-based key establishment schemes. Table 1 provides a summary of the most widely used apps [39], [40] and relevant security features[41]–[43].

All above protocols, expect WECHAT, offer or will soon introduce end-to-end encryption to their respective services. To explain the principles of encryption key establishment in these applications, we will take WHATSAPP as an example.

Every WhatsApp user obtains a long-term key when they first install the application, this is stored on the device memory and typically accessible to the user. This key is subsequently used to create shared keys using the key agreement protocol depicted in figure 4.

To communicate with another WhatsApp user, a WhatsApp client first needs to establish an encrypted session. Once the session is established, clients do not need to rebuild a new session with each other until the existing session state is lost

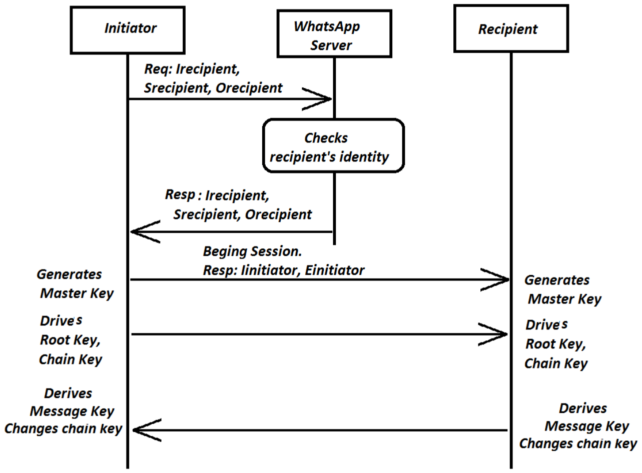
Initiator
WhatsApp Server
Recipient
Req: Irecipient, Srecipient, Orecipient
Checks recipient's identity
Resp : Irecipient, Srecipient, Orecipient
Beging Session.
Resp: Iinitiator, Einitiator
Generates Master Key
Drives Root Key, Chain Key
Derives Message Key Changes chain key
Generates Master Key
Drives Root Key, Chain Key
Derives Message Key Changes chain key

FIGURE 4. Flow diagram of WhatsApp encrypted session setup.

through an external event such as an app reinstall or device change. To establish a session:

1. The initiating client ("initiator") requests the public Identity Key, public Signed Pre-Key and a single public One-Time Pre-Key for the recipient.

The identity key, called *Recipient* is a long-term curve22519 key pair. The signed pre-key, called *Srecipient* is a medium-term curve22519 key pair and signed by *Irecipient*. The one-time pre-key, called *Orecipient* is a list of curve22519 key pairs mainly for one-time use. All these keys are generated during installation, reinstallation, or change of device.

2. The server returns the requested public key values. A One-Time Pre-Key is only used once, so it is removed from server storage after being requested.
3. The initiator saves the received keys and generates an ephemeral Curve25519 key pair, *Einitiator.*
4. The initiator loads its own Identity Key as Iinitiator.
5. Using these keys generated and requested in the above step the initiator can now calculate a shared secret with the recipient - Master Key - ECDH (Iinitiator, Srecipient) || ECDH(Einitiator, Irecipient) || ECDH(Einitiator, Srecipient) || ECDH(Einitiator, Orecipient). If there is no One Time Pre Key, the final ECDH is omitted.
6. The initiator uses a Hashed Key Derivation Function (HKDF) to derive the root key, Chain Keys from the master_secret. It takes the master key as the input keying material and extracts from it a fixed-length pseudorandom key. This key expands into several additional pseudorandom keys, resulting in the root and chain keys, both with a 32-byte value.
7. After building a long-running encryption session, the initiator can immediately start sending messages to the recipient. The initiator includes the session setup information (including includes the initiator's Einitiator and Initiator) in the header of all messages sent till the recipient responds.
8. Using this session information, the recipient calculates at its end shared secret

It is estimated, based on the above protocol, that the key agreement for establishing the end-to-end encryption in WhatsApp requires a few hundred bytes (∼400), this is significantly less than what is incurred by the TLS protocol. This is because the WhatsApp protocol establishes an encrypted using the built-in public key obtained at the installation time, instead of exchanging certificates for each connection, as the case in TLS.

The internet data usage of other messaging apps is of the same order as above, as they follow similar principles.

d: ESTIMATION OF THE GLOBAL ENERGY COSTS OF KEY EXCHANGE SCHEMES FOR SECURE MESSAGING APPS

The previous section has shown that key agreement in secure messaging apps is run only when a new encrypted session is established, i.e., when a user starts communicating with a new connection, therefore, the annual energy costs associated with the internet data usage for key exchange agreement for secure messaging apps can be calculated using equation (4).

$$E_{SM} = \sum\nolimits_{i=1}^{N} ECDU * B_i * User_i * Number\ of\ New\ Encypted\ Sessions\ per\ user\ per\ year \tag{4}$$

wherein

N: The total number of social messaging apps.

B: Data usage overhead for each key agreement run.

User: The number of users for each app.

Most secure messing users are subscribed to one of the apps listed in Table 1, so these will be sufficient for this estimation. ECDU and B have been previously calculated. The number of encrypted sessions established per year varies greatly among users depending on the level of their social activities, therefore this calculation considers a range of values for this parameter. Based on the above assumptions, it is now possible to compute the annual global energy overhead of internet data usage using equation (4), this is estimated to be 2140 to 21400 KWH, which corresponds to 10, and 100 encrypted sessions established per user annually, respectively.

B. GLOBAL ENERGY COSTS OF INTERNET DATA USAGE OF DIGITAL SIGNATURES IN CRYPTOCURRENCY

1) PRINCIPLES OF DIGITAL SIGNATURES USAGE IN CRYPTOCURRENCIES

Digital signatures are major applications for public-key algorithms, they are used heavily in cryptocurrencies to sign transactions. For example, in Bitcoins, an Elliptic curves-based algorithm generates a pair of keys private and public for each user [44]. A bitcoin address is generated from the public key using a hashing function. The ownership and control over the private key are the roots of user control over all funds associated with the corresponding bitcoin address. When spending bitcoin, the owner of the funds presents their public key and a digital signature generated from their private key. This information is propagated through the Bitcoin network to be validated by each node, to confirm that person transferring the bitcoin owned them at the time of the transfer. The size of each Bitcoin transaction varies between 400 to 600 bytes [44], [45].

2) ESTIMATION OF THE ENERGY COSTS OF DATA USAGE OF TRANSECTIONS IN CRYPTOCURRENCIES

Digital signatures are also used in other cryptocurrency schemes to verify transactions like the above. Therefore, the energy costs of the data usage of cryptocurrency transactions can be estimated using equation (5)

$$E_{CR} = \sum\nolimits_{i=1}^{k} ECDU * T_i * ACRT_i * Nodes \tag{5}$$

wherein

K: is the total number of cryptocurrency schemes

T: Length of Each Transaction for each scheme

ACRT: The number of transactions per year

Node: The number of nodes needed to validate each transaction

Parameters required for Equation (5) are obtained from the published data. For example, the number of daily transactions is obtained from [46], this is used to estimate the total number of transactions for 2021. The number of bytes per transaction is assumed to be 600 bytes based on the analysis of the previous section. The estimate in this section considers the top 5 performing cryptocurrency schemes according to [46], these include Bitcoins, Ethereum, Litecoin, Stellar, and Ripple, which generate the vast majority of global transactions. Based on this analysis, it is estimated that the energy costs of internet data usage for cryptocurrency transactions are approximately 269000 KWH in 2021.

C. DISCUSSION

The analysis results, summarized in fique 5, show that a significant portion of energy consumption related to data usage is incurred by the secure browsing application. This amounts to 3.16 million KWH in the year 2021, which is sufficient to provide electricity to 1000 UK households for a year [47]. The large energy overhead of the key agreement schemes for secure web browsing may be attributed to the large size of certificates that need to be exchanged during each TLS run (6kB on average), this contrasts with the few hundred bytes required for key agreement in secure messaging apps or digital signature in cryptocurrency transactions.

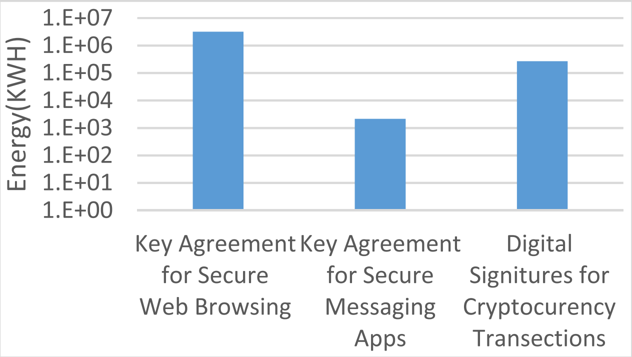

FIGURE 5. A comparison of the annual energy costs of internet data usage for three major applications of public key cryptograph.

IV. COMPARATIVE ANALYSIS OF ENERGY COSTS BETWEEN SYMMETRIC AND ASYMMETRIC KEY ALGORITHMS

This section explores the potential energy savings if solutions based on symmetric key algorithms are to replace public key-based schemes. This comparison is performed using two methods. The first approach uses the ECDU metric to evaluate the global energy costs associated with internet data usage. The second method uses an experimental technique based on constructing a small-scale network of wireless embedded devices. This is subsequently used to compare two key establishment schemes, symmetric and asymmetric, which allows for comparing the computation and communication costs of each solution in a controlled environment, and more importantly estimating the energy consumed by each device participating in the protocol.

A. COMPARISON OF GLOBAL ENERGY COSTS OF INTERNET DATA USAGE

The analysis from section 3 has concluded that the public key-based TLS protocol used for establishing secure connections on the web consumes more than 92% of the global energy incurred by public-key cryptography uses. Therefore, this study only considers the secure web browsing application to perform the comparison with symmetric schemes.

1) PRINCIPLES OF SECURE COMMUNICATION USING THE SYMMETRIC-KEY BASED PROTOCOL KERBEROS

Kerberos is an authentication protocol that allows nodes communicating over a non-secure network to prove their respective identity to one another in a secure manner and to establish a shared session key. This protocol uses symmetric key algorithms; therefore, it was adopted in this study for comparison as a possible alternative for the public key-based TLS

There are three parties involved in Kerberos, a client (e.g., node A), a server (e.g., node B), and a trusted third party that acts as a key distribution center (KDC).

Each entity on the network shares a long-term secret key with the KDC before any communication. This key is used to prove the node's identity, and to establish an encrypted session with the KDC. The Latter, subsequently delivers a Ticket Granting Ticket (TGT), which can be used to establish secure links with other nodes in the network. In essence, the KDC provides two services Authentication (AS) and Ticket Granting (TGS). The former is only done once for each entity in the network, as a result, the entity obtains a TGT which can be used to obtain additional tickets. An overview of the Kerberos version 5 protocol is provided below and shown in Figure 6 [48].

1. The Client sends a message (AS_REQ)) to the authentication service (AS) in the KDC requesting the credentials of a given node in the network (e.g., Server B).

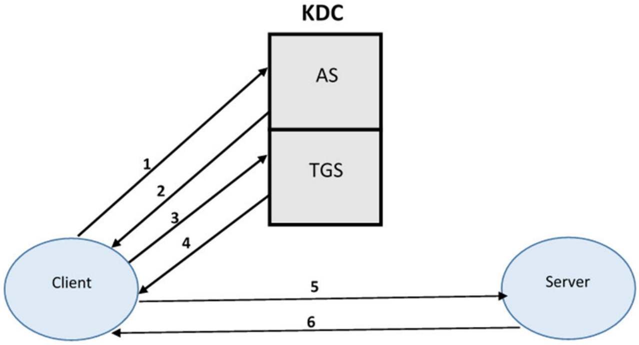

FIGURE 6. An overview of kerberos protocol.

This message also includes the client's identity and an expiration time until when the authentication will remain valid.

2. The KDC checks the validity of the client's identity and responds with a message (AS_REP), which consists of two parts
   a) A client ticket that contains a session key ($K_{client,KDC}$), the expiration time and its TGS service name. These data are all encrypted with the secret key of the client (K_{client}), stored in the KDC.
   b) A granting ticket that contains a session key, the expiration time, and the client's identity. This information is encrypted with the secret key of the KDC (K_{KDC}). This part of the message is referred to as the Ticket Granting Ticket (TGT), it can be used to request additional tickets but cannot be decrypted or modified by the client, otherwise, it will be rejected.

The client is now ready to establish secure communication with other nodes in the network, as follows.

3. The client sends a request to the KDC (TGS_REQ), which includes the following fields
   a) An authenticator field that includes a timestamp and a checksum, both encrypted using $K_{client,KDC}$.
   b) A ticket-granting ticket field that includes the TGT received in Step 2 (i.e., during the authentication phase).
   c) The Identity of the node the client wants to establish a connection with.
   d) An expiration time for the TGT.
4. The KDC responds with a message (TGS_REP) that consists of two parts
   a) A client ticket that includes a new session key containing a new session key ($K_{client,server}$) that can be used by the client and the nodes they want to communicate with (e.g., server) for authentication and encryption. This part also encloses the name of the node requested and the expiration time of the new ticket. This information is encrypted using $K_{client,KDC}$.
   b) A server ticket that includes the new session key ($K_{client,server}$), the client's identity and the expiration time of the ticket. This information is encrypted using the server key K_{Server}.
5. The client now sends a request to the server (AP_REQ), which has the following fields
   a) The server ticket received from the KDC in Step 4
   b) An authenticator field that includes a timestamp and a checksum, both encrypted using the shared session key $K_{client,server}$ received in Step 4.

The server uses his K_{Server} to decrypt the server ticket and extract the shared session key $K_{client,server}$ and to verify the client's identity

6. The server, optionally, responds with an AP_REP message containing a timestamp encrypted using the shared key $K_{client,server}$. This allows the client to verify the identity of the server.

The above protocol works for devices that are within one ''realm''. The latter refers to the domain over which a Kerberos authentication server has the authority to authenticate a device, in other words, all devices whose credentials are managed by one KDC. Therefore, to enable secure communication between network nodes from different realms, Kerberos allows sharing of inter-realm keys between KDCs. Cross-realm communication requires additional steps [48]. This includes a TGS_REQ that is sent from the client to the remote KDC and a TGS_REP sent from the remote KDC to the client. The format of both messages is the same as explained previously. Effectively, the client, in this case, needs first to ask for the credential of the KDC of the server's realm, then communicate with it directly to request the credential for the server. Figure 7 shows a block diagram of the inter-realm authentication using Kerberos, wherein a client from (realm 1) establishes a secure communication from a server in (realm 2). It is reasonable to assume the inter-realm version of the Kerberos protocol will be used for establishing secure communication on the internet because clients and servers are most likely placed in different realms.

2) COMPARISON OF ENERGY COSTS OF DATA USAGE BETWEEN TLS AND KERBEROS

The total energy costs of data usage in 2021 for potentially using Kerberos can be estimated similarly to the TLS case, using equation (6).

$$\begin{aligned} E_{Kerberos} = {} & ECDU * Number\ of\ web\ connnections * \\ & Percentage\ of\ HTTPS\ connections * \\ & Number\ of\ Bytes\ per\ Kerberos\ handsahe \end{aligned} \tag{6}$$

The inter-realm scheme shown in figure 7 is used to estimate the number of bytes per Kerberos handshake, assuming AES 128 encryption scheme is used. It was found that the size of data exchanged is approximately 2.7 KB. Other parameters from equation 6, including *ECDU* and *the number of secure web connections,* are the same as previously estimated for the case of TLS.

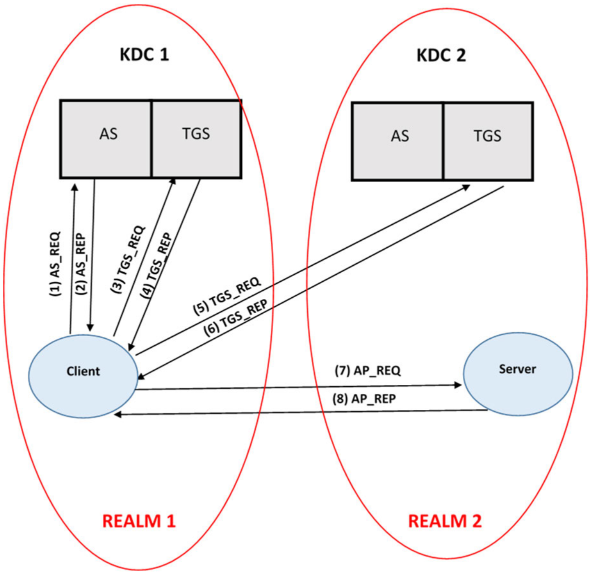

FIGURE 7. An overview of inter-realm Kerberos Protocol.

Based on the above assumption, the energy cost of internet data usage for Kerberos would be 1.33 million KWH IN 2021, which is 42% of the energy incurred by TLS.

Finally, it is worth noting that the current use of Kerberos in Microsoft Windows as an authentication protocol has shown a significant overhead for the ticket size, reaching thousands of KB [49]. Although the proposed use in this study is limited to authentication and session agreement, based on a pre-shared key, the scalability challenge of Kerberos [8]and the inflation of its ticket size have to be addressed before the reported saving above can be achieved.

B. COMPARISON OF ENERGY COSTS IN A WIRELESS SENSOR NETWORK

The ECDU metric is useful for estimating the global energy costs of data transactions, however, it does not evaluate the energy costs of each protocol as incurred by the communicating nodes, which is important for energy-constrained devices. This section develops an experiment that estimates the computation and communication costs of two key agreement schemes, based on symmetric and asymmetric key algorithms respectively, which gives a better insight into the energy efficiency of these schemes from devices' perspectives.

1) EXPERIMENTAL SETUP

The experimental platform in this work was constructed using Zolertia Zoul devices [50], which are equipped with CC2538 core ARM Cortex-M3, 512 KB ROM, and 32 KB RAM, these devices are typically used for the energy-constrained environment. A lightweight operating system called "Contiki" is used [51], this was also originally developed for tiny network sensors. The experiment compares two key agreement schemes, symmetric and asymmetric. The first is the "light-weight" variant of the Kerberos key distribution scheme that requires four steps as shown in figure 8 [11], this optimized version removes the ticket-granting service, but still requires a trusted third party that is assumed to have shared long terms key with the network's nodes. The implementation of the scheme assumes that 128 AES encryption is used. The second protocol is based on ECMQV, an authenticated version of the elliptic curve Diffie-Hellman key exchange, and uses a 256-bit prime field GF(p) as the underlying algebraic structure [11], this scheme only requires two message exchanges to establish a shared key between a server and a client.

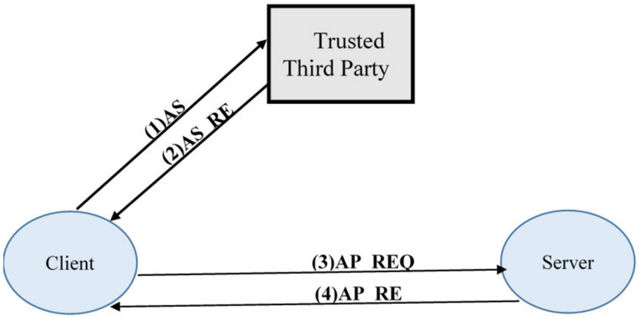

FIGURE 8. An overview of inter-realm Kerberos Protocol A light-Wight variant of Kerberos[11].

2) IMPLEMENTATION AND ENERGY ESTIMATION

The Kerberos implementation was developed using C language and was based on the AES-CBC mode, this was subsequently embedded into Contiki. In this case, three Zolertia Zoul devices are used to construct the network, a client, a server, and a trusted party. A similar approach was also adopted to implement the ECDH key exchange algorithm, in this case only two devices were needed, a client and a server. Both implementations are built on top of the UDP connection method over IPv6 by calling uip_udp_packet_send and uip_udp_packet_receive functions.

To measure the energy consumption of each component in the devices, an application called the "energy module" on the Contiki operating system was used. The energest module measures time by taking the readings of clock ticks while the device is receiving, transmit state, processing (CPU) mode, and low power mode. The processing time of each component in milliseconds (ms) is calculated by Formula

$$Processing\ time\,[ms] = \frac{Energest\_Value\,(ticks) \times 1000}{CLOCK\_SECOND} \quad (7)$$

To calculate the energy consumption of these states, the following formula was used:

$$E = \frac{Energest\_Value \times Current \times Voltage}{CLOCK\_SECOND} \quad (8)$$

where the *Energest_Value* is read off the terminal directly while the program is running. The *voltage* and *current* at different operating levels are obtained from Table 2.

3) RESULTS AND DISCUSSION

Energy consumption results for the implementation of Kerberos and ECQVM protocols are shown in figures 9, 10, and 11 respectively. These results were obtained by running each protocol ten times on the Zoul devices and taking the

TABLE 2. Zolertia Zoul device power breakdown.

Components	CPU	LPM	Tx	Rx
Current	0.6 mA	0.0004mA	24 mA	20 mA
Operating Voltage	3.4V			

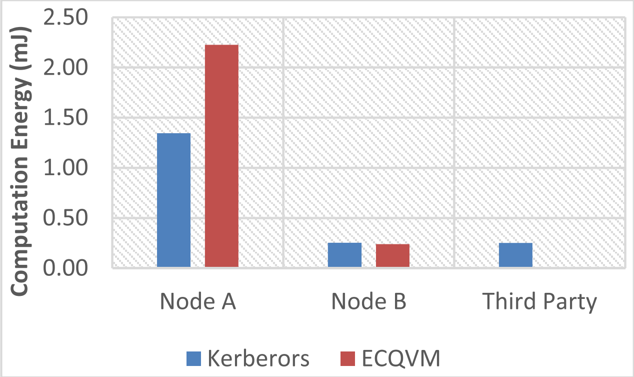

FIGURE 9. Computation energy costs of key agreement schemes.

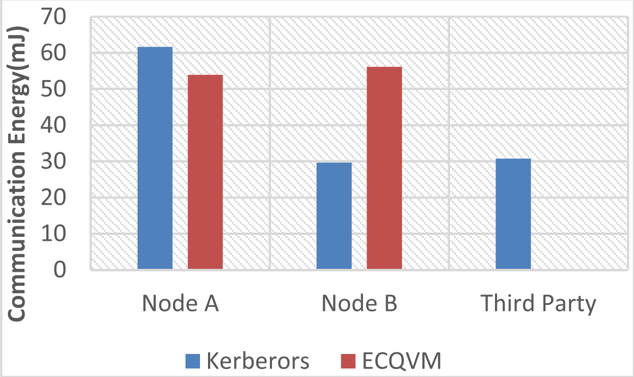

FIGURE 10. Communication energy costs of key agreement schemes.

average energy consumption of all runs. Figure 9 shows the computation energy costs of each protocol, calculated as the sum of the energy consumed during the CPU and low power mode (i.e. ideal state). The results show that for node A (the client), the asymmetric scheme dissipates approximately 66% more energy. For node B, both schemes consume similar amounts of power. The total energy cost in each case is 1.87 and 2.46 (mJ) for Kerberos and ECQVM respectively, this includes the energy dissipated by the third-party nodes. This is due to the higher computation complexity required by the asymmetric scheme.

Figure 10 shows the communication energy costs of each protocol, calculated as the sum of the energy consumed during the transmission and reception. The results show that for node A (the client), the symmetric scheme dissipates approximately 10% more energy. For node B (the server), the asymmetric scheme consumes 19% more energy. The total energy cost in each case is 121 and 110 (mJ) for Kerberos and ECQVM respectively, this includes the energy dissipated by the third party node. The slightly higher energy consumption of the symmetric scheme (∼10%) is since the latter requires more messages to establish a key.

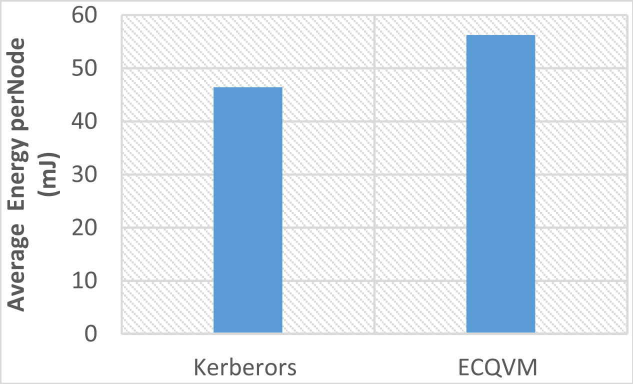

FIGURE 11. A Comparisons of average energy consumed per communicating node.

An important metric that should be considered in this context, to assess the energy efficiency of each scheme, is the average energy consumption per node. This is assuming that any node can be the initiator of the communication (i.e. the client) or the target (i.e. the server). Figure 11 shows the average energy costs consumed by each scheme per node. This calculation excludes the third-party node, as the latter is typically a resource-rich device. The results show that the symmetric scheme requires around 20% less energy per node compared to the public key encryption protocol, which makes the former a more efficient approach for energy-constrained devices.

Finally, it is worth noting here it is expected that more savings can be achieved in this case if AES is replaced by a light cipher such as that proposed in [52].

V. DISCUSSION OF FINDINGS

This study has investigated the energy cost of security applications that are based on public-key algorithms compared to these using symmetric-key schemes. The work considered two application scenarios where energy efficiency is important. The first case study assessed the global energy cost of using public key security solutions, and whether this can be reduced by adopting techniques and protocols based on symmetric key algorithms. To perform this calculation, this work proposed a metric called energy cost of data usage (ECDU), which estimates the energy dissipated for the transmission of each bit over the internet. This was subsequently used to estimate the energy costs of the major applications of the asymmetric key algorithm, which include key agreement protocols (e.g. for secure web browsing and messaging apps) and digital signatures (e.g. for cryptocurrency transactions). Our analysis concluded that the annual global cost of asymmetric key solutions is predicted to be 3.16 million KWH in the year 2021, which is sufficient to provide electricity to 1000 UK households for a year, mostly consumed using TLS for secure web browsing. It was also found that this figure can be reduced to 1.33 million KWH (i.e. 42% of the original cost) if solutions based on symmetric key algorithms are used instead, more specifically, the Kerberos inter-realm protocol. However, there are still challenges related to the scalability and key distribution problem that need to be addressed before symmetric key solutions can be adopted.

The second case study assessed the energy efficiency of asymmetric key schemes in resource-constrained systems, and whether this can be reduced by adopting techniques and protocols based on symmetric key algorithms. This is especially important due to the predicted increase in the number of internet-of-things devices and the need to secure such systems in the post-quantum era. To perform this analysis, a wireless sensor network using Zolertia Zoul devices was constructed. Next, the energy costs for each run of two key agreement protocols were compared, an asymmetric scheme, based on a lightweight variant of the Kerberos protocol, and an asymmetric scheme (ECQVM). The results showed that a 20% saving of the average energy per communicating node can be saved if the symmetric key algorithm is used. Based on the above analysis, it can be concluded that security techniques based on symmetric key systems can provide a more energy-efficient alternative than current public-key-based solutions.

VI. CONCLUSION AND FUTURE WORK

The proliferation of energy-constrained internet of things devices, combined with the need to adopt higher complexity quantum resilient cryptographic algorithms, makes it more challenging to continue to use public-key algorithms for all applications. This work has shown that symmetric-key-based security solutions for key establishments offer a more efficient solution from an energy-saving perspective. Our results show that a 58% saving in global energy costs of public key-based applications can be achieved by adopting symmetric key systems. In addition, this work has also shown that a 20% reduction of the energy consumed by a wireless device during a key agreement protocol, can be achieved if symmetric key encryption is used.

At present, there are several initiatives to introduce changes to the cryptographic primitives used to secure the Internet. We have mentioned the NIST Post-Quantum Cryptography project [1] that is attempting to find quantum-resistant key establishment algorithms and digital signatures suitable for standardization. These algorithms will have a notably different computational profile and bandwidth requirements than current methods and further study should investigate this both when suitable candidates are selected and when the final parameterizations and variants are formally set in a standard. The different performance profiles may also lead to changes to existing Internet standards as researchers seek to work around some of the limitations of the new algorithms. Example proposals include OPTLS[53] and KEMTLS [54] as well as post-quantum digital currencies such as ABCMint and Tidecoin. If these proposals are globally adopted, then they too should be included in further study.

Initiatives for new cryptographic algorithms are not restricted to public-key cryptography and there is also a NIST Lightweight Cryptography project [55] intended to produce symmetric algorithms for authenticated encryption with associated data (AEAD). The successful candidates from this process will be even more energy-efficient than the symmetric algorithms of this study and again further study should investigate this both when suitable candidates are selected and when the final parameterization and variants are standardized. Likewise, if more efficient protocols for the symmetric cryptographic key establishment or symmetric authentication are proposed, these too should be investigated.

REFERENCES

[1] NIST. (Jul. 21, 2021). *NIST RSA Project*. [Online]. Available: https://csrc.nist.gov/glossary/term/RSA

[2] NIST. (Jul. 21, 2021). *NIST Project on Elliptic Curve Cryptography*. [Online]. Available: https://csrc.nist.gov/Projects/elliptic-curve-cryptography

[3] J.-P. Aumasson, "The impact of quantum computing on cryptography," *Comput. Fraud Secur.*, vol. 2017, pp. 8–11, Jun. 2017.

[4] NIST. (2021). *NIST Project on Post-Quantum Cryptography*. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography

[5] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT systems: Design challenges and opportunities," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2014, pp. 417–423.

[6] T. M. Fernández-Caramés, "From pre-quantum to post-quantum IoT security: A survey on quantum-resistant cryptosystems for the Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6457–6480, Jul. 2020.

[7] A. Braeken, "Public key versus symmetric key cryptography in client–server authentication protocols," *Int. J. Inf. Secur.*, vol. 21, no. 1, pp. 103–114, 2022.

[8] J. Großschädl, A. Szekely, and S. Tillich, "The energy cost of cryptographic key establishment in wireless sensor networks," in *Proc. 2nd ACM Symp. Inf., Comput. Commun. Secur.*, 2007, pp. 380–382.

[9] M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and G. Zaverucha, "Post-quantum zero-knowledge and signatures from symmetric-key primitives," in *Proc. ACM Signal Conf. Comput. Commun. Secur.*, 2017, pp. 1825–1842.

[10] N. Anhao, "Bitcoin post-quantum," Tech. Rep., 2018.

[11] *The Kerberos Network Authentication Service (V5)*, MIT, Internet Soc., Reston, VA, USA 2005.

[12] R. Hintemann and S. Hinterholzer, "Energy consumption of data centers worldwide," in *Proc. 6th Int. Conf. ICT Sustain. (ICT4S)*, Lappeenranta, Finland, 2019.

[13] G. Fagan, J. P. Gallagher, L. Gammaitoni, and D. J. Paul, "Energy challenges for ICT," in *ICT—Energy Concepts for Energy Efficiency and Sustainability*. London, U.K.: IntechOpen, 2017.

[14] S. Kemp, "Digital 2021: Global overview report," Tech. Rep., 2021.

[15] B. Raghavan and J. Ma, "The energy and emergy of the internet," in *Proc. 10th ACM Workshop Hot Topics Netw.*, 2011, pp. 1–6.

[16] D. Costenaro and A. Duer, "The megawatts behind your megabytes: Going from data-center to desktop," in *Proc. ACEEE Summer Study Energy Efficiency Buildings (ACEEE)*, Washington, DC, USA, 2012, pp. 13–65.

[17] L. S. Vailshery. (Jul. 13, 2021). *IoT and Non-IoT Connections Worldwide 2010–2025*. [Online]. Available: https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/

[18] H. Tankovska. (Jul. 13, 2021). *WhatsApp—Statistics & Facts*. [Online]. Available: https://www.statista.com/topics/2018/whatsapp/

[19] A. de Vries, "Bitcoin's energy consumption is underestimated: A market dynamics approach," *Energy Res. Social Sci.*, vol. 70, Dec. 2020, Art. no. 101721.

[20] J. Yuventi and R. Mehdizadeh, "A critical analysis of power usage effectiveness and its use in communicating data center energy consumption," *Energy Buildings*, vol. 64, pp. 90–94, Sep. 2013.

[21] V. K. Sachan, S. A. Imam, and M. T. Beg, "Energy-efficient communication methods in wireless sensor networks: A critical review," *Int. J. Comput. Appl.*, vol. 39, pp. 35–48, Feb. 2012.

[22] E. E. Dikel, Y. E. Li, M. Vuotari, and S. Mancini, "Evaluating the standby power consumption of smart LED bulbs," *Energy Buildings*, vol. 186, pp. 71–79, Mar. 2019.

[23] *Amazon Echo (1st Generation) Power Consumption*. [Online]. Available: https://www.amazon.co.uk/gp/help/customer/display.html?nodeId=202086760

[24] M. Pickavet, W. Vereecken, S. Demeyer, P. Audenaert, B. Vermeulen, C. Develder, D. Colle, B. Dhoedt, and P. Demeester, "Worldwide energy needs for ICT: The rise of power-aware networking," in *Proc. 2nd Int. Symp. Adv. Netw. Telecommun. Syst.*, Dec. 2008, pp. 1–3.

[25] *Calculating Power Requirements for MX960 Routers*, Juniper Netw., Sunnyvale, CA, USA, Mar. 2021.

[26] W. Vereecken, W. Van Heddeghem, M. Deruyck, B. Puype, B. Lannoo, W. Joseph, D. Colle, L. Martens, and P. Demeester, "Power consumption in telecommunication networks: Overview and reduction strategies," *IEEE Commun. Mag.*, vol. 49, no. 6, pp. 62–69, Jun. 2011.

[27] L. S. Vailshery, "Number of tablet users worldwide from 2013 to 2021 (in billions)," Tech. Rep., 2021.

[28] S. O'Dea, "Number of smartphone subscriptions worldwide from 2016 to 2026," Tech. Rep., 2021.

[29] N. D. Loisy, *Transportation and the Belt and Road Initiative: A Paradigm Shift*. 2019.

[30] *Logistics Facts*, SCMO, 2019.

[31] *Cisco Annual Internet Report (2018–2023) White Paper*, CISCO, San Jose, CA, USA, 2020.

[32] A. S. G. Andrae and T. Edler, "On global electricity usage of communication technology: Trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015.

[33] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," *IEEE Trans. Mobile Comput.*, vol. 5, no. 2, pp. 128–143, Feb. 2006.

[34] *VNI Complete Forecast Highlights*, CISCO, San Jose, CA, USA, 2021.

[35] J. Hardwick. (Jul. 25, 2021). *Top 100 Most Visited Websites by Search Traffic*. [Online]. Available: https://ahrefs.com/blog/most-visited-websites/#:~:text=Top%20100%20most%20visited%20websites%20globally%20%20,%20%201%2C202%2C065%2C409%20%2069%20more%20rows%20

[36] O. Djuraskovic. (Jul. 25, 2021). *How Many Websites are There?* [Online]. Available: https://firstsiteguide.com/how-many-websites/

[37] W3Techs. (2021). *Usage Statistics of Default Protocol HTTPS for Websites*. [Online]. Available: https://w3techs.com/technologies/details/ce-httpsdefault

[38] Google. (2021). *Google Transparency Report: HTTPS Encryption on the Web*. [Online]. Available: https://transparencyreport.google.com/https/overview?hl=en_GB

[39] Statista Research Department. (2021). *WhatsApp—Statistics & Facts*. [Online]. Available: https://www.statista.com/topics/2018/whatsapp/

[40] Statista Research Department. (Jul. 26, 2021). *Most Popular Global Mobile Messenger Apps as of April 2021*. [Online]. Available: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-a

[41] C. Hou, J. Shi, C. Kang, Z. Cao, and X. Gang, "Classifying user activities in the encrypted WeChat traffic," in *Proc. IEEE 37th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Nov. 2018, pp. 1–8.

[42] *WhatsApp Encryption Overview*, WhatsApp, Menlo Park, CA, USA, 2020.

[43] K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, "A formal security analysis of the signal messaging protocol," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroSP)*, Apr. 2017, pp. 451–466.

[44] A. M. Antonopoulos, "Keys," in *Mastering Bitcoin*. vol. 1, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2017, ch. 4.

[45] *Analysis of Bitcoin Transaction Size Trends*, TradeBlock, 2015.

[46] Statista and Coin Metrics. (Jul. 27, 2021). *Several Daily Transactions in Bitcoin, Ethereum, and Nine Other Cryptocurrencies From January 2017 to July 11, 2021*. [Online]. Available: https://www.statista.com/statistics/730838/number-of-daily-cryptocurrency-transactions-by-type/

[47] Statista, Government of the United Kingdom, and Department for Business, Energy and Industrial Strategy. (Jul. 27, 2021). *Average Domestic Electricity Consumption Per Household in Great Britain in 2019, by Region (in Kilowatt-Hours) [Graph]*. [Online]. Available: https://www.statista.com/statistics/517845/average-electricity-consumption-uk/

[48] K. Fabrice, "Understanding Kerberos v5 authentication protocol," SANS Inst., Bethesda, MD, USA, Tech. Rep., 2003.

[49] E. Hadsell. (2011). *Kerberos and Access Token Limitations*. [Online]. Available: https://www.giac.org/paper/gsec/5111/kerberos-access-token-limitations/104962

[50] (2016). *Zolertia RE-Mote Platform*. [Online]. Available: https://github.com/Zolertia/Resources/wiki/RE-Mote

[51] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, Nov. 2004, pp. 455–462.

[52] B. Aboushosha, R. A. Ramadan, A. D. Dwivedi, A. El-Sayed, and M. M. Dessouky, "SLIM: A lightweight block cipher for Internet of Health Things," *IEEE Access*, vol. 8, pp. 203747–203757, 2020, doi: 10.1109/ACCESS.2020.3036589.

[53] H. Krawczyk and H. Wee. *The OPTLS Protocol and TLS 1.3*. [Online]. Available: https://eprint.iacr.org/2015/978.pdf

[54] P. Schwabe, D. Stebila, and T. Wiggers, "Post-quantum TLS without handshake signatures," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 1461–1480.

[55] *The NIST Lightweight Cryptography Project*. [Online]. Available: https://csrc.nist.gov/projects/lightweight-cryptography

BASEL HALAK is currently an Associate Professor in secure electronics and the Director of the Cyber Security Academy, University of Southampton. He is also a Visiting Scholar with the Technical University of Kaiserslautern, an Industrial Fellow of the Royal Academy of Engineering, a Senior Fellow of the Higher Education Academy, and a National Teaching Fellow of Advance HE U.K. He has published more than 100 refereed conference and journal papers and authored five books on the security and reliability of electronic devices and systems. His research interests include hardware security, digital design, and embedded systems. He is with several technical program committees, such as HOST, IEEE DATE, DAC, IVSW, ICCCA, ICCCS, MTV, and EWME. He is a member of the Hardware Security Working Group of the World Wide Web Consortium (W3C). He is an Associate Editor of IEEE ACCESS and the Editor of the *IET Circuits, Devices and Systems Journal*.

YILDIRAN YILMAZ received the Graduate degree in computer engineering from Pamukkale University, Pamukkale, Turkey, in 2010, and the M.Sc. degree (Hons.) and the Ph.D. degree in electronic and computer science from the Department of Cyber Security, University of Southampton, Southampton, U.K., in 2015 and 2019, respectively. His M.Sc. project was about enhanced security of any type of file and text storage. He is currently an Assistant Professor with RTEU University, Turkey. His research interests include the design and implementation of lightweight authentication for the Internet of Things devices, security evaluation of resource-constrained devices and programming, and simulations of the Contiki operating systems.

DANIEL SHIU is currently the Chief Cryptographer at Arqit. He worked for GCHQ, the U.K.'s intelligence, cyber, and security agency for 20 years. He was the U.K.'s Head of Cryptographic Design and Quantum Information Processing, part of the initial National Technical Authority function assumed by the new National Cyber Security Centre (NCSC). He also worked as the Head of the Heilbronn Institute for Mathematical Research (HIMR), which is a linchpin of the government's "Advanced Mathematics" strategy. He represented GCHQ in helping to found and direct the National Quantum Technologies Program. During his career, his results and expertise have earned him multiple prizes, including an international award for best crypto-mathematician, and on three separate occasions an international award for the year's best cryptanalytic achievement.