Flexible End-to-End Content Security in CCN

Christopher A. Wood and Ersin Uzun
Palo Alto Research Center
Palo Alto, CA 94304
Email: Christopher.Wood@parc.com, Ersin.Uzun@parc.com

***Abstract*—Content-centric networking (CCN) project, a flavor of information-centric networking (ICN), decouples data from its source by shifting the emphasis from hosts and interfaces to information. As a result, content becomes directly accessible and routable within the network. In this data-centric paradigm, techniques for maintaining content confidentiality and privacy typically rely on cryptographic techniques similar to those used in modern digital rights management (DRM) applications, which often require multiple consumer-to-producer (end-to-end) messages to be transmitted to establish identities, acquire licenses, and access encrypted content. In this paper, we present a secure content distribution architecture for CCN that is based on proxy re-encryption. Our design provides strong end-to-end content security and reduces the number of protocol messages required for user authentication and key retrieval. Unlike widely-deployed solutions, our solution is also capable of utilizing the opportunistic in-network caches in CCN. We also experimentally compare two proxy re-encryption schemes that can be used to implement the architecture, and describe the proof of concept application we developed over CCNx.**

I. Introduction

In recent years there has been a thrust of research focused on information-centric networking (ICN) as a future architecture for the Internet. One of the driving forces behind this paradigm is the need to make more efficient use of the available bandwidth. Content Centric Networking project (CCN), a well-known instantiation of the ICN paradigm, addresses this problem by decoupling data, or content, from its source. As a result, content, rather than host and interface addresses, become directly addressable entities in the network. A necessary byproduct of this decoupling is that the security of all sensitive (premium) content must be safeguarded from each producer to its respective consumers with end-to-end protection.

Digital Rights Management (DRM) applications are one particular class of applications that require such end-to-end content protection. Modern DRM technologies typically employ some form of hybrid public and private-key encryption scheme to secure and individualize content for its respective consumers. Licenses containing the cryptographic information, such as an encrypted content key that can be used to decrypt the primary content of interest, are typically issued by the producer and the encrypted content is traditionally distributed using Content Delivery Networks (CDNs).

For example, Netflix, the leading provider of online content in the United States and Canada responsible for roughly 30% of all downstream traffic, leverages Microsoft's PlayReady DRM technology and large-scale content delivery networks (CDNs) to securely distribute its premium content to all subscribed consumers [1]. However, most of the existing DRM solutions are tightly coupled to the host-based Internet architecture and usually require consumers to interact with numerous servers for authentication, license acquisition and retrieval of the protected content.

In this paper, we propose a CCN-friendly architecture for secure content dissemination that does not rely on trusted CDNs or traditional PKI cryptosystems. Instead, we leverage identity-based cryptography to enable simple and intuitive derivation of public keys that can be associated with users and their access rights and the use of proxy re-encryption (PRE) for true end-to-end security. Our full PRE-based architecture has many benefits in the context of CCN, including minimal keys to store for producers and consumers (e.g. consumers need only store two private keys associated with their identity - one for identity-based encryption and one for proxy re-encryption), complete end-to-end content security while still leveraging the use of network caches, and easy accommodation of intermediary nodes that can perform content re-encryption on behalf of the producer, among others.

In addition, we study the performance of two significantly different PRE schemes, one identity-based construction proposed by Green and Ateniese [12] and another based on a combination of ElGamal encryption and Schnorr's signature scheme proposed by Chow et al. [8], to determine the feasibility and usability of integrating PRE into a DRM application. We narrow our focus on DRM applications because of the inherent content security requirements that must be enforced for all content transferred between producers and consumers. We then present a proof of concept implementation for the proposed architecture operating over CCNx, the open source implementation of CCN, and discuss its advantages and shortcomings.

II. Related Work

Online content providers such as Netflix account for a large amount of the downstream Internet and mobile network traffic in North America [18]. Previous research has revealed that the content delivery strategy driving Netflix's business model is partitioned among their own data center, a set of Amazon EC2 cloud servers, and a plethora of widely-distributed and strategically located CDNs. Each piece of media streamed from a CDN node is encoded and DRM protected before being sent to the consumer's client device. Currently, Netflix uses Microsoft's Silverlight PlayReady DRM technology to protect its content. Under the hood, PlayReady uses a hybrid encryption approach that encrypts each piece of content with a unique symmetric key. The encrypted content is then distributed over the CDNs, whereas the symmetric key to decrypt this content is distributed by producers to specific consumers by first encrypting it with the public key of the target consumer and then transferring it through a secure tunnel.

Adobe and Spotify use similar DRM technologies with slightly different hybrid encryption techniques. For example, Adobe's DRM technology protects data in transit using a stream cipher keyed with a symmetric key shared between the producer and consumer [20]. This session key can be generated using any standard key exchange technique, such as the Diffie Hellman key exchange protocol. Spotify also uses a symmetric key stream cipher to protect data in transit, but adds further protection by encrypting the data using the AES block cipher before sending it across the network [10]. The key to decrypt this data is also encrypted using the stream cipher before being transferred.

These three DRM technologies share the same properties that multiple end-to-end messages between the consumer and producer must be exchanged to access premium content, and the user devices are required to store many different decryption keys. In scenarios where the bandwidth between consumers and producers is expensive or severely limited, these requirements may impede a consumer's ability to access the content.

Cryptographic techniques to solve the problem of secure cloud storage are quite similar in spirit to those used in DRM technologies. The goal is to protect content in public clouds from being inadvertently accessed by unauthorized users. As such, individualizing the content in the cloud through the use of hybrid encryption schemes is a popular solution. The primary difference between these two settings is that consumers now serve as producers, but use the cloud (e.g. a CDN) to deliver content to others on their behalf. For the purpose of individualizing content, attribute-based encryption techniques show great promise [3], [14] for enforcing fine-grained access control to content stored in a public location. Other researchers have turned to PRE as a means of individualizing content in the cloud. Recently, Xiong et al. [21] leveraged PRE in the design of a subscription-based content delivery system called CloudSeal. Their system uses PRE in a hybrid-encryption design that protects information which is used to encrypt content before it is stored in a publicly-accessible cloud. Unfortunately, the security of their PRE variant is not formally proven using any complexity assumptions in any adversarial model.

Broadcast encryption is another well known technique for secure content distribution. In the typical scenario, a provider will encrypt some message for a subset of users and then broadcast the ciphertext on a specific channel. Each user belonging to the original subset and who is also listening on that channel may then use their secret key to decrypt the content [16]. In [17], Misra et al. showed how broadcast encryption can be used with an appropriate secret sharing scheme to preserve content accessibility in the event that producers are taken offline. Unfortunately, despite this benefit, direct applications of broadcast encryptions schemes are limited in that the group of users is finite, which requires content to be re-encrypted quite often in the presence of a rapidly changing set of consumers.

III. Preliminaries

A. Content-Centric Networking

Content-centric networking (CCN) is one of several proposed information-centric network designs for the future Internet architecture that decouples data from its origin and enables relevant content to be pushed and stored throughout the network to minimize bandwidth consumption. Network caches and addressable content, rather than addressable hosts or interfaces, enable this new architecture to reduce the overall network congestion and latency by keeping content closer to its intended recipients. CCN also stipulates that all content is signed by its original producers, thus enabling security properties that are independent from the channel through which content moves.

The process by which content is requested is through the issuance of an *interest*. The components of an interest are simply strings and can therefore be used to store any type of data, including human readable names or an arbitrary pieces of binary data encoded as URL-friendly strings (i.e. ones in which the alphabet does not use characters reserved for interest parsing).

Upon receiving an interest, a router looks for a match in its *content store* (CS), which is the cache that persists content already requested from other consumers. Matching is done based on content names; complete interest name matches in the CS allow the router to satisfy the interest and forward the respective content to the downstream router. Interests that cannot be satisfied from the CS are recorded in the router's *pending interest table* (PIT) together with the corresponding downstream interface and forwarded to the appropriate upstream router based on the *forward interest base* (FIB) table. This table is populated using routing protocols, as in the case of the current Internet design. For efficiency reasons, multiple interests for the same name are aggregated at the PIT to prevent duplicate interests being sent upstream. Once a content matching a PIT entry is received by a router, the content is cached and sent to all downstream interfaces associated with the PIT entry. Upon completion, the entry is cleared,.

B. Proxy Re-Encryption Overview

Proxy Re-Encryption (PRE), first conceptualized by Blaze et al. [4], is a family of cryptographic schemes in which an untrusted proxy is allowed to transform a ciphertext encrypted under Alice's public key to one encrypted under Bob's public key, given an appropriate conversion key provided by Alice. As an extension of traditional PKI schemes that enables decryption rights to be selectively delegated, PRE has many practical benefits that can enable message transfer in secure email systems, fine-grained access control in secure cloud storage, and, most importantly for this work, improved DRM technologies [12].

Formally, a PRE scheme is a tuple of six algorithms (Setup,KeyGen,Encrypt,ReKeyGen,ReEncrypt,Decrypt) defined below in general terms for *multi-hop* schemes. A *single-hop* scheme only permits a piece of ciphertext to be re-encrypted once. In this context, a level 1 ciphertext is the original encrypted ciphertext, a level 2 ciphertext is the result of a level 1 ciphertext being re-encrypted, and so on.

- Setup(1^k): This procedure takes a security parameter k, which determines the size of the underlying group upon which all operations take place, and generates and outputs the public set of parameters params. This procedure may also output a master secret key.
- KeyGen(params): Generate and output a private and public key pair. Identity-based schemes typically specify a particular identity and master key that are used in the creation of the private key.
- Encrypt(params, pk, m):Encrypt plaintext $m \in \mathcal{M}$ using the input public key pk (or identifier) and output the resulting level one ciphertext c_i^1.
- ReKeyGen(params, pk_i, pk_j): Generate and output a re-encryption key $rk_{i \to j}$ using the public parameters and public keys for users i and j.
- ReEncrypt(params, $rk_{i \to j}$, c_i^n): Re-encrypt the level n ciphertext c_i^n, which is encrypted under the public key of user i, to a new level $n+1$ ciphertext c_j^{n+1} using the re-encryption key $rk_{i \to j}$ that may then be decrypted by the secret key of user j.
- Decrypt(params, sk_j, c_j^n): Parse the level n ciphertext c_j^n to determine n, decrypt the ciphertext accordingly using the secret key sk_j, and output the original plaintext m.

Further distinctions between different PRE schemes can be made based on whether they are *unidirectional* or *bidirectional*. In a unidirectional scheme, a ciphertext originally produced by Alice and then transformed by the untrusted proxy for Bob cannot be transformed back to one for Alice. Naturally, a bidirectional PRE scheme allows transformations to be performed in both directions. The non-interactivity of a PRE scheme is another important property by which PRE schemes are characterized. A PRE scheme is said to be non-interactive if re-encryption keys $rk_{i \to j}$ can be generated without input or collaboration with user j.

Following the breakthrough identity-based encryption scheme from Boneh and Franklin [5], Green and Ateniese proposed a non-interactive, identity-based, single-hop PRE scheme in [12]. Built upon bilinear maps, the security of this particular scheme depends on the computational intractability of the Decisional Bilinear Diffie Hellman (DBDH) assumption, and has been proven to be CCA-secure in the random oracle model. Many subsequent efforts have been made to build upon this work, including replicating the scheme without the use of pairings [8], proving security in the standard model [13], and adding conditions which limit when and how re-encryption keys may be used [15].

TABLE I
COMPARISON OF THE RELEVANT PROPERTIES OF THE TWO SELECTED PRE SCHEMES USED IN THIS WORK.

PRE Property	*Identity-Based* [12]	*ElGamal-Schnorr* [8]
1. Unidirectional	✓	✓
2. Non-interactive	✓	✓
3. Multi-hop	✗	✗
4. Non-transitivity	✓	✓

Our selection of the two PRE schemes was motivated by their relative performance, flexibility, and security. As one of the core components for our content dissemination application, it was critically important to pick a scheme that could support fast client-side re-encryption and decryption. Given that the only two flavors of efficient PRE constructions in the literature rely on elements from either identity-based encryption (i.e. pairings) or ElGamal-type cryptosystems (i.e. modular arithmetic over groups with large primes), we chose one scheme from each of these respective categories to test in the context of our application.

The identity-based variant of our application design uses the original PRE scheme from Green and Ateniese [12], which builds upon the construction in [2]. The second PKI-based application design, which does not rely on pairings, is based on the work of Chow et al. [8]. A comparison of the relevant properties for both schemes is shown in Table I, and is succeeded in the following sections with specific details about each one that influenced our selection of a single candidate for our application.

C. Identity-Based PRE Overview

The identity-based PRE scheme (IBP2) from Green and Ateniese [12] is based on a modified version of Hierarchical Identity-Based encryption from Gentry and Silverberg [11] and the underlying identity-based encryption scheme designed by Boneh and Franklin using the Weil pairing [5]. The public parameters returned from the Setup() procedure consist of a generator g (in the symmetric pairing case) and corresponding element g^s, where s is the master secret key msk, as well as a tuple of hash functions used in the remaining procedures. Since unique identities take the place of public keys in identity-based schemes, the KeyGen() procedure requires the public parameters, the master secret key msk, and a unique identity id to output the corresponding secret key, sk_{id}. Encryption only requires the target recipient's identity id, public parameters params, and the actual message m. The ReKeyGen() procedure must be performed by the entity in possession of msk and requires the public parameters params, the source identity id_i, and the destination identity id_j. The output of this procedure is the re-encryption key $rk_{id_i \to id_j}$.

Re-encryption takes the public parameters params, level 1 ciphertext $c_{id_i}^1$ encrypted for the identity id_i, and re-encryption key $rk_{id_i \to id_j}$ as input and outputs a transformed level 2 ciphertext $c_{id_j}^2$ that is encrypted under the identity of user j. Decryption of this level 2 ciphertext simply requires the public parameters params, secret key of the identity id_j, sk_{id_j}, and the level 2 ciphertext $c^2_{id_j}$.

D. ElGamal-Schnorr PRE Overview

Motivated by the desire to avoid the computational difficulty of computing pairings to construct PRE schemes, Chow et al. [8] presented one of the few such constructions in the literature that does not rely on pairings. Instead, it combines a "hashed" CCA-secure ElGamal-type encryption scheme with the Schnorr signature scheme using a token-controlled approach to enable single-hop re-encryption. Much of the scheme is influenced by previous PRE proposals, e.g., [9], [19], and the public parameters consist of a generator g for the group $\mathbb{G} \subset \mathbb{Z}_q^*$, where the number of bits in q is equal to the security parameter κ, two parameters l_0 and l_1 which are polynomial in κ (for a message space that is l_0 bits), and a set of hash functions. The consumer key pairs consist of a pair of two secret keys $x_{i,1}$,$x_{i,2}$ and two public keys $pk_{i,1} = g^{x_{i,1}}$,$pk_{i,2} = g^{x_{i,2}}$. The ReKeyGen() operation generates a conversion key $rk_{i \to j}$ used to mask the original ciphertext encrypted with the secret keys of entity i to one that may be decrypted with only the secret key $x_{j,2}$ of entity j. The Encrypt() procedure masks the input message m with the hash of a group element generated based on the message and a randomly chosen string of length l_0. Concurrently, the token for this encrypted message is computed based on both secret keys of entity i (the producer), such that, during the ReEncrypt() procedure, the influence of these parameters are canceled out by $rk_{i \to j}$ so that entity j may use and verify the token.

The construction of both level one and two ciphertexts are publicly verifiable and delegator-safe, which means that the proxy may generate level two ciphertexts after verifying input level one ciphertexts without revealing or relying upon the secret keys of either party involved in the transformation. This is achieved by masking the secret keys with the hash of their public keys. Finally, the Decrypt() procedure for a level 2 ciphertext for user j extracts the information necessary to unmask the original plaintext using their secret key $sk_{j,2}$ and outputs the message.

IV. Content Dissemination Architecture

In this section we describe the reference architecture for our DRM application over CCN based entirely on the identity-based PRE scheme discussed in the previous section. We use PRE to obtain strong end-to-end content security. Furthermore, our design is general enough to be leveraged in any information or content-centric networking application.

A. Full PRE-Based Architecture

In an ideal setting, all content would be protected with the PRE scheme so as to (1) enable effective use of network caches, (2) prevent key and content leakages (i.e. users exposing decryption keys or the decrypted content), (3) simplify key management, and (4) improve overall security by true end-to-end encryption. This ecosystem is captured in Figure 1. With an architecture based entirely on PRE for content protection, each piece of content would be encrypted once by the producer using the identity of the respective content name. Since the producer is the only entity that can generate and store the corresponding secret key for this identity, the content remains secure as it is distributed throughout the network. When a consumer wishes to use the content they would need to request a corresponding re-encryption key from the producer. If privacy in terms of content-to-user linkability is a concern, the interest to request the key and and corresponding content could also be easily encrypted using the public keys of the producer and the consumer, respectively.

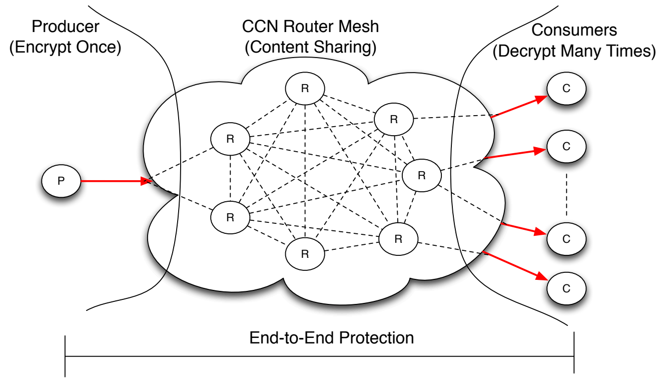


Fig. 1. Secure content distribution architecture context diagram.

After receiving the re-encryption key, the consumer's DRM application would re-encrypt the content and store the newly encrypted content in their device. In the event the device is corrupted and the original or transformed content are made publicly available, only the producer who owns the master secret key or the consumer who possesses the secret key for the transformed ciphertext may decrypt the content. Therefore, public disclosure of the content in either encrypted form will not put the original content at risk.

B. Improved Performance with Relaxed Security

To assess the feasibility of this fully PRE-based architecture in the context of CCNx [7], the open-source implementation of CCN by PARC, we implemented both of the PRE schemes discussed in Section III-B in Java. The identity-based scheme was implemented using the Java Pairing Based Cryptography library [6], and the ElGamal-Schnorr scheme was implemented using standard modular arithmetic operations provided by the Java `BigInteger` class. We instantiated each scheme with security parameters of similar strength - 256 bits for elliptic curve groups and 3072 bits for the Diffie Hellman groups, both of which are approximately providing 128 bits of security - and measured the time to perform the Encrypt(), ReKeyGen(), ReEncrypt(), and Decrypt() (for level 2 ciphertexts only) procedures. Our performance results (without pairing pre-computations) are depicted in Figure 2.

Even though the ElGamal-Schnorr scheme is not based on bilinear pairings over elliptic curves, the large security parameter needed to attain equivalent strength lead to worse performance than the identity-based scheme in our implementation. In addition, despite the clear difference in performance, it is still not practical to use either of them to encrypt large digital media, such as movies or music albums that are MBs or GBs in size.

This performance impediment led us to consider a hybrid encryption approach similar to those used by existing DRM technologies discussed in Section II. Our hybrid scheme uses the PRE scheme to protect an AES symmetric key for encrypting and decrypting premium content. The scheme is divided into a setup and online phase: the setup phase configures the producer and each consumer with the appropriate information (i.e. PRE public parameters and consumer secret keys) and the online phase handles the distribution of content when the application is live. For brevity, we omit the details of the setup phase as they can be implemented in a variety of ways, such as during software installation or device fabrication. The flow of messages during the online phase are shown in Figure 3. In this hybrid scheme, we refer to a content tuple (M', SK') as a piece of content encrypted with AES and the corresponding symmetric key that is encrypted with the PRE scheme.

As most PKI schemes use hybrid encryption to protect large files, and since the overhead from the ReKeyGen() and ReEncrypt() procedures is virtually negligible if only performed once for a symmetric key, we chose this encryption scheme to improve client-side performance without sacrificing much end-to-end content security. The details of the hybrid encryption procedures used in the online phase are captured in algorithms 4, 5, and 6, which list the explicit steps required for the Encrypt() and ReKeyGen() procedures in the producer and the ReEncrypt() and Decrypt() procedures in the consumer.

```
1: procedure PEncrypt($M, k$, params)
2:     $sk \overset{\$}{\leftarrow} \{0,1\}^k$
3:     $M' \leftarrow E(K, M)$
4:     $sk' \leftarrow$ Encrypt(params, $N(M)$, $sk$)
5:     return $(M', sk')$
6: end procedure
```

Fig. 4. Producer Encrypt() algorithm. $N(M)$ denotes the name of content M that would be used when issuing an interest for that particular piece of content.

```
1: procedure PReKeyGen($N(M)$, params, $A$)
2:     $K_M \leftarrow$ KeyGen($N(M)$, params)
3:     $rk_{M \to A} \leftarrow$ ReKeyGen(params, $K_M$, $N(M)$, $A$)
4:     return $(rk_{M \to A})$
5: end procedure
```

Fig. 5. Producer ReKeyGen() algorithm. $N(M)$ denotes the name of content M that would be used when issuing an interest for that particular piece of content, and A is the public identity of the requesting consumer.

C. In-Network Transformation

In our original architecture we assume the producer is always available to generate a re-encryption key. However, as argued in [17], this may not always be true in the real world. When the producer is offline, the use of hybrid encryption using PRE can be leveraged to enable continual access to content, which is not feasible with traditional DRM technologies. One way to support this feature would be to introduce intermediary nodes, e.g., retailers or trusted distribution nodes, between the producer and consumers that serve the re-encryption keys by generating them on demand. In other words, such nodes could preemptively request and re-encrypt content keys on behalf of users so that, in the event the original producer is offline, the consumer can still obtain their content re-encryption key from these nodes [1].

PRE also enables us to go farther to have these intermediary nodes perform the content re-encryption on behalf of the user. Such nodes equipped with appropriate computational power can re-encrypt the content, or the symmetric content encryption-keys in the hybrid case, for individual users. This would enable all content to be delivered by the network to an interested consumer to be ready for immediate decryption by that consumer's private key.

```
1: procedure CDecrypt(params, $K_A$, $rk_{M \to A}$, $M'$, $sk'$)
2:     $sk'' \leftarrow$ ReEncrypt(params, $rk_{M \to A}$, $sk'$)
3:     $sk \leftarrow$ Decrypt(params, $K_A$, $sk''$)
4:     $M \leftarrow D(sk, M')$
5:     return $M$
6: end procedure
```

Fig. 6. Consumer Decrypt algorithm that makes use of the internal PRE ReEncrypt() and Decrypt() procedures. Note that $E(\cdot,\cdot)$ and $D(\cdot,\cdot)$ refer to AES symmetric-key encryption and decryption, respectively, and that K_A is the secret key for consumer A generated from the (offline) KeyGen() procedure.

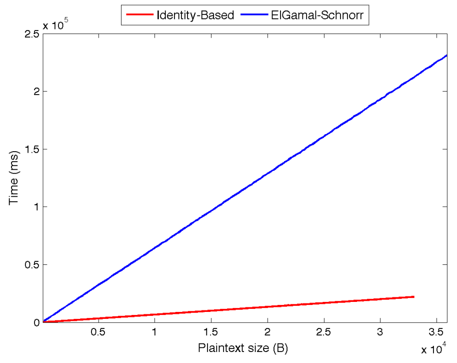
V. Application Implementation for CCNx

To assess the correctness and implementation efficiency of the proposed architecture, we developed and tested a prototype implementation of the full PRE-based architecture presented in Section IV-A as a Java CCNx application.

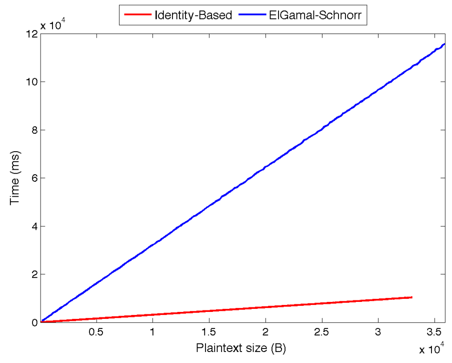
In the implementation, there is a single content producer and multiple consumer processes that are spawned with a running instance of `ccnd`, the daemon process that implements the CCN protocol at the network layer in the CCN stack. Prior to initializing these processes, the public parameters used in the PRE scheme are generated and saved to persistent storage (i.e., a file on the disk). When the producer process is started, it first reads the public parameters and initializes the rest of the internal components needed for PRE functions, registers the `ccnx:/p/` prefix, and waits for incoming interests from consumers. Consumer processes also utilize same public parameters to initialize the PRE functions.

The remaining five steps in the identity-based PRE scheme, KeyGen(), Encrypt(), ReKeyGen(), ReEncrypt(), and Decrypt(), are performed online between the consumer and

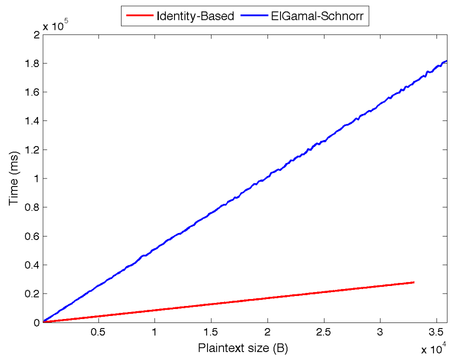
[1]This modification requires that the PRE scheme be multi-hop to enable level $n > 2$ ciphertexts to be generated.



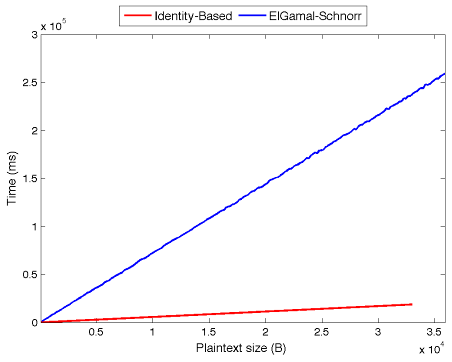
(a) Encrypt() time comparison.



(b) ReKeyGen() time comparison.



(c) ReEncrypt() time comparison.



(d) Decrypt() time comparison.

Fig. 2. Comparison of the Encrypt(), ReKeyGen(), ReEncrypt(), and Decrypt() times for the identity-based scheme from [12] and ElGamal-Schnorr scheme from [8].

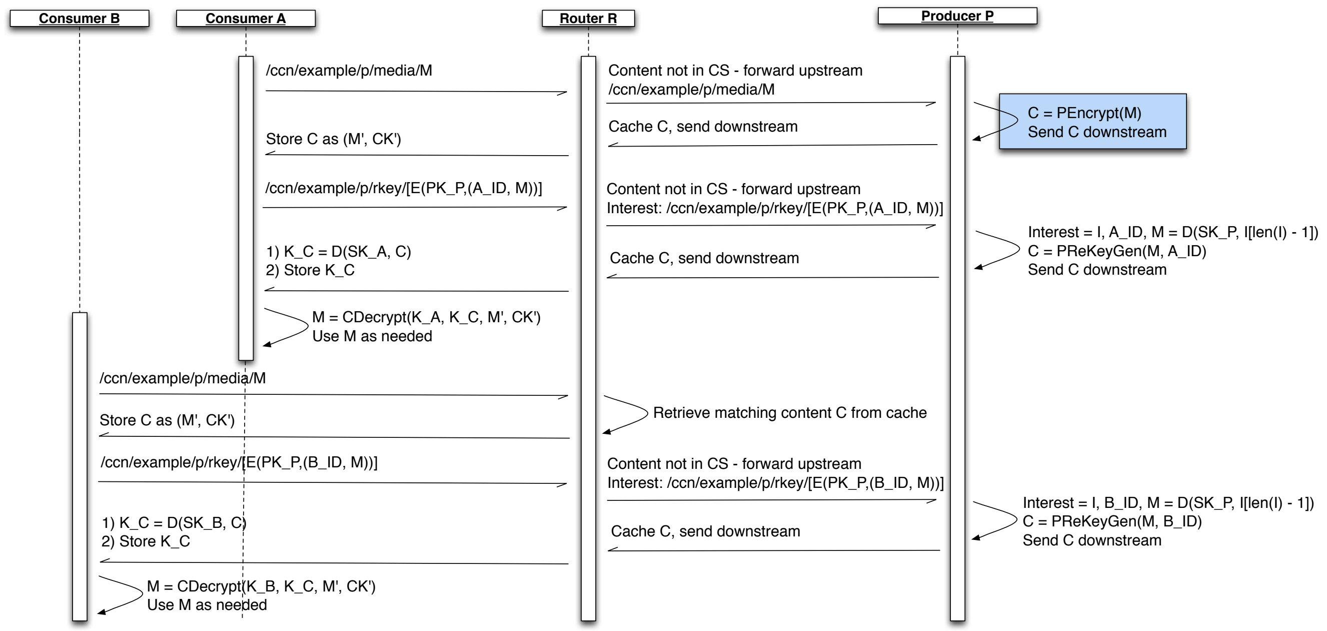

Consumer B
Consumer A
Router R
Producer P
/ccn/example/p/media/M
Content not in CS - forward upstream
/ccn/example/p/media/M
C = PEncrypt(M)
Send C downstream
Cache C, send downstream
Store C as (M', CK')
/ccn/example/p/rkey/[E(PK_P,(A_ID, M))]
Content not in CS - forward upstream
Interest: /ccn/example/p/rkey/[E(PK_P,(A_ID, M))]
Interest = I, A_ID, M = D(SK_P, I[len(I) - 1])
C = PReKeyGen(M, A_ID)
Send C downstream
Cache C, send downstream
1) K_C = D(SK_A, C)
2) Store K_C
M = CDecrypt(K_A, K_C, M', CK')
Use M as needed
/ccn/example/p/media/M
Retrieve matching content C from cache
Store C as (M', CK')
/ccn/example/p/rkey/[E(PK_P,(B_ID, M))]
Content not in CS - forward upstream
Interest: /ccn/example/p/rkey/[E(PK_P,(B_ID, M))]
Interest = I, B_ID, M = D(SK_P, I[len(I) - 1])
C = PReKeyGen(M, B_ID)
Send C downstream
1) K_C = D(SK_B, C)
2) Store K_C
Cache C, send downstream
M = CDecrypt(K_B, K_C, M', CK')
Use M as needed

Fig. 3. Sequence diagram depicting the flow of traffic required to retrieve a piece of content M by two different consumers. The $params$ argument is omitted from all PRE-related procedures for brevity. Also, the PEncrypt() procedure is highlighted to emphasize that it only happens once for each piece of media.

the producer processes. Each consumer process requests a single piece of content from the producer, writes the content to disk, and then terminates upon completion. To accomplish this simple task, the consumer and producer complete the set of interactions below. Note that in all cases the producer always uses a CCNOutputStream instance to transfer raw binary data to the consumer, and the consumer uses a paired CCNInputStream to read this data.

1) The consumer process requests its secret key in the PRE scheme corresponding to their identity, which is stored internally as an array of bytes, by encoding the byte array into a Base64 string A_{ID} that is CCN URL-friendly. The consumer builds and issues the interest `ccnx:/p/sk/`A_{ID}. Upon receipt, the producer decodes the consumer identity and then runs the KeyGen() procedure using the encoded identity to construct a secret key K_A. Finally, the producer encrypts the raw bytes of K_A using the identity of the consumer and then sends the resulting ciphertext downstream.
2) The consumer encodes the target content name M as a Base64 string $N(M)$ and then builds and issues the interest `ccnx:/p/media/`$N(M)$. Upon receipt, the producer decodes the content name $N(M)$ as the last name component and performs the following check. If a file with the name $N(M)$ exists, the contents of the file are read and encrypted in chunks using the PRE Encrypt() function and then returned downstream. If the name $N(M)$ does not match the name of a file, then the producer treats the bytes of $N(M)$ as an unsigned integer n and generates n random bytes to be encrypted and returned to the consumer to generate synthetic traffic.
3) The consumer creates an instance of a key request object and stores their identity and the name of the target content $N(M)$ in the two fields. Then, the consumer serializes this object to a byte array, encrypts the byte array using the identity of the producer, encodes the corresponding ciphertext into a Base64 string C, and finally, builds and issues the interest `ccnx:/p/rkey/`C. The producer then decodes C from the interest, decrypts the ciphertext using their secret key, and then deserializes the object to obtain the consumer's identity and target piece of media $N(M)$. The producer then runs the ReKeyGen() step to generate a re-encryption key, encrypts the raw bytes of the key using the identity of the consumer, and returns the raw bytes of this key to the consumer.
4) The consumer finishes the transaction by using the re-encryption key to re-encrypt the encrypted piece of content using the PRE ReEncrypt() procedure, and then decrypts the content using their secret key K_A.

Note that we omitted any consumer authentication during for the first step as it will happen only once when a user first registers to the system. However, the access control and authentication in later steps are implicit as only the intended consumer can decrypt the content with its private key.

VI. Preliminary Evaluation

By leveraging PRE in the context of CCN we can achieve virtually the same benefits of current content dissemination and DRM technologies while reducing the overall infrastructure requirements and improving the protection of content during its lifetime. With the ability to cache encrypted content in the network that can only be decrypted by the producer (they possess the secret key associated with the identity $N(M)$) or the consumer upon re-encryption, the proposed scheme is particularly appealing to ICNs as it can serve individually encrypted content and take full advantage of in-network caches at the same time.

The use of PRE also provides strong end-to-end security for all content. Content objects always stay encrypted during transport and at rest, including the consumers' persistent storage. The only time content objects need to be decrypted is within the DRM enabled software (e.g., media player) at the time of consumption. Moreover, this architecture significantly simplifies key management at consumer devices since only one key needs to be safeguarded: the user's secret key.

In our hybrid encryption variant, using an AES key size that is comparable in security to the PRE scheme would ensure that the security guarantees are not degraded due to the use of symmetric encryption. Recall that each symmetric key in a content tuple that is cached in the network is encrypted under the public identifier of $N(M)$. Since the producer is the only entity who possesses the corresponding secret key, the network is free to cache and transfer this content tuple in any way necessary. Users are even free to share content tuples among their own devices or with different users offline. For a single user, this means that one may transfer content tuples and wrapped keys from one device to another, both of which have the user's secret key, and enjoy access to them without having to contact the producer. The proposed solution relies on the proven CCA-security of the identity-based PRE scheme to enable routers and users to freely distribute encrypted content throughout the network without the risking unauthorized access to the content.

The proposed scheme fares well for routers, producers, and consumers at the same time. Routers need not change operation or treat encrypted media any differently than ordinary content, and so there is no performance impact that could affect the service. Producers can make best use of the caches in the network without sacrificing the security. As a result, consumers can enjoy high quality of service with reduced transmission delays.

In order to be able to decrypt and use a specific piece of content, a consumer needs to issue an interest for the content re-encryption key. As shown in Figure 3, such interests may be sent asynchronously and even before the content interest to reduce the overall latency. Moreover, in most cases the consumers may potentially start decrypting and using chunks of the content without waiting to retrieve the content in its entirety. This feature is particularly appealing for media distribution applications. Since the symmetric key used to encrypt the entire piece of content (i.e. all chunks of said content) only needs to be re-encrypted once using the PRE ReEncrypt() procedure, the re-encryption overhead is a negligible at around 200ms on a standard personal computer.

VII. Conclusion

In this paper, we presented an architecture for secure content distribution over CCN that leverages proxy re-encryption for enhanced security and content individualization. It reduces the overall bandwidth consumption and requires fewer message transactions than existing solutions. We showed that two specific PRE schemes, one based based on pairings and the other not, are still computationally impractical for encrypting large pieces of content. As a result, a hybrid PRE-based encryption scheme can be used for secure and efficient distribution of content over CCN today with negligible performance penalties. Fortunately, with the continuous improvement of cryptographic primitives, as well as the advances in available commodity hardware, it is likely that the full PRE-based scheme with more pronounced security benefits will also be practical in near future – perhaps just in time to meet the adoption of a new Internet architecture.

References

[1] Vijay Kumar Adhikari, Yang Guo, Fang Hao, Matteo Varvello, Volker Hilt, Moritz Steiner, and Zhi-Li Zhang. Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery. *In the Proceedings of IEEE INFOCOM 2012* (2012).

[2] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved Proxy Reencryption Schemes with Applications to Secure Distributed Storage. *In the 12th Annual Network and Distributed System Security Symposium* (2005), 29-43. Full version available at http://eprint.iacr.org/2005/028.

[3] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-Policy Attribute-Based Encryption. *2007 IEEE Symposium on Security and Privacy, SP'07* (2007).

[4] M. Blaze, G. Bleumer, and M. Strauss. Divertible Protocols and Atomic Proxy Cryptography. *In Proceedings of Eurocrypt 98* **1403** (1998), 127-144.

[5] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *Advances in Cryptology - CRYPTO 2001, Springer Berlin Heidelberg* (2001).

[6] A. De Caro and V. Iovino. jPBC: Java Pairing Based Cryptography. *IEEE Symposium on Computers and Communications (ISCC)* (2011).

[7] CCNx Project. Available online at http://www.ccnx.org/. Last accessed: 8/5/13.

[8] S. Chow, J. Weng, Y. Yang, and R. Deng. Efficient Unidirectional Proxy Re-Encryption. *Progress in Cryptology - AFRICACRYPT 2010*. Springer Berlin Heidelberg (2010), 316-332.

[9] R. H. Deng, J. Weng, S. Liu, and K. Chen. Chosen-Ciphertext Secure Proxy Re-Encryption without Pairings. *CANS*. Spring Lecture Notes in Computer Science Volume **5339** (2008), 1-17.

[10] The Despotify Project (2012). Available online at http://despotify.sourceforge.net/. Last accessed: 8/1/13.

[11] C. Gentry and A. Silverberg. Hierarchical ID-Based Cryptography. *Advances in Cryptology - ASIACRYPT 2002*. Springer Berlin Heidelberg (2002), 548-566.

[12] M. Green and G. Ateniese. Identity-Based Proxy Re-Encryption. *Applied Cryptography and Network Security. Springer Berlin Heidelberg* (2007).

[13] T. Isshiki, M. H. Nguyen, and K. Tanaka. Proxy Re-Encryption in a Stronger Security Model Extended from CT-RSA2012. *In Topics in Cryptology CT-RSA 2013*. Lecture Notes in Computer Science Volume 7779 (2013), 277-292.

[14] S. Kamara and K. Lauter. Cryptographic Cloud Storage. *Financial Cryptography and Data Security. Springer Berlin Heidelberg* (2010), 136-149.

[15] K. Liang, Z. Liu, X. Tan, D. S. Wong, and C. Tang. A CCA-Secure Identity-Based Conditional Proxy Re-Encryption without Random Oracles. *In Information Security and Cryptology ICISC 2012*. Lecture Notes in Computer Science Volume 7839 (2013), 231-246.

[16] J. Lotspiech, S. Nusser, and F. Pestoni. Anonymous Trust: Digital Rights Management using Broadcast Encryption. *Proceedings of the IEEE* **92.6** (2004), 898-909.

[17] S. Misra, R. Tourani, and N. E. Majd. Secure Content Delivery in Information-Centric Networks: Design, Implementation, and Analyses. *In Proceedings of the 3rd ACM SIGCOMM Workshop on InformationCentric Networking (ICN 2013)* (2013).

[18] Sandvine, Global Internet Phenomena Report - Spring 2012. Located online at http://www.sandvine.com/downloads/documents/Phenomena_1H_2012/Sandvine_Global_Internet_Phenomena_Report_1H_2012.pdf. Last accessed: 8/1/13.

[19] J. Shao and Z. Cao. CCA-Secure Proxy Re-Encryption without Pairings. *Public Key Cryptography*. Springer Lecture Notes in Computer Science Volume **5443** (2009), 357-376.

[20] R. Wang, Y. Shoshitaishvili, C. Kruegel, and G. Vigna. Steal This Movie - Automatically Bypassing DRM Protection in Streaming Media Services. Available online at http://seclab.cs.ucsb.edu/media/uploads/papers/sec13-final215.pdf.

[21] H. Xiong, X. Zhang, W. Zhu, and D. Yao. CloudSeal: End-to-End Content Protection in Cloud-based Storage and Delivery Services. *Security and Privacy in Communication Networks. Springer Berlin Heidelberg* (2012), 491-500.