Beyond PKI: The Biocryptographic Key Infrastructure

W.J. Scheirer, W. Bishop and T.E. Boult

Abstract Public Key Infrastructure is a widely deployed security technology for handling key distribution and validation in computer security. Despite PKI's popularity as a security solution, Phishing and other Man-in-the-Middle related attacks are accomplished with ease throughout our computer networks. The major problems with PKI come down to trust, and largely, how much faith we must place in cryptographic keys alone to establish authenticity and identity. In this chapter, we look at a novel biometric solution that mitigates this problem at both the user and certificate authority levels. More importantly, we analyze the problem of applying unprotected biometric features directly into PKI, and propose the integration of a secure, revocable biometric template protection technology that supports transactional key release. A detailed explanation of this new *Biocryptographic Key Infrastructure* is provided, including composition, enrollment, authentication, and revocation details. The BKI provides a new paradigm for blending elements of physical and virtual security to address network attacks that more conventional approaches have not been able to stop.

1 Introduction

To motivate the contribution of this paper, we first turn to the technology that underpins the problem. Public Key Infrastructure (PKI) [10, 1, 9, 30] has been a popular, yet often maligned technology since its widespread adoption in the 1990s. PKI (Fig-

W. J. Scheirer and T.E. Boult
Vision and Security Technology Lab, Department of Computer Science, University of Colorado, Colorado Springs, CO 80918-7150, USA
e-mail: wjs3@vast.uccs.edu, tboult@vast.uccs.edu

W. Bishop
Securics, Inc., Colorado Springs, CO 80918-7150
e-mail: bill.bishop@levault.net

Pre-print of chapter to appear in the book *Security and Privacy in Biometrics*. The original publication is available at http://www.springerlink.com.

ure 1) is the infrastructure for handling the complete management of digital certificates (x.509 compliant), which contain a piece of trusted information: a public key. PKI attempts to solve an important problem in key management - namely, how can Alice verify that Bob's public key is really Bob's? Addressing this problem remains a paramount concern, as the Internet has experienced an explosion of successful Phishing and other Man-in-the-Middle attacks in recent years. Users of networks, both those well-informed and those blissfully ignorant of security protocols, routinely ignore security provisions put into place by PKI to guard them against such attacks (how often have *you* blindly clicked through a browser certificate warning?). Sadly, providers of information security services are also to blame, by providing PKI as a catch-all security solution, and ignoring its limitations.

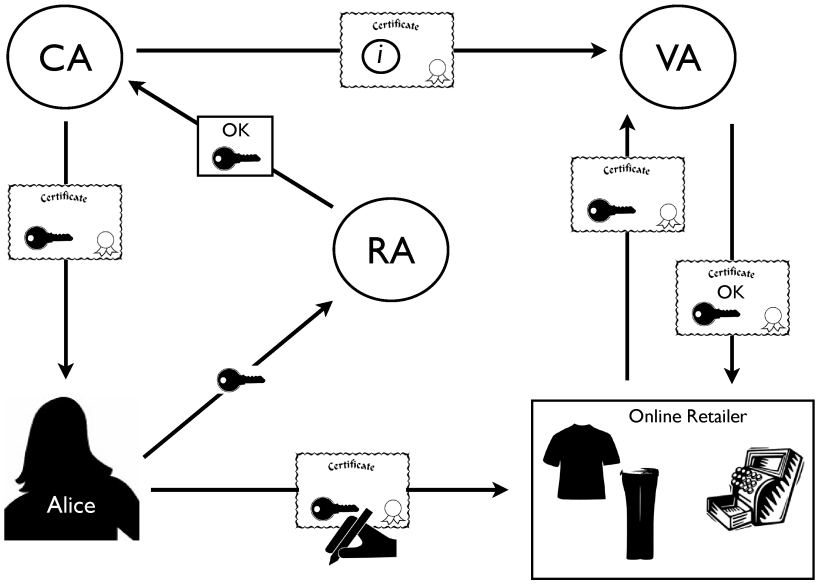

Fig. 1 An overview of Public Key Infrastructure. A website owner, Alice, wants to obtain a digital certificate for her web store. She applies to a Registration Authority (RA) with her public key for the certificate. The RA confirms Alice's identity, and then contacts the Certificate Authority (CA), which issues the certificate. With a valid certificate, Alice can now digitally sign off on contracts involving her web store. Alice's identity can be confirmed when visitors to her web store present her certificate to a Validation Authority (VA), which receives information about issued certificates from the CA.

The problems with PKI [16, 15] are well-known, and have remained mostly unsolved thus far. Ellison and Schneier [14] specifically highlight a series of identity related PKI risks by asking the following questions:

1. Who do we trust, and what for?
2. Who is using my key?

3. Is a name a unique identifier?
4. Is the user part of the security design?
5. How did the CA identify the certificate holder?

The overarching criticism stems back to the notion of trust in a PKI system - why would we place any trust in a system with entities (both certificate authorities and users) signifying their identities with only randomly generated keys? A practical and recent attack [35, 36] highlights the ease with which a rogue certificate authority can be established, using an MD5 hash collision attack against the digital signatures used for certificate validation (illustrated in Figure 2). With all trust being placed in digital signatures, presumedly derived from *legitimate* keys, there is no way to tell the difference between a Man-in-the-Middle and a legitimate site - if a collision has been located that matches the legitimate certificate's signature. While MD5 enables the attack in this instance, the entire infrastructure will always be susceptible to trust related attacks if any cryptographic component is flawed. What if we extend the notion of trust beyond keys to include identity specific information?

By adding a second factor, we can mitigate these inherent identity related trust problems with PKI. Biometrics, those methods of uniquely recognizing humans based on physiognomy or behavior have become ubiquitous in many areas of technology and society, and are mature to the point of being generally accepted as valid and useful security tools. For PKI, the addition of biometric data has a very attractive feature - if a user or certificate authority presents a key and biometric during some action, we have more confidence that this action is legitimate (but this does not absolutely prove that the owner of the key and biometric actually performed the action - stolen keys and spoofing attacks are not prevented by two-factor authentication). With biometrics, we have improved *non-repudiation*. A series of related concerns follow the trust problem: the security of the verifying computer, certificate authority establishment, and general certificate issue. With the proper protocols including a biometric component, we can address each of these.

But to solve these problems correctly, we cannot simply use standard biometric templates (the data representation of the collected biometric feature) embedded within x.509 certificates, because a *revocation* of raw biometric data can only happen for a limited number of times (we have 1 face, 2 irises, 10 fingers). ISO/IEC 19794-2 standard templates, while being an abstract representation of the original biometric features, are still effectively invertible [8]. Moreover, unprotected biometric data that is even under the control of "trusted" entities is still vulnerable to attack. To understand why, we first must take a look at Figure 3, which depicts what we term the *Biometric Dilemma*. In essence, as the use of biometrics increases, so does the chance for compromise. If a malicious attacker, Mallory, wishes to impersonate Alice at an area of high security, she can obtain the exact biometric data she needs from a different, much lower security area. How well might Alice's gym be protecting her biometric data that she uses to access her locker? Low-hanging fruit is plentiful, and can often be obtained legitimately. In 2001, the state of Colorado tried to sell its DMV face and fingerprint databases [22] to anyone who wanted to buy them. The resulting protests moved the data back off the market, but the state still offers them to any requesting law enforcement agency.

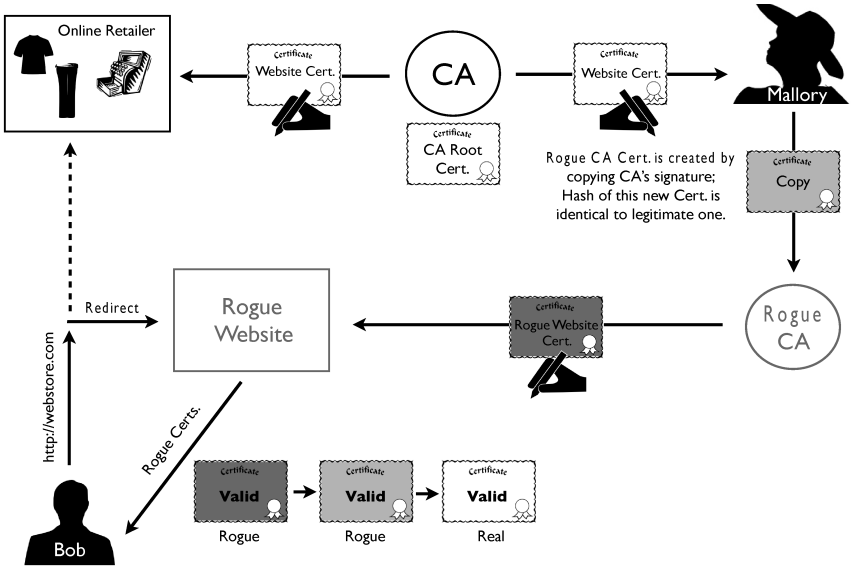

Fig. 2 The chosen-prefix collision attack of [36]. A malicious attacker, Mallory, wishes to deploy a rogue website that after a redirect attack will look identical to a legitimate online retailer, with valid certificates. To accomplish this, she requests a legitimate website certificate from a real CA (cert. in upper right of the diagram). Mallory crafts a rogue CA certificate (light gray cert. in the diagram) by copying the signature from her legitimate website certificate to an illegitimate CA certificate. She is able to do this by creating a CA certificate that will collide with the legitimate signature when the same hash function is applied. Mallory can now create a rogue website certificate (dark gray cert. in the diagram) that bears the the online retailer's identity, but contains a different public key and is signed by the rogue CA. When Bob is redirected to the rogue website, he can successfully validate in sequence the rogue website's certificate, the rogue CA's certificate and the real CA's root certificate. The transaction will appear to be legitimate.

A second, and equally dangerous attack is what we term the *doppelganger threat*, which takes advantage of the operational security performance characteristics of the underlying biometric matching algorithm. If the False Accept Rate (FAR) of a system is 1 in X attempts, then a doppelganger attack consists of trying more than X different biometric samples. This attack is the biometric equivalent of a dictionary attack. Once again, one need not break the law to gather the data necessary for a successful attack. The hundreds of thousands of fingerprints that are publicly available (including four well known algorithm challenges [5], testing data from NIST [27], and even 100,000+ prints being offered by a private company [13]) provide at least a basic doppelganger dictionary for anyone willing to spend a small amount of money.

Previous work on the integration of biometrics into PKI has not considered the biometric dilemma or doppelganger attack, favoring a simplistic unsecured application of biometrics. Proposed standards for PKI with biometrics go back to the mid 1990s, with recommendations from the defense space [29], commercial Inter-

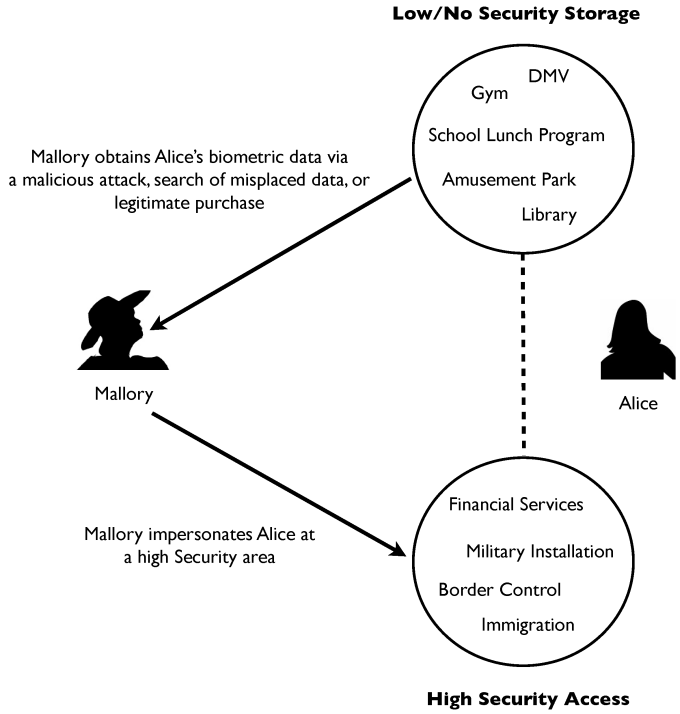

Fig. 3 The biometric dilemma: as the usage and storage of biometric data increases, so does the vulnerability. A malicious attacker, Mallory, may obtain Alice's biometric data with relative ease from a low/no security source (possibly through a legitimate transaction) to attack a high security target. The indiscriminate use of biometrics makes this threat possible, and if not addressed, limits the integration of biometrics into existing security infrastructure like PKI.

net related interests [40], and NIST [23]. Both Benavente [3] and Martinez-Silva et al. [25] suggest augmenting x.509 certificates with BioAPI [4], which provides the templates and matching capability needed to use the biometric data. Dawson et al. [11] also recommend augmenting x.509 certificates with biometric data, as part of a much larger defense-in-depth approach to authentication. Finally, Kwon and Moon [24] suggest the use of a biometric PKI scheme for border control applications. These previous standards recommendations and research works do not place adequate safeguards around biometric data. All store and match unprotected templates, and have no facility for biometric template revocation and re-issue.

In response to the threat of permanent biometric feature compromise, very recent research [18] has emerged from both the pattern recognition and cryptography communities to address the problem of *biometric template security*. Solutions to this problem create a transformation of original biometric features that can be matched in an encoded space, and revoked and reissued if a compromise is detected, in much the same manner as a traditional password or PIN. For unattended

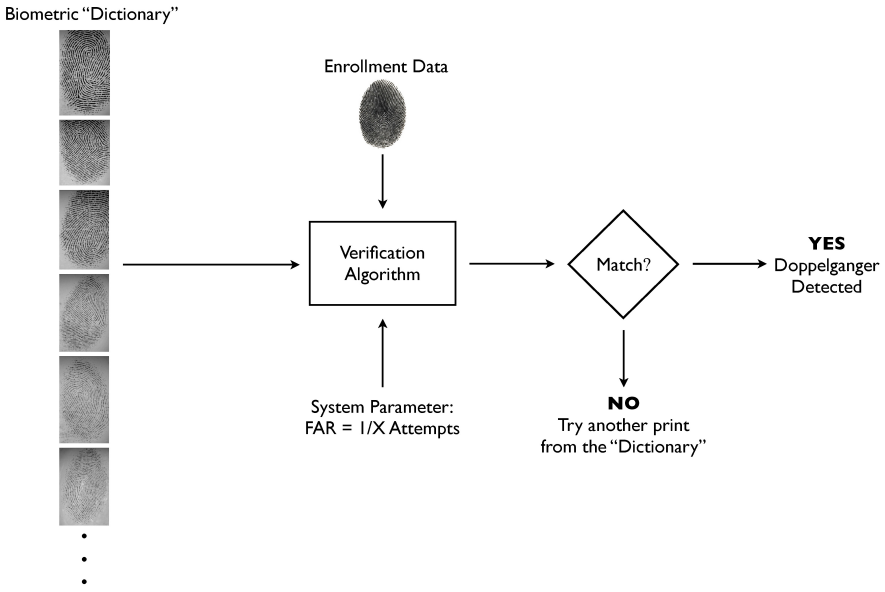

Fig. 4 The Doppelganger Threat. What happens when a large database of biometric data is stolen, hacked, or sold? For example, consider a database containing unique records for 4 Million individuals. If the matching system is operating at a False Accept Rate (FAR) of 1 in 1,000, a malicious attacker in possession of the database may, on average, be given a choice of approximately 4,000 identities to use to compromise the matching system. Even at a FAR of 1 in 1,000,000, the attacker still gets four choices, on average, to compromise the matching system.

network authentication, the risk of spoofing is greatly reduced by secure templates. Unique templates can be generated for different domains and applications, making a template harvested by an attacker at one domain useless when applied to a different domain. This addresses the biometric dilemma described above. A wide variety of approaches have been proposed in the literature, including non-invertible transforms [28], fuzzy commitment [20], fuzzy vaults [19], fuzzy extractors [12], BioHashing [37], and revocable biotokens [38]. Even more interesting for trusted data transfer is that certain classes of these schemes support key release upon successful matching. *Key-binding* biometric cryptosystems bind key data with the biometric data. *Key-generating* biometric cryptosystems derive the key data from the biometric data. Both classes support a key release that may be used for cryptographic applications, including standard symmetric key cryptography, where key storage is problematic.

Secure templates enable completely new ways to transfer secret information. Consider a key-binding biometric cryptosystem where the key can be bound even after enrollment and which also provides public secure templates that can be used in the same manner as a public key. With these components, we have the building blocks for a *Biocryptographic Key Infrastructure*. The primary benefit (Figure 5) of

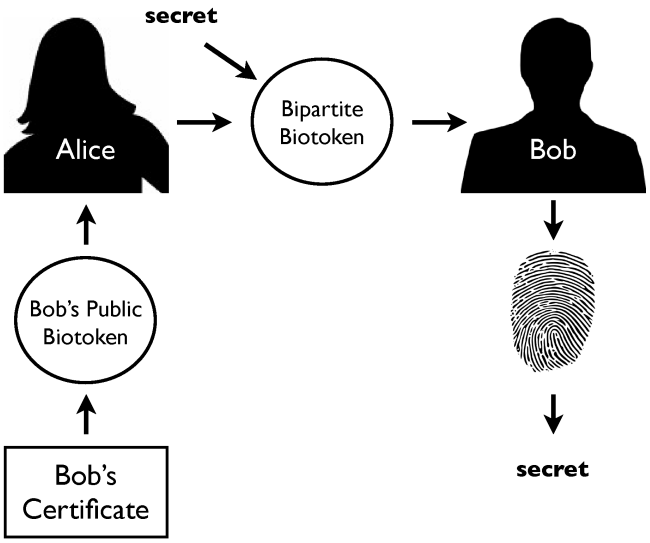

Fig. 5 The primary benefit of a Biocryptographic Key Infrastructure: the ability to store *public* biotokens in digital certificates. Any entity in the infrastructure can send secret data that only the owner of the biotoken can unlock. In this example, Alice wants to convey a secret message to Bob. Bob's public biotoken can be retrieved from his certificate, allowing Alice to transform it into a bipartite biotoken, which conveys an embedded secret. Alice has assurance that an identity must be present to unlock the secret – not just a key.

such an infrastructure is the ability to store public templates (referred to in this article as *biotokens*) in x.509 compliant digital certificates. Through a transformation of a public template with an embedded secret (the transformed template is referred to in this article as a *bipartite biotoken*) by an entity that wants to convey information to the template's owner, secure key exchange and unique transactions can be supported. Only the owner of the public biotoken can unlock the secret. By adding this biometrically derived data to a certificate, an additional component must validated (with the help of a validation authority in possession of enrollment data), making attacks such as the one described above much more difficult to perpetrate.

The rest of this chapter introduces the details for the Biocryptographic Key Infrastructure. In Section 2, the fundamental biometric requirements are defined, including the properties necessary for protecting the biometric data, secure key release, and revocation support. In Section 3 our full infrastructure is described, including a description of the overall composition, the enrollment process for both biometric certificate authorities and users, the certificate validation process, authentication protocols, and revocation and re-issue procedures. Section 4 takes these ideas into the real world with suggestions on how BKI can be applied in place of PKI with stronger security. In the concluding remarks of Section 5, we make the

case for standards consideration of revocable biometric template technologies and BKI.

2 Fundamental Biometric Requirements for BKI

Many different secure template technologies exist, but not all are appropriate for use in a PKI-like framework. To be useful for PKI, a secure template technology must possess the following properties:

1. Cryptographically strong protection of the underlying biometric features.
2. The ability to revoke and re-issue the template.
3. Nested re-encoding, allowing a hierarchy of templates to be generated from a single base template.
4. Support for public templates that cannot be used to match other public templates, and private templates that are generated dynamically from a biometric sample during matching and immediately discarded following.
5. Key-binding capability without the need of intervention by the person associated with the template.

The first and second properties ensure resilience against the biometric dilemma and doppelganger attacks by not exposing the original biometric features during matching, allowing the creation of application specific templates, and rendering a compromised template useless by replacing it with a new template via different cryptographic keys and/or transformations. Cryptographically strong protection implies that it should not be feasible for an attacker to retrieve the original biometric features from a compromised secure template without knowledge of relevant transformation information (such as keys used to protect the biometric data). The third, fourth, and fifth properties guarantee the PKI-like operations we'd like our secure templates to possess to be useful for protocols common to PKI.

Throughout the rest of this chapter, we will use *revocable biotokens* [38, 32, 33] as a case study for the BKI described herein, though any secure template technology supporting the five aforementioned properties could be used. To date, only revocable biotokens support all five. Some secure template technologies are appropriate for authentication protocols [41, 39] but lack support for key transfer, while others support key transfer [7] but lack the flexibility for unique transactions. A scheme such as the one presented in [21] could be used to support some of the functionality of BKI (namely requirement 5), though it does not support nested re-encoding and is susceptible to attack[1]. We briefly introduce the fundamentals for revocable biotokens in the remainder of this section as an illustration of the biometric requirements. Interested readers should refer back to the prior published work on revocable

[1] The template described in [21] consists of a secret key + error correction θ_{ps} XOR*ed* with shuffled biometric data θ_{canc}, yielding θ_{lock}. If an attacker knows θ_{ps}, they can simply XOR it with θ_{lock}, yielding θ_{canc}, which can be used by the attacker to match from that point forward. This is a straightforward application of the SKI attack [31].

biotokens [38, 32, 33] for modality specific algorithm details and security analysis, though that level of depth is not necessary to understand the higher level concepts that enable BKI.

The notion of data splitting to support revocable biotokens was introduced by Boult et al. [38]. In general, encrypted biometric data cannot be matched, because of the unstable nature of the data, which can vary as a function of environment, age, and acquisition circumstances. However, many biometric modalities yield features that can be split into stable and unstable (or residual) components. By encrypting the stable component, matching can occur in the encrypted space because this portion of the data is not impacted by any instability at the bit level. Additional residual matching adds accuracy [38]. Using this knowledge, and the concept of public key cryptography, we can develop the re-encoding methodology for revocable biotokens. The re-encoding property, introduced by Scheirer and Boult [32], is essential for supporting a viable transactional framework - tokens with unique data must be generated quickly and automatically to support cryptographic transactions (such as session key exchange). The *bipartite biotoken* form of a revocable biotoken supports data-binding (key-binding) at the transactional level. Bipartite biotoken generation from a stored biotoken allows the required data release when only matching against tokens generated from data derived from original biometric features during the course of the transaction.

Assuming the biometric produces a value v that is obfuscated via scaling and translation to $v' = (v-t)*s$, the resulting v' is split into the stable component q, and the the residual component r. The amount of stable and unstable data is a function of the biometric modality being considered. In a basic scheme, for a user j, their residual $r_j(v')$ is left un-encoded. For the initial *transformation* $w_{j,1}(q_j(v'), T_1)$ some transformation function T (which may be a strong hash function like SHA-256 that is minimally impacted by collision attacks, or another application of public key cryptography) is applied. For nested re-encodings, w_j is re-encoded using further transformations, creating a unique new encoding for each hash or key that is applied: $w_{j,1}(q_j(v'), T_1)$, $w_{j,2}(w_{j,1}, T_2), \ldots, w_{j,n}(w_{j,n-1}, T_n)$

If public key cryptography is used for every transformation, the nesting process can be securely invertible if the private keys all the way back to the first stage (the root) of encoding are available. Partially inverting the nesting facilitates revocation and automatic re-issue of the biotoken, which is an attractive feature for the BKI system. A tree introducing our standard hierarchy of biotokens with descriptions for each is shown in Figure 6. We note that any public keys used for encoding here are strictly for this biotoken generation process, and are different from the keys contained in the user's certificate. With this nesting in mind, we can define three properties for the bipartite biotoken:

1. Let B be a secure biotoken, as described in [38]. A bipartite biotoken B_B is a transformation $bb_{j,k}$ of user j's kth transformation of B. This transformation supports matching in encoded space of any bipartite biotoken instance $B_{B,k}$ with any secure biotoken instance B_k for the biometric features of a user j and a common series of transforms $T_1, T_2, \ldots, T_k$.

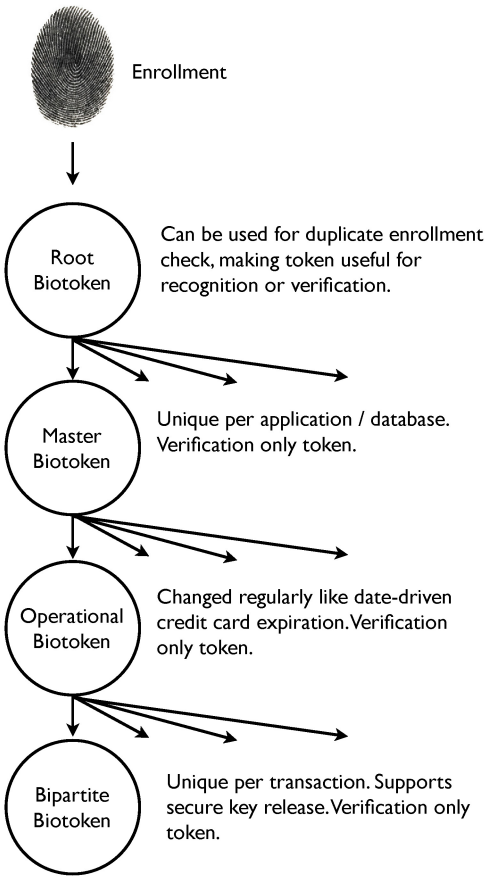

Fig. 6 The biotoken issue/re-issue tree. Biotokens can be re-encoded, starting from the root token generated at enrollment time, through subsequent applications of public key encryption (supporting automatic revocation and re-issue), or a hash function.

2. The transformation $bb_{j,k}$ must allow the embedding of some data d into B_B, represented as: $bb_{j,k}(w_{j,k}, T_k, d)$.
3. The matching of B_k and $B_{B,k}$ must release d if successful.

The design of bipartite biotokens that satisfies the above properties is an extension of the fuzzy vault [19] concept, where a polynomial embedding hides the data d. The bipartite representation implements Reed-Solomon (RS) for error correction, and does not store the points at which the embedded polynomial is evaluated. For efficiency, we choose to work over a Galois Field of size 2^8, where the coefficients and evaluation points are all 8 bit quantities. We represent the data d to be stored as a K-byte block, with E bytes of error correction, yielding a total payload block $N = K + E$. The polynomial encodes the N bytes of data. The Reed-Solomon poly-

nomial representing the N byte payload body is then evaluated at a set of points, with the value of the resulting polynomial stored in the template.

For illustrative purposes, assume that a biometric sample produces three features a_1, a_2, a_3 that must be protected. Let sa_1, sa_2, sa_3 be the stable components of these features, and let ra_1, ra_2, ra_3 be the residuals. For polynomial evaluation, the 24 bits of sa_1, sa_2, sa_3 are hashed into i, an 8 bit quantity that is stored in the template. The value i is then hashed, per transaction, a second time to define the point at which the polynomial is evaluated. To support multiple embedded data "columns," this second hash h is evaluated for different polynomials yielding values $rs_1, \ldots, rs_4$. Note the evaluation point/hash value h is not stored.

The result is an "encoded bipartite row" that contains the unprotected fields and 6 protected fields (the encoded stable field w used for matching, index i and 4 columns of evaluated polynomials). For data d that is less than 512 bits we spread the data over columns in 16 rows; above 512 bits, we spread equally over the columns taking as many rows as needed. We require at least 14 rows, and pad d if it does not require 4 columns to represent it. The location of w is randomized per row. The evaluated Reed-Solomon polynomials for the 4 key columns, $rs_1, \ldots, rs_4$, follow w using a circular mapping of the 6 slots. For example, if the random index was 3, then the sequence would be: $[rs_3, rs_4, w, rs_1, rs_2, i]$

When matching a probe, the system creates all the fields for each of its rows, including the "un-stored" hash value h for polynomial evaluation. A probe row potentially matches a gallery row if it finds a matching w among the encoded fields and the residuals (ra_1, ra_2, ra_3) are within some threshold. This test is necessary, but not sufficient, for a correct match. With w identified, the algorithm can then extract the evaluated polynomial values, $rs_1, \ldots, rs_4$. If w is incorrectly identified, if the row is an accidental match, or if the underlying hash value h is incorrect (because of a random collision in generating/matching w), some values labeled $rs_1, \ldots, rs_4$ will be extracted, but will be incorrect. We extract the k values for each of the j embedded data columns and obtain a set of hash evaluation points h_j and their Reed-Solomon polynomial evaluations $rs_{j,k}$ at the associated points.

Now comes one of the important implementation details, addressing both security and efficiency. One could effectively improve robustness by increasing the level of ECC, but doing so increases the ease with which an attacker can compromise d. Instead we use a two level hashing to improve robustness. Our two level hashing will, in general, map multiple sa_1, sa_2, sa_3 sets to the same index. Next, a procedure is followed to collect the multiple values during the mapping, check for consistency and use that consistency to help resolve any conflicts that arise when noisy data is mapped. The consistency check accounts for noise in the matching process, and the many-to-one mapping that permits non-unique mapping results. The result of the mapping and consistency check is a vector of length N polynomial values (some of which may be missing) that holds the values of the evaluated Reed-Solomon polynomial for each location. The vector of length N, with gaps marked, is used as input to the Reed-Solomon decode function, which allows us to recover d with up to g gaps and e errors, as long as $2g + e < E$, where E is the number of ECC bytes used. Each key column is recovered separately, with larger keys being the concatenation

of multiple columns. For added security, a checksum is computed over the 6 unprotected columns of the enrollment biotoken. The data d are XORed with a checksum before embedding, and again after decoding, which prevents any tampering with the biotoken.

When implemented, the above design for bipartite biotokens lays the foundation for the protocols of BKI. The primary benefit of BKI is the ability to store public biotokens that any user in a particular infrastructure can retrieve and use to generate a bipartite biotoken to send some secret back to the owner of the biotoken, with the assurance that the certificate containing the biotoken is valid (a validation process is described in Section 3.1). The security of such a scheme to publicly distribute biotokens derived from biometrics is of course a concern. It has been shown [38, 33] that revocable biotokens are cryptographically secure and guard against the secure template attacks of Scheirer and Boult [31]. Considering the doppelganger attack of Section 1, prior work [33] shows a test of over 500 *Million* impostor trials, with no false accepts - possibly the largest trial to date for this sort of test.

The amount of information leaked by the residual component of the biotoken, in an information theoretical sense, has yet to be analyzed. However, while unencoded, this information is still protected via the obfuscation scheme of folding the residual data back into the encrypted stable data, thus hiding its original position (described in Section 2 of [38]). Thus, an attacker would have to resolve the positional ambiguity of the residual data, before beginning to mount some sort of correlational attack with the collected residuals from multiple biotokens. The variational and very limited nature of the residual data (4 bytes per row component of the Bozorth fingerprint implementation [38]) makes their value as a unique identifier questionable. We also note that the residuals can be discarded, leading to a small reduction in accuracy when just the stable components of the features are matched, thus completely alleviating this concern. From these considerations, we have confidence that revocable biotokens can be used in a public setting.

3 A Biocryptographic Key Infrastructure

A Biocryptographic Key Infrastructure must incorporate elements from several different domains, including biometrics, cryptography and network security. Network entities (including clients and servers), enrollment procedures, validation procedures, data structures, authentication protocols and revocation protocols are all necessary for a fully functional infrastructure. Here we examine those details.

3.1 Composition, Enrollment and Validation

An overview of the Biocryptographic Key Infrastructure is shown in Figure 7. Several distinct entities are shown in the BKI graph. *Biometric Certificate Authori-*

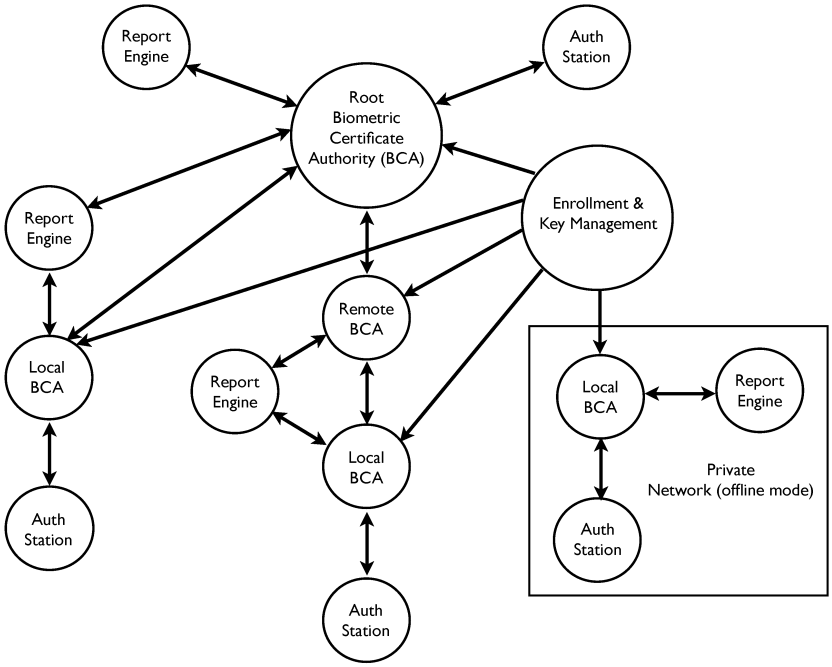

Fig. 7 Overview of Biocryptographic Key Infrastructure flow. The BKI can be viewed as a graph of interconnected nodes, each with a specific role. In a particular BKI, we find a root Biometric Certificate Authority (BCA), as we would have a root authority in PKI. The BCA trust path follows back centrally to the root, with individual local BCAs managing their own end-user enrollees. Offline BKI components (standalone computers or private networks) can also be supported.

ties (BCAs) are certificate authorities that support both public keys and revocable biotokens, and are biometrically verified by higher-level authorities, in a process described in detail below. As in PKI, a central root authority exists to authorize all BCAs below it. Enrollment and key management follows from each BCA up to the root. Auth Stations exist at the outermost regions of the graph, and are the places where users submit their biometric samples to generate enrollment biotokens or biotokens for a particular session. Report Engines can also be deployed throughout the BKI graph to propagate registration and transaction reports to other authorities.

In order to support the biotoken, we add some additional fields to the base x.509 v3 certificate via its extensions provision, similar to the approach of Martinez-Silva et al. [25]. This is shown in Figure 8. We can use certificates in both an online and offline setting, as is shown in Figure 7. If we are operating in an offline setting, such as a standalone computer or private network, we are not able to connect to BCAs on outside networks, including the root. In order to indicate the operating mode to the underlying BKI software, the certificate contains an "Online Only" flag and an "Offline Only" flag. For the user's biotoken, we first note the type of biotoken included. Recall from Figure 6 that a tree of different biotokens exists

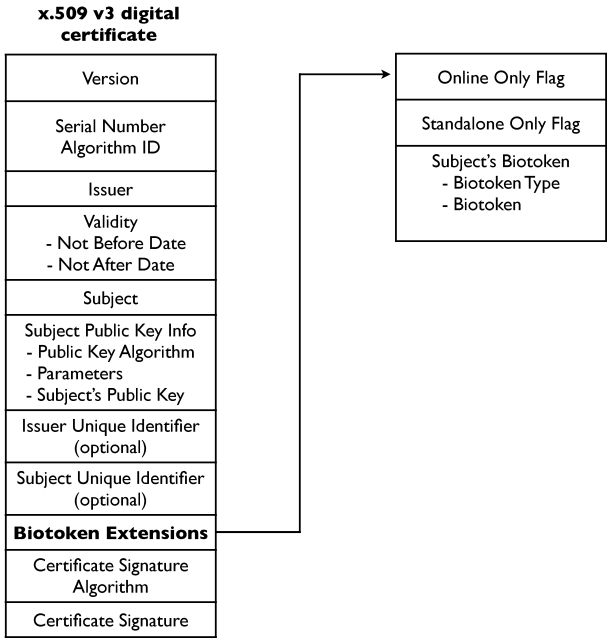

Fig. 8 Digital certificate supporting both public keys and biotokens.

for a particular user, with the possibility of a Root Biotoken, Master Biotoken, or Operational Biotoken being included in a certificate. Following the "Biotoken type" flag, the biotoken itself is included.

We need BCAs to trust each other, and we need to be able to place some trust in our end-users. To do this, we need an enrollment process where we require that someone biometrically register with the root BCA, which can search for this person in the existing records. To introduce an increased level of trust with biometrics, the standard Certificate Signing Request (CSR) [30] is augmented as per Figure 9. The CSR is the message sent to a CA by a user requesting a new certificate. The augmentation takes advantage of the open nature of registration information detail for new text fields, and the open extensions in the certificate template, as defined by [30].

Specifically for enrollment, BKI requires that a representative of an organization making a request generate an *enrollment biotoken*, which is passed up to the root authority for a *duplicate enrollment check* (which can tell us if this person been flagged as a malicious user, or if they are impersonating someone else). The enrollment biotoken is always generated as a Root Biotoken (Figure 6) using the root authority's public key, to enable matching across all enrollees (if keys differ between enrollments at this stage, it will not be possible to match any of them). The enrollment token is stored at the root BCA for use in all future enrollment checks. While this does not protect the privacy of the organizational representative at the database

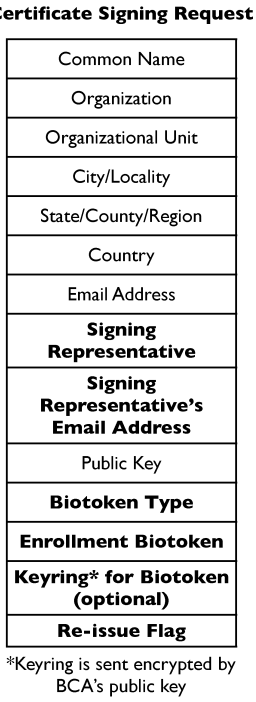

Fig. 9 A modification of the typical CSR message, including biotoken enrollment information.

level, it does maintain the integrity of the BCA establishment, and still protects the security of the representative's biometric data. This process is illustrated in Figure 10.

The same process follows for end users, except the enrollment token need not be passed up all the way back to the root from the user's Auth Station; more local BCAs can manage it. This is also illustrated in Figure 10. For both BCA and end user certificates, the validation process includes an analysis of the certificate with a BCA that is established as a VA. This is similar to the standard process for PKI, with a further step of biotoken validation to ensure a Man-in-the-Middle has not replaced the public biotoken in the certificate with his own. From a stored operational biotoken at the BCA, a local biotoken can be generated and matched against a bipartite biotoken generated from the public biotoken in question [32]. If the match is successful, then the certificate can be validated. This reduces the threat of the collision attack described in Section 1 in both the BCA and user scenarios, since an attacker would have to find a hash collision that validates the rogue certificate and compromise biometric data that will correctly match against the biotoken of the authorized representative of the BCA or end user.

3.2 Authentication Framework

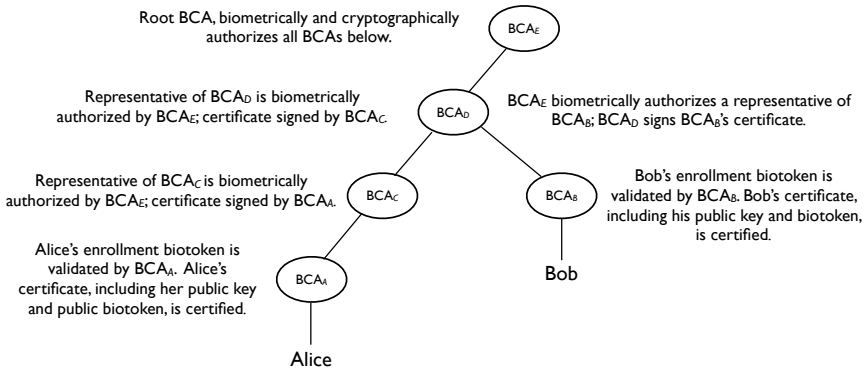

Fig. 10 The path from Alice, who wants to obtain Bob's certificate, to BCA_B, which certifies Bob's certificate, and ultimately Bob, who possess a certificate with his public key and biotoken.

For authentication, we must first understand how certificates are retrieved by parties wishing to communicate with some properly certified entity in the BKI structure. This procedure follows from PKI [34]. In Figure 10, an example of certificate retrieval is depicted, whereby a user Alice traverses an infrastructure composed of five different BCAs ($BCA_A, \ldots, BCA_E$) to retrieve another user Bob's certificate. Alice's certificate containing her public key and biotoken is certified by BCA_A; Bob's is certified by BCA_B. BAC_C has a certificate signed by BCA_A, so Alice can begin following the path through the graph to Bob. BCA_D has a certificate signed by BCA_C, and BCA_B has a certificate signed by BCA_D. By moving through the graph to BCA_D, and then down to Bob, Alice can validate Bob's certificate, and retrieve his public key and biotoken for use in some protocol/transaction. BCA_E is the root BCA, signing every BCA's certificate below it and biometrically authorizing all BCA representatives, and having its certificate signed by the same BCAs.

Extending the protocols defined in Section 24.9 of Schneier [34], we can support authentication with stronger non-repudiation. For the following three protocols, presume Alice has established a certification path to Bob, as described above, and Bob's certificate, containing his public key and biotoken. The numbering of the protocols is sequential, with each protocol after the first relying on the protocol(s) before it.

3.2.1 The one-way protocol:

1. Alice generates a random number, R_A.

2. Alice constructs a message, $M = (T_A, R_A, I_B, B_{BB}(d))$, where T_A is Alice's timestamp, I_B is Bob's identity, and d is a small piece of arbitrary data. d is embedded into a bipartite biotoken $B_{BB}(d)$ that is generated from Bob's biotoken.
3. Alice sends $(C_A, D_A(M))$ to Bob. (C_A is Alice's certificate; D_A is Alice's private key.)
4. Bob verifies C_A and obtains E_A. He makes sure these keys have not expired. (E_A is Alice's public key).
5. Bob uses E_A to decrypt $D_A(M)$. This verifies both Alice's signature and the integrity of the signed information.
6. Bob checks the I_B in M for accuracy.
7. Bob checks the T_A in M and confirms that the message is current.
8. Bob submits a biometric sample to a sensor; a local biotoken B_{BL} is then generated from the sample. B_{BL} is then matched against $B_{BB}(d)$, releasing d.
9. As an option, Bob can check R_A in M against a database of old random numbers to ensure the message is not an old one being replayed.

This protocol is an improvement over the "signature server" protocol previously introduced by Scheirer and Boult [32] for several reasons. Most obviously, it consists of a single message, as opposed to a 4-way transaction, and establishes the identities of both Alice and Bob and the integrity of any information sent from Alice to Bob, especially if d is a shared secret. This protocol also works by encrypting d with Bob's public key - but with the biometric version, Bob does not need to have his private key handy. Further security is provided if Alice has access to a private BCA that holds Bob's certificate, which would make Bob's biotoken a shared secret. Thus, a successful Man-in-the-Middle would need to know not only Alice's private key, but Bob's secret stored biotoken and secret d, as well.

3.2.2 The two-way protocol:

10. Bob generates another random number, R_B.
11. Bob constructs a message $M' = (T_B, R_B, I_A, B_{AB}(d))$, where T_B is Bob's timestamp, I_A is the identity of Alice, and d is the same data as in step 2. d is embedded into a bipartite biotoken $B_{AB}(d)$ that is generated from Alice's biotoken, obtained from C_A.
12. Bob sends $D_B(M')$ to Alice.
13. Alice uses E_B to decrypt $D_B(M')$. This verifies both Bob's signature and the integrity of the signed information.
14. Alice checks the I_A in M' for accuracy.
15. Alice checks the T_B in M' and confirms that the message is current.
16. Alice submits a biometric sample to a sensor; a local biotoken B_{AL} is then generated from the sample. B_{AL} is then matched against $B_{AB}(d)$, releasing d. If this d matches the d sent in the first transmission, Alice can be assured that Bob's biometric was used to unlock $B_{BB}(d)$.
17. As an option, Alice can check R_B in M' to ensure the message is not an old one being replayed.

Now Alice has further assurance Bob is actually Bob, and not an impostor. But Bob still has no assurance of Alice's identity beyond her certificate. This can be solved by a three-way protocol, where in addition to the original d, Bob also sends a d' in the same token (step 16). Alice can verify d (step 17), and send d' back to Bob for validation.

3.2.3 The three-way protocol:

18. Alice takes the recovered d' from step 16, and sends $D_A(d')$ back to Bob.
19. Bob uses E_A to decrypt $D_A(d')$, unlocking d'. Bob can be assured that Alice's biometric was used to unlock $B_{AB}(d)$ in step 17.

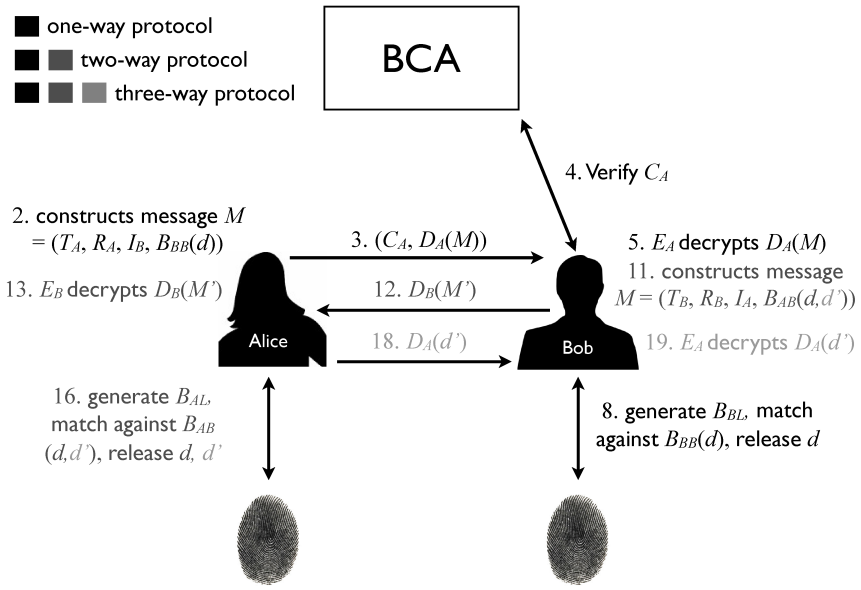

Fig. 11 The data-transfer steps for the one-way, two-way, and three-way protocols described in Section 3.2. It is assumed that Alice has Bob's certificate C_B at the beginning of the one-way protocol.

3.3 Revocation and Reissue

Unlike standard PKI, we cannot just revoke a certificate, generate a new random key and re-issue - we must address the biometric re-issue as well. While many works

in the research literature describe revocation as a property of a particular template protection scheme, only one work [2] has gone as far as describing the revocation procedure, albeit on a per template basis. Below we detail three different scenarios for BKI protocol driven revocation and re-issue. When we describe compromise here, we mean a compromise of the biotoken itself, and not the original biometric features.

3.3.1 Scenario 1: Manual Re-issue

The BCA that issued the certificate must maintain a certificate revocation list (CRL). This list only contains revoked certificates, and not expired certificates. If the user's key has been compromised, or the user's biotoken has been compromised, or the BCA's key has been compromised, or because the BCA no longer wants to certify the user, the user's certificate can be revoked. In this scenario, it is presumed that the BCA has not retained any transformation information necessary to invert the biotoken it stores.

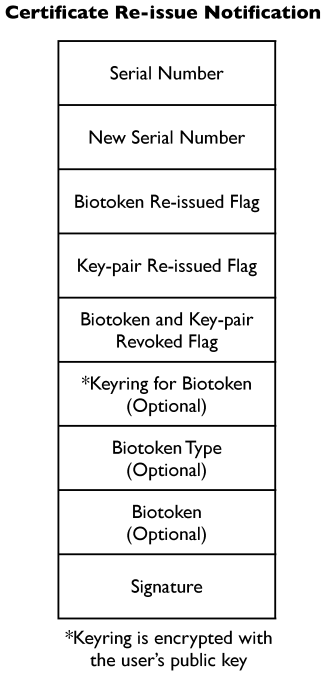

Fig. 12 The newly defined CRN message for certificate revocation and re-issue.

To begin the revocation process with re-enrollment, the BCA places the certificate in question on its CRL, and notifies the owner with a Certificate Re-issue Notification (Figure 12) (CRN) via the contact information provided in the CSR. This CRN is a new notice introduced in this work. If the owner is allowed to re-issue, a new public-private key pair and a new biotoken are generated at the Auth Station. This information is sent back to the BCA in the form of a new CSR. If this CSR is accepted, a new certificate is issued.

In an alternate, yet valid, scenario for manual re-issue, re-enrollment is not required. If the user's biotoken, or biotoken and key pair, has been compromised, and the BCA possesses a stored uncompromised base biotoken that was used to generate the compromised biotoken, the owner can re-issue their certificate by varying the transformations used for encoding on their end, while not needing to submit another biometric sample. To begin this revocation process, the BCA places the certificate in question on its CRL, and notifies the owner with a CRN via the contact information provided in the CSR. This CRN contains the owner's base biotoken. The owner will generate new keys for biotoken re-encoding, and use them to generate a new biotoken. This new biotoken, and optionally a new public key, is sent back to the BCA in a new CSR.

While two scenarios for automatic re-issue are discussed below, if a public key and biotoken are compromised for a particular certificate, then manual re-issue with re-enrollment will always be forced. Manual re-issue with re-enrollment is also forced if the BCA's key has been compromised, where trust can no longer be placed in the existing data stored at the BCA.

3.3.2 Scenario 2: Automatic Re-issue of Biotoken

In cases where the BCA detects a compromise (especially in its own infrastructure) of a stored biotoken, it is very desirable to revoke and re-issue certificates in some automated fashion. To support this, the BCA must possess the necessary keys to invert the token, and subsequently generate a new token based on stored information. This stored information *need not be* the original biometric features. Referring back to the biotoken issue/re-issue tree of Figure 6, any level of token can be generated by an Auth Station, and transmitted on to the BCA. Thus, if the biotoken exists at the 2nd - *n*th level of encoding, any BCA (except possibly the root, as described in Section 2) performing the inversion will not be able to recover the original biometric features.

The initial enrollment process is modified in this scenario to transmit the transformation key information used to create the enrollment biotoken to the BCA. The CSR contains an optional field (shown in Figure 9) to include a keyring with all of the necessary keys / passwords / identifiers used to encrypt the stable (that is, some encoding $w_{j,n}(w_{j,n-1}, T_n)$, where $n > 1$, if the original biometric features are to be protected) portion of the biotoken, during the transform. The requesting entity will include this keyring, encrypted by the BCA's public key, in its CSR. The BCA

will store this encrypted keyring for later use if revocation and re-issue becomes necessary.

If the user's biotoken has been compromised, the user's certificate can be revoked and re-issued automatically. To begin the revocation process, the BCA places the certificate in question on its CRL, and notifies the owner via the contact information provided in the CSR. If the owner is allowed to re-issue, the BCA will take it upon itself to invert the biotoken back a level (to $w_{j,n-1}$, where $n > 1$), generate a new set of transformation key information, and re-encode the biotoken (producing $w'_{j,n}$). A new certificate is then created with the new biotoken, and the original public key. The BCA then sends the owner of the certificate a CRN, which indicates the serial number of the revoked certificate, the serial number of the re-issued certificate, and the new keyring for the new biotoken (encrypted with the user's public key). This message is signed by the BCA.

Automatic re-issue may happen transparently to the user, with the underlying BKI software taking note of the CRN, and updating the transformation key information for biotoken generation at the user's Auth Station.

3.3.3 Scenario 3: Automatic Re-issue of key-pair

Similar to Scenario 2, it is very desirable to revoke and re-issue certificates in some automated fashion when the public/private key-pair becomes compromised. To support this, the BCA can use a bipartite biotoken generated from the uncompromised biotoken stored in the user's certificate to convey a secret back to the user.

If the user's key-pair has been compromised, the user's certificate can be revoked and reissued automatically. To begin the revocation process, the BCA places the certificate in question on its CRL, and notifies the owner via the contact information provided in the CSR. If the owner is allowed to re-issue, the BCA will take it upon itself to generate a new key-pair. A new certificate is then created with the new public key, and the original biotoken. The BCA then embeds the new private key into a bipartite biotoken generated from the user's biotoken. The BCA then sends the owner of the certificate a Certificate Re-issue Notification (CRN), which indicates the serial number of the revoked certificate, the serial number of the re-issued certificate, and the bipartite biotoken containing the embedded private key. This message is signed by the BCA.

The automatic re-issue process will require some intervention by the user here. Namely, the user must submit his/her biometric at the Auth Station to release their new private key from the bipartite biotoken in the CRN.

4 Applications

Now that we've seen the underlying infrastructure and protocols, we can begin to think about the utility of BKI for different applications. Internet tools are of primary

interest, because they are at the front-line of the security battleground. Enforcing server validation is a must, if we want to defeat Phishing and Man-in-the-Middle attacks. In Section 3.2.1 we introduced a one-way protocol that is suitable for server validation, and forces the user to take action by presenting a biometric sample when receiving any certificate. For server validation, d can be a "welcome message" that Bob enters during enrollment. The biometric component of this scheme forces Bob to validate the integrity of the server, even if the certificate check has occurred, and has been ignored. If Bob can unlock $B_{BB}(d)$ and get his "welcome message" back, the server is indeed valid. If Bob's biotoken is a shared secret, he has further confidence the server is legitimate. This protocol can be integrated into common Internet tools, such as web browsers, email clients, and instant messaging clients that already support PKI. The only difference to the user is that they are required to submit a biometric sample upon receiving a certificate from a server.

In terms of network services, BKI enabled services can allow for robust authentication, giving the user more confidence in the server, and the server more confidence that it is dealing with a legitimate user. The work of [32] suggested the use of bipartite biotokens with Kerberos [26], but in a standalone configuration without certificates certifying biotokens. One can also envision an S/Key-like [17] one-time password scheme using bipartite biotokens. In this scheme, once receiving the request, the authentication server generates a one-time password, and creates a bipartite biotoken containing this password. If the client matches the bipartite biotoken sent from the authentication server, it will release the password, and complete the authentication. In order to solve the biotoken distribution problem for network authentication, PKI-enabled LDAP [6] can be used in the same manner for BKI applications. Thus, a wide variety of authentication schemes can take advantage of a common certificate repository, including user records, keys, and biotokens.

Digital documents represent another important application area for BKI. Many sensitive documents, including medical records, financial records, and government records are protected using PKI and digital signatures, but we cannot tell who exactly is accessing these documents beyond knowing that a particular key unlocks or verifies them. Using biparite biotokens, the key used to encrypt a document that belongs to Bob can be embedded into Bob's bipartite biotoken. Thus, only Bob can release the key, and access the document. For digital signatures, a signature server protocol [32] can add a biometric authorization component to the standard signature process. Again, with BKI providing the certificate distribution mechanism, a full security solution for document management is realized.

In all of these applications, usability is, of course, a legitimate concern. By adding a second physical factor, we also add more work for the user, and a small cost for the additional sensor hardware. However, not much more work is required to submit a biometric sample - it can be as simple as placing a finger down on a sensor for just a few seconds. Thanks to the recent prevalence of biometric systems, many corporate, government, and even home users are already used to this. Many laptops are already equipped with inexpensive fingerprint sensors, and many PC vendors offer low-cost fingerprint enabled mice. A good compromise is the judicious use of the biometric component; if the user is very concerned about the security of their

financial activities, they may choose to only use BKI for particular sites related to financial services, and take their chances with more conventional PKI provisions for everything else.

5 Conclusions

In this chapter, we have taken a look at security issues with both PKI and biometrics, and introduced a Biocryptographic Key Infrastructure incorporating a secure template technology that solves problems with both. In summary, PKI suffers from problems related to the trust that is presumed for all entities in the infrastructure. By incorporating a secure biometric template technology such as revocable biotokens into digital certificate signing requests, we can achieve improved non-repudiation, and thus increase the trust placed in both certificate authorities and users, while addressing the biometric dilemma and biometric doppelganger attack. Moreover, with a second factor that allows the secure transfer of embedded data, we can support automatic certificate revocation and re-issue. Ultimately, the goal here is to prevent common Phishing and Man-in-the-Middle attacks, which can be accomplished using the protocols we have defined for secure authentication between two parties using keys and biotokens. With the base protocols, we can go on to enhance common applications such as LDAP, Internet tools (browsers, email clients, IM clients), and digital document signing.

Proposed standards for PKI including biometrics have been constrained to the direct application of traditional biometric templates into certificates. Secure template technologies, including revocable biotokens, have matured to the point of being useful for systems integration. To date, no formal document exists outlining requirements or specifications for secure template technology, let alone a combination of secure templates and PKI. This hampers the widespread adoption of a good two-factor solution to the shortcomings of PKI. Thus, we propose moving this emerging paradigm out of the realm of pure research and into the hands of a standards body, such as IETF, for serious consideration. It is our hope that the sketch of BKI presented here will provide a solid foundation to the first round of a standards process.

Acknowledgements This work was supported in part by NSF STTR Award Number 0750485 and NSF PFI Award Number 065025.

References

1. Adams, C., Farrell, S.: Internet X.509 Public Key Infrastructure Certificate Management Protocols. RFC 2510 (Proposed Standard) (1999). http://www.ietf.org/rfc/rfc2510.txt. Accessed 18 May 2011

2. Arndt, C.: Biometric Template Revocation. In: A. Jain, N. Ratha (eds.) Biometric Technology for Human Identification. Proceedings of the SPIE, vol. 5404, pp. 164–175. SPIE (2004)
3. Benavente, O.: Authentication Services and Biometrics: Network Security Issues. In: Proc. of the 39th Annual International Carnahan Conference on Security Technology (CCST 2005), pp. 333–336 (2005)
4. BioAPI: Business Objectives and Values. BioAPI Consortium (2010). http://www.bioapi.org/objectives.asp. Accessed 18 May 2011
5. BioLab: FVC 2006: Fingerprint Verification Competition. University of Bologna (2006). http://bias.csr.unibo.it/fvc2006/databases.asp. Accessed 18 May 2011
6. Boeyen, S., Hallam-Baker, P.: Internet X.509 Public Key Infrastructure Repository Locator Service. RFC 4386 (Proposed Standard) (2006). http://www.ietf.org/rfc/rfc4386.txt. Accessed 18 May 2011
7. Boyen, X., Dodis, Y., Katz, J., Ostrovsky, R., Smith, A.: Secure Remote Authentication Using Biometric Data. In: Proc. of EUROCRYPT, pp. 147–163 (2005)
8. Cappelli, R., Lumini, A., Maio, D., Maltoni, D.: Fingerprint Image Reconstruction from Standard Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(9), 1489–1503 (2007)
9. Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S.: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. RFC 3647 (Proposed Standard) (2003). http://www.ietf.org/rfc/rfc3647.txt. Accessed 18 May 2011
10. Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., Polk, W.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard) (2008). http://www.ietf.org/rfc/rfc5280.txt. Accessed 18 May 2011
11. Dawson, E., Lopez, J., Montenegro, J., Okamoto, E.: BAAI: Biometric Authentication and Authorization Infrastructure. In: Proc. of the International Conference on Information Technology: Research and Education (ITRE 2003), pp. 274–278 (2003)
12. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy Extractors. In: P. Tuyls, B. Skoric, T. Kevenaar (eds.) Security with Noisy Data: Private Biometrics, Secure Key Storage and Anti-Counterfeiting, chap. 5, pp. 79–99. Springer-Verlag (2007)
13. East-Shore: Fingerprint Image Database. East Shore Technologies (2007). http://www.east-shore.com/data.html. Accessed 18 May 2011
14. Ellison, C., Schneier, B.: Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. Computer Security Journal **16**(1), 1–7 (2000)
15. Gerck, E.: Overview of Certification Systems: X.509, PKIX, CA, PGP and SKIP. The Bell **1**(3), 8 (2000)
16. Gutmann, P.: PKI: It's not Dead, Just Resting. IEEE Computer **35**(8), 41–49 (2002)
17. Haller, N.: The S/KEY One-Yime Password System. RFC 1760 (Proposed Standard) (1995). http://www.ietf.org/rfc/rfc1760.txt. Accessed 18 May 2011
18. Jain, A., Nandakumar, K., Nagar, A.: Biometric Template Security. EURASIP Journal on Advances in Signal Processing (2008)
19. Juels, A., Sudan, M.: A Fuzzy Vault Scheme. In: Proc. of the IEEE International Symposium on Information Theory, p. 408 (2002)
20. Juels, A., Wattenberg, M.: A Fuzzy Commitment Scheme. In: Proc. of the 6th ACM Conference on Computer and Communications Security, pp. 28–36 (1999)
21. Kanade, S., Petrovska-Delacrétaz, D., Dorizzi, B.: Generating and Sharing Biometrics Based Session Keys for Secure Cryptographic Applications. In: Proc. of the IEEE Fourth International Conference on Biometrics: Theory, Applications, and Systems (BTAS 2010) (2010)
22. Krause, M.: The Expanding Surveillance State: Why Colorado Should Scrap the Plan to Map Every Driver's Face and Should Ban Facial Recognition in Public. The Independence Institute (2001). http://www.i2i.org/articles/8-2001.PDF. Accessed 18 May 2011
23. Kuhn, D., Hu, V., Polk, W., Chang, S.: Introduction to Public Key Technology and the Federal PKI Infrastructure. In: National Institute of Standards and Technology, SP 800-32 (2001)
24. Kwon, T., Moon, H.: Multi-modal Biometrics with PKI Technologies for Border Control Applications. In: Intelligence and Security Informations (Springer Lecture Notes in Computer Science), vol. 3495, pp. 99–114 (2005)

25. Martinez-Silva, G., Henriquez, F., Cortes, N., Ertaul, L.: On the Generation of X.509v3 Certificates with Biometric Information. In: Proc. of the 2007 International Conference on Security and Management (SAM '07) (2007)
26. Neuman, C., Yu, T., Hartman, S., Raeburn, K.: The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard) (2005). http://www.ietf.org/rfc/rfc4120.txt. Accessed 18 May 2011
27. NIST: National Institute of Standards and Technology NIST Special Database 29. Standard Reference Data (2011). http://www.nist.gov/srd/nistsd29.cfm. Accessed 18 May 2011
28. Ratha, N., Chikkerur, S., Connell, J., Bolle, R.: Generating Cancelable Fingerprint Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(4), 561–572 (2007)
29. Reinert, L., Luther, S.: User Authentication Techniques Using Using Public Key Certificates, National Security Agency, Central Security Service (1997)
30. Schaad, J.: Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF). RFC 4211 (Proposed Standard) (2005). http://www.ietf.org/rfc/rfc4211.txt. Accessed 18 May 2011
31. Scheirer, W., Boult, T.: Cracking Fuzzy Vaults and Biometric Encryption. In: Proc. of the 2007 Biometrics Symposium, held in conjunction with the Biometrics Consortium Conference (BCC 2007), Baltimore, MD. (2007)
32. Scheirer, W., Boult, T.: Bio-cryptographic Protocols With Bipartite Biotokens. In: Proc. of the IEEE 2008 Biometrics Symposium, held in conjunction with the Biometrics Consortium Conference (2008)
33. Scheirer, W., Boult, T.: Bipartite Biotokens: Definition, Implementation, and Analysis. In: Proc. of the IEEE/IAPR International Conference on Biometrics, pp. 775–785 (2009)
34. Schneier, B.: Applied Cryptography, Second Edition. John Wiley & Sons, Inc. (1996)
35. Sotirov, A., Stevens, M., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D.A., de Weger, B.: MD5 Considered Harmful Today. HashClash Project (2008). http://www.win.tue.nl/hashclash/rogue-ca/. Accessed 18 May 2011
36. Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D.A., de Weger, B.: Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA certificate. In: Proc. of the International Cryptology Conference on Advances in Cryptology (2009)
37. Sutcu, Y., Sencar, T., Memon, N.: A Secure Biometric Authentication Scheme Based on Robust Hashing. In: Proc. of the 7th ACM Workshop on Multimedia and Security (MM-SEC 2005) (2005)
38. T. Boult, W.S., Woodworth, R.: Secure Revocable Finger Biotokens. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007) (2007)
39. Tang, Q., Bringer, J., Chabanne, H., Pointcheval, D.: A Formal Study of the Privacy Concerns in Biometric-Based Remote Authentication Schemes. In: Proc. of the Information Security Practice and Experience Conference (2008)
40. The Open Group: Architecture for Public-key Infrastructure (APKI) (1999)
41. Ueshige, Y., Sakurai, K.: A Proposal of One-Time Biometric Authentication. In: H. Arabnia, S. Aissi (eds.) Proc. of the International Conference on Security and Management (SAM 2006) (2006)