U-Prove Technology Overview V1.1

Revision 2

Microsoft Corporation

Author: Christian Paquin

April 2013

Summary

The U-Prove technology can be used to reconcile seemingly conflicting security and privacy requirements in electronic communication and transaction systems. This overview explains the foundational features of the U-Prove technology as specified in the U-Prove Cryptographic Specification V1.1.

Contents

List of Figures

1 Introduction

Organizations are increasingly looking to securely identify individuals who access their services, both on the Internet and offline. More and more they also seek to learn other identity-related information about individuals that is held by other organizations. These authentication and data sharing imperatives are driven by cost and efficiency considerations, by new business models that leverage personal information, and by the explosive rise of phishing, identity theft, and other security threats.

Conventional mechanisms for user authentication and data sharing, such as plastic cards and paper certificates, are costly, vulnerable to counterfeiting, and problematic for online use. As a result there is a rapidly growing interest in mechanisms that can be implemented in software or hardware, can run over electronic networks, and can be relied on by many organizations. SAML, WS-Federation, OpenID, and OAuth protocols, PKI, eID cards and (other) approaches to single sign-on and federated identity are examples of increasingly popular mechanisms to achieve these objectives. The demand for such mechanisms is particularly urgent in enterprise identity and access management, critical information infrastructure protection, government online service delivery, e-commerce, electronic health record management, and social networking.

The transition to digital mechanisms for secure authentication and verifiable data sharing is also beneficial to individuals. It is, however, not without peril to the personal security, privacy, autonomy, and civil liberties of individuals. As more and more identity-related information is shared with and between organizations, individuals lose all control over the extent to which organizations can monitor and profile their actions, impersonate them, and prevent them from transacting autonomously. The threats originate not just from malicious personnel and other insiders but also from hackers and computer malware that manage to gain insider status. The problem is exacerbated by the incredible ease with which digital information can be collated, shared, and leaked.

Similarly, parties relying on authentication and other identity-related statements may have concerns for reasons of their own (whether competitive, security-related, or other). It is one thing to trust an organization with being an authoritative source for certain identity-related information; it is an entirely different thing to have to also trust that same organization with being highly available, not impersonating the relying party's clients, and not spying on those clients (i.e., who is accessing what service at which relying party at what particular time). Relying parties may also be concerned with the ability of issuing organizations (and hackers and malware) to deny individuals access to their services. These risks become more severe when individuals are tethered to issuing organizations in the sense that they must retrieve a new statement whenever they visit a relying party.

Contrary to popular belief, privacy and autonomy interests in user authentication and data sharing systems are not diametrically opposed to security interests. This is where the U-Prove technology comes in.

2 About the U-Prove technology

U-Prove is an innovative cryptographic technology that enables the issuance and presentation of cryptographically protected statements in a manner that provides what is known as “multi-party security:” issuing organizations, users, and relying parties can protect themselves not just against outsider attacks but also against attacks originating from each other. At the same time, the U-Prove technology enables any desired degree of privacy (including authenticated anonymity and pseudonymity) without contravening multi-party security. These user-centric aspects make the U-Prove technology ideally suited to create the digital equivalent of paper-based credentials and the plastic cards in one's wallet.

In the context of identity and access management, the U-Prove technology enables organizations to exchange identity-related information in cryptographically protected form via the individuals to whom it pertains or via other intermediating parties (such as agents, brokers, and outsourcing suppliers). Intermediating parties can see the protected information that is shared, can store it upon issuance (whether for offline use or reuse), and can selectively disclose only those aspects required for a transaction. Organizations cannot learn anything beyond the disclosed aspects, even if they collude and have unlimited resources to analyze disclosed protocol data.

More generally, the U-Prove technology can be used to reconcile seemingly conflicting multi-party security and privacy requirements in all sorts of electronic communication and transaction systems. Examples include digital rights management, electronic voting, electronic payment instruments, electronic health records, electronic postage, online auctions, public transport ticketing, road-toll pricing, loyalty schemes, and e-gaming. The U-Prove technology can also be applied to protect identity-related information pertaining to non-human entities, such as computer processes, software applications, hardware devices, and so forth. Furthermore, since entities can securely share information via any untrusted party while delegating partial control over its release to that party, the U-Prove technology enables the design of new applications with no physical-world analogy; one example area of interest is cloud computing services that can perform limited operations on integrity-protected input data from different sources.

The U-Prove Cryptographic Specification V1.1 [UPCS] specifies the cryptographic protocols for the foundational U-Prove features; these features are described in this overview. The full U-Prove technology provides additional features not specified in the current version of the specification; see Appendix 1.

For the cryptographic details of how the U-Prove properties are achieved, see [Brands].

3 U-Prove technology basics

At the heart of the U-Prove technology is the notion of a *U-Prove token*. A U-Prove token is a cryptographically protected collection of information of any kind (referred to as *attributes*). It is issued by an authoritative source to a user via an issuance protocol, and subsequently presented by the user to a relying party via a presentation protocol. See Figure 1. Because a U-Prove token is just a binary string it can be issued and presented over any electronic network. To perform the U-Prove protocols, all participants require computing devices that function on their behalf.

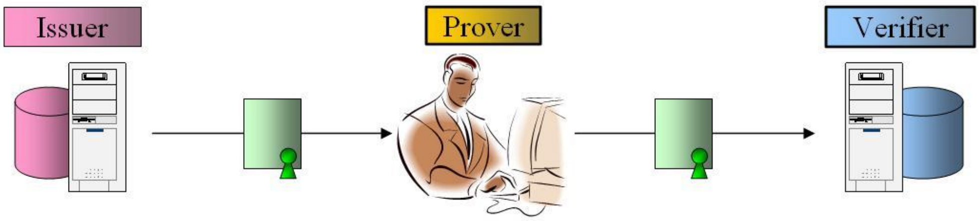


Figure 1: U-Prove issuance and presentation protocols

In the rest of this overview the authoritative source is referred to as the *Issuer*, the user is referred to as the *Prover* (for reasons that will become clear), and the relying party is called the *Verifier*. Issuers, Provers, and Verifiers are basic roles. Throughout this overview we refer to them either as computing devices that interact via the U-Prove token protocols, as entities represented by these devices, or as a mix of both; the appropriate interpretation will be clear from the context. In practice, multiple roles may be performed by the same entity or a role may be split across several entities.

This section describes the basic structure and protections of a U-Prove token; see Appendix 2 for a schematic representation of the complete token structure (including fields that will be explained in the next section).

3.1 Issuing a U-Prove token

In order for a Prover to retrieve a U-Prove token from an Issuer, the two parties must engage in an instance of the *U-Prove issuance protocol*. This is a cryptographic protocol that takes as its inputs, among others, any attributes to be encoded into the token. The innovative features of the U-Prove technology derive from the cryptographic design of the issuance protocol, which is based on advances in modern cryptography. For the purposes of this overview it suffices to know that the Issuer's signature is not a conventional RSA or DSA signature, and that issuance is a three-leg interactive protocol enabling the Prover to hide certain token elements from the Issuer.

Prior to issuing a U-Prove token to a Prover, its Issuer may assess the Prover's eligibility to receive a token. In addition, the Issuer may want to ascertain that the statements it will encode into the token pertain to the right Prover. These objectives may require the Issuer to authenticate Provers within its own domain or to verify other statements; nothing prevents the Issuer from relying on U-Prove tokens issued by other Issuers. All of these considerations are outside the scope of the U-Prove technology.

The U-Prove issuance protocol enables the Issuer to protect U-Prove tokens against unauthorized manipulations, in accordance with application-specific requirements. The following basic protections are always in place:

- Integrity and source authenticity: Each issued U-Prove token contains an unforgeable digital signature of its Issuer on the entire contents, created by the Issuer by applying its private key. *The Issuer's signature* serves as its authenticity mark on the U-Prove token; it enables anyone to verify that the U-Prove token was issued by the Issuer and that its contents have not been altered.

- Replay attack prevention: Each issued U-Prove token also contains a token-specific public key that is known only to the Prover. The Prover randomly generates it during the issuance protocol, together with a corresponding private key for the U-Prove token. In contrast to the token's public key, this private key is not part of the U-Prove token; the Prover never discloses it when using the U-Prove token. In the next section we will explain how this prevents Verifiers from replaying presented U-Prove tokens.

Beyond these basic protections, U-Prove tokens can be protected using various optional security measures that will be described in Section 5.

The Issuer may issue arbitrarily many U-Prove tokens by using the same private key, without any loss of security. The Issuer's corresponding public key is part of the *Issuer parameters*; this public information should be available to anyone interested in verifying issued and presented U-Prove tokens. Conceptually, the Issuer parameters are equivalent to the certificate of a Certificate Authority in a PKI.

3.2 Presenting a U-Prove token

To present a U-Prove token to a Verifier, the Prover and the Verifier engage in an instance of the *U-Prove presentation protocol*. The Prover provides the token attributes (or, as we will see in Section 4.3, only a subset of the attributes), the Issuer's signature, and the token-specific public key. The Prover also sends a response to the Verifier. To compute this response the Prover applies the private key for the U-Prove token to a *presentation challenge* of the Verifier.[1] This presentation challenge must include a nonce, that is, a unique number that is never reused; a large random number will do, as will a timestamp or a counter appended to a unique Verifier identifier.[2]

We refer to the Prover-computed response as the *presentation proof*; it is a cryptographic proof of possession of the private key corresponding to the presented U-Prove token. It proves that the private key has been applied to the presentation challenge but the private key itself remains secret; this security guarantee holds even if all Verifiers and the Issuer collude, examine arbitrarily many presentation proofs created with the same U-Prove token, and deviate from the issuance and the presentation protocols. As a result, Verifiers cannot replay the U-Prove tokens presented to them.[3]

The verification of a presented U-Prove token does not require any secret information, special-purpose hardware, or real-time communication with the Issuer or any other party; all that is needed is an authentic copy of the Issuer parameters under which the U-Prove token was issued.

Upon verification of a presented U-Prove token, the rest of the session between the Prover and the Verifier may be securely tied to the initial authentication event; how this is accomplished is outside the scope of the U-Prove technology.

[1] In the U-Prove Cryptographic Specification V1.1 [UPCS11], the presentation challenge must be encoded into the presentation message. See Section 3.6.

[2] Verifiers could allow Provers to pick their own nonce, assuming they can verify the uniqueness of proposed nonces. If the nonce is a random number concatenated to an indication of the current day and a Verifier identifier, say, Verifiers need to locally store used nonces only for the duration of a day. In some cases (e.g., for on-demand U-Prove tokens discussed in Section 3.4) nonce values should also be unpredictable to prevent pre-computation of responses.

[3] An alternative is for the Issuer to encode a Verifier designation into the token. This, however, may be undesirable for reasons of privacy, availability, or usage flexibility; also, it does not prevent replay by a Verifier to itself.

By reusing the same U-Prove token in subsequent visits to the same Verifier, the Prover can establish a relationship with the Verifier that spans multiple sessions. By using different U-Prove tokens on different occasions, on the other hand, the Prover's activities can be segmented in a manner that protects the Prover's privacy (even in the face of collusions involving the Issuer) and minimizes the risk of identity theft; see Section 4.2.

A Prover can create arbitrarily many presentation proofs with the same U-Prove token, within and across sessions with Verifiers, unless reuse limitations prevent such. This enables Provers to securely re-authenticate to Verifiers using the same U-Prove token; see Section 3.5 for more information.

While U-Prove tokens do not require encryption over the wire to prevent replay attacks, encryption may nonetheless be desirable for message confidentiality. Depending on application requirements, encryption may be applied to a token in its entirety or to selected token contents. Encryption is outside the scope of the U-Prove technology.

3.3 Token information field

To make a U-Prove token more informative, during the issuance protocol the Issuer can encode application-specific attributes into its *token information field*. Information encoded into this field is invariably disclosed when using the U-Prove token; without it, the Issuer's signature cannot be verified. See Figure 2. The token information field is primarily intended for expiry dates, token usage restrictions, and token metadata.

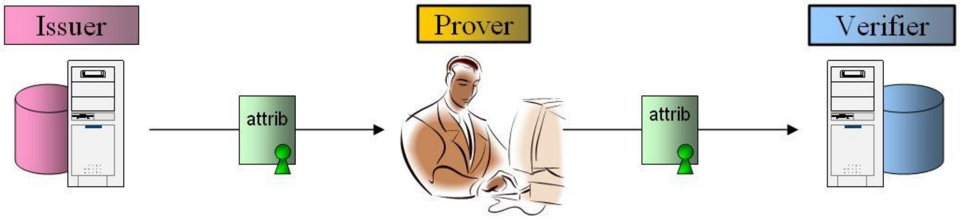


Figure 2: Token information field

As we will see in Sections 4.1 and 4.2, presented U-Prove tokens cannot be traced to their issuance or linked to other U-Prove tokens of the same Prover beyond the extent enabled by disclosed application-specific attributes.[4]

Any attributes that the Issuer encodes into a U-Prove token must be supplied by or agreed upon by the Prover; this enables Provers to boycott inappropriate choices.[5] In practice, the inspection and approval of Issuer-provided attributes may be handled exclusively by software running on the Prover's device or may be implemented in a manner that requires human involvement; this is up to application requirements.

3.4 On-demand and long-lived U-Prove tokens

U-Prove tokens may be either on-demand or long-lived:

- On-demand tokens must be obtained from their Issuer while interacting with a Verifier and can be used only with that particular Verifier for the duration of the session. The Verifier can accomplish this by

[4] As we will see in Sections 4.3 and 4.4, respectively, two additional types of token field are available that enable Provers to hide encoded attributes from Verifiers and from the Issuer.

[5] That is not to say that Issuers cannot encode hashed or encrypted data into U-Prove tokens, but Provers must cooperate in including the resulting values.

specifying an unpredictable nonce in the presentation challenge and expecting the Prover to present a token that encodes this nonce. See Section 4.4 for more information.

- Long-lived U-Prove tokens are stored (in memory or in persistent storage) by their Provers upon issuance and can be used arbitrarily many times until they expire or are revoked (see Section 5.2), unless reuse limitation measures are taken at issuance or presentation time.

On-demand U-Prove tokens have several benefits over long-lived U-Prove tokens: Issuers can revoke them on the basis of their Provers' identities (see Section 5.2), Issuers can encode Prover-related assertions into U-Prove tokens in direct response to each Verifier's request, there is no need to encode expiry dates (eliminating the need for Verifiers to securely maintain a synchronized clock), and Verifiers can be assured of the freshness of any attributes encoded in presented U-Prove tokens.

On the downside, on-demand U-Prove tokens cannot be reused and require a real-time connection between Provers and the Issuer at presentation time. This may be costly, infeasible, or burdensome for Provers or the Issuer, and introduces availability and denial-of-service risks. It may also interfere with the autonomy interests of Provers and Verifiers. Namely, the Issuer can (on a per-case basis) provide wrong attributes, withhold U-Prove token issuance, and demand Verifier-related information prior to issuing a U-Prove token. The Issuer also learns the starting time and date of each session between a Prover and a Verifier. Long-lived U-Prove tokens, in contrast, may be retrieved in batch at any convenient moment independent of when they are presented.

Long-lived U-Prove tokens are also preferable over on-demand U-Prove tokens for Provers (and Verifiers) who want to protect their privacy vis-à-vis the Issuer and other parties; see Sections 4.1 and 4.2. More generally, in contrast to on-demand tokens, long-lived U-Prove tokens enable Provers and Verifiers to interact in a truly autonomous (untethered) fashion.

Whether or not long-lived U-Prove tokens are favorable over on-demand U-Prove tokens is entirely application-dependent, possibly depending on per-interaction needs of Provers and Verifiers. An Issuer may issue both on-demand and long-lived U-Prove tokens on the basis of the same Issuer parameters and private key.

The term "long-lived" does not imply a long lifetime: nothing prevents an Issuer from issuing a long-lived U-Prove token with an expiry date that is within a short time of its issuance.

3.5 Token identifier

At the application level, presented U-Prove tokens may be hooked up to Prover accounts, enabling Verifiers to recognize repeat visitors.[6] To facilitate account indexing, a unique *token identifier* can be computed from each U-Prove token. Since the token identifier is computed as a cryptographically secure hash of the U-Prove token's public key and the Issuer's signature, Provers and Verifiers can be assured of the universal uniqueness of token identifiers. No one, including any Issuer, can create another U-Prove token with a matching token identifier. This makes the token identifiers of presented U-Prove tokens ideally suited for Verifiers as account indexes for Provers in their domains.

Once the token identifier of a presented U-Prove token has been hooked up to an account, the Verifier need not verify the Issuer's signature again in subsequent visits with the same U-Prove token. In fact, if the Verifier stores the U-Prove token with the account then in subsequent visits the Prover need merely send the token identifier and a new presentation proof created with the private key of the U-Prove token.

Similarly, U-Prove tokens can be hooked up to legacy accounts that are access-protected using a conventional authentication method (e.g., a password, Kerberos ticket, or X.509 certificate) by asking Provers to

[6] The term "account" is used here in the broadest possible sense, i.e., any set of attributes that pertains to a particular entity; it includes such notions as profiles, records, and dossiers.

authenticate one last time using the legacy authentication method; following this, the token identifier of a presented U-Prove token can be hooked up to the legacy account of its Prover.

As we will see in Section 4.2, unwanted account linkages can be prevented by hooking up different U-Prove tokens to different accounts.[7] In this manner Provers can protect their privacy interests while Verifiers can reliably identify and authenticate Provers in their own domains.

When Provers and Verifiers are not looking to build account relationships that span multiple sessions, Provers may use a new U-Prove token for each session. In these cases there is no point for Verifiers in using token identifiers for account indexing. However, there may be other reasons to compute and store the token identifiers of presented U-Prove tokens. Notably, the token identifier of a U-Prove token is also useful for revocation (see Section 5.2) and for keeping track of the number of token uses.

Because the token identifier of a U-Prove token is computed by hashing inputs that the Issuer and the Prover generate mutually at random, it is a unique random number. Applications may require the presentation of U-Prove tokens that specify "meaningful" identifiers, such as human-readable names and contact addresses; these can be encoded at issuance time into U-Prove tokens in the form of attributes. Alternatively, or in addition, meaningful identifiers can be securely bound to U-Prove tokens at presentation time, as we will see in the following section.

It is sometimes desirable to derive a persistent identifier from a token attribute rather than the token cryptographic material. This can be achieved by presenting a scope-exclusive pseudonym; see Section 5.4.

3.6 Proof of token presentation and Prover signatures

The transcript of an execution of the presentation protocol between a Prover and a Verifier constitutes a cryptographic proof that the presented U-Prove token, including the disclosed attributes, was presented by a party knowing its private key. Anyone can verify this cryptographic proof at any later time by applying the exact same procedure as the Verifier applies when verifying the token presentation. Since these cryptographic proofs are unforgeable even if Verifiers and the Issuer collude and examine arbitrarily many presentations of the same U-Prove token, they can serve as unforgeable entries in an audit log. Verifiers and Provers can store or forward these for inspection by auditors and other parties. See Figure 3.

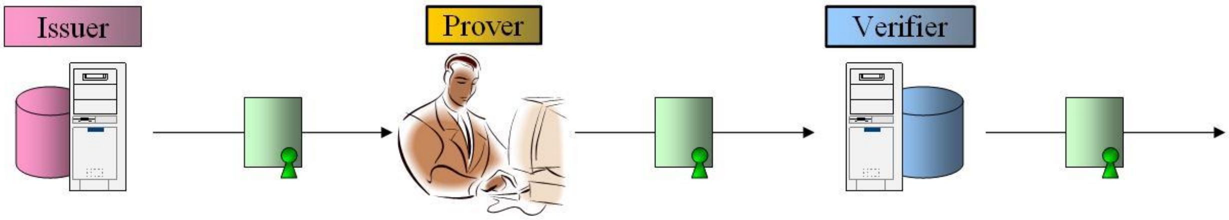


Figure 3: Proof of the presentation of a U-Prove token

In the U-Prove presentation protocol, the Prover and the Verifier can enrich the cryptographic proof of the presentation by including application-specific information (such as, for example, the time and date of the interaction, the actions taken by the Verifier upon accepting the U-Prove token, and the Prover's identifier in the Verifier's domain) in the presentation challenge in the form of a *presentation message*. This message must be supplied by or agreed upon by the Prover; this enables Provers to boycott inappropriate choices. The resulting cryptographic proof also establishes that the private key of the U-Prove token was applied to the

[7] Naturally, it may be possible to correlate the information held in the accounts. This is entirely application-specific and outside the scope of the U-Prove technology.

message. As such, the transcript of the token presentation can be considered to be an interactively formed signature of the Prover on the message.

Instead of presenting a U-Prove token to a Verifier, its Prover can also use it in a non-interactive fashion to digitally sign data of any type, such as documents, Web forms, and instructions. See Figure 4. As with interactively formed signatures, a Prover can safely create arbitrarily many non-interactive signatures with the same U-Prove token (unless reuse limitations prevent such).

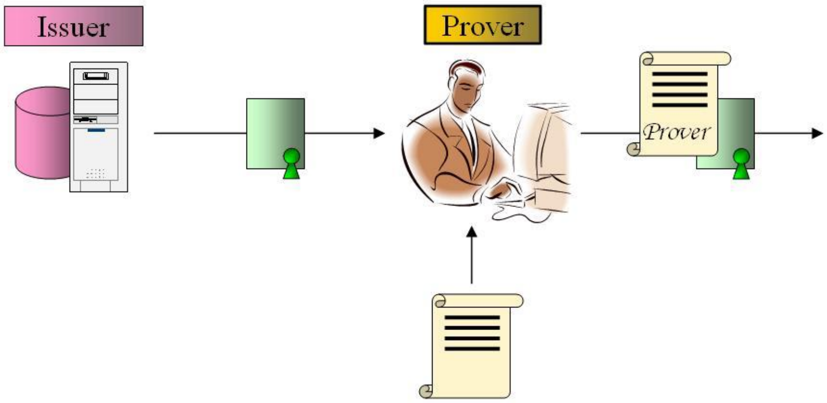


Figure 4: Signing data with a U-Prove token

4 Privacy features

This section describes the privacy features of the U-Prove technology as specified in [UPCS]. All privacy features hold even in the face of collusions between Issuers and Verifiers; they cannot learn more than what can be inferred from the attributes that are expressly disclosed by Provers. This seemingly very strong privacy guarantee is not a new concept: today, when you spend a coin, cast a vote, or present a movie or bus ticket, you are anonymous.

4.1 Untraceability

In spite of being an essential party to the issuance protocol, the Issuer never sees its own digital signature on an issued U-Prove token, nor does it see the public key of the U-Prove token. As a result, U-Prove tokens do not contain any inherent correlation handles that can be used to associate their use to their issuance; the only information in U-Prove tokens that may create correlations between issuance and use are application-specific attributes that may have been encoded into the token information fields. See Figure 5.

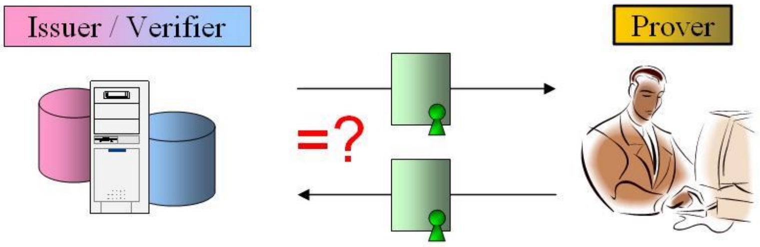


Figure 5: Untraceability property

This untraceability property holds unconditionally in the following sense: the Issuer and all Verifiers cannot learn even a single bit of information beyond what can be inferred from the disclosed attributes in presented U-Prove tokens, even if they would collude from the outset (indeed, they may be the same entity) and would have unlimited[8] computing time and resources at their disposal in a coordinated attempt to (1) build cryptographic backdoors into the Issuer parameters, (2) strategically deviate from the U-Prove token protocols, and (3) analyze the resulting protocol data flows. In other words, each Prover need merely trust the proper behavior of his or her own computing device.

We stress that Provers need not be anonymous or pseudonymous vis-à-vis the Issuer to enjoy the untraceability property. This restriction would greatly reduce the power of U-Prove tokens: without "knowing" Provers within its own context the Issuer may be unable to verify Prover qualifications, cannot limit the number of U-Prove tokens a Prover can obtain, cannot prevent Provers from transferring U-Prove tokens to others, cannot revoke the U-Prove tokens of Provers on the basis of their identity, and so forth.

The encoding of "rich" attributes into the token information field may contravene the privacy benefits of untraceability. The presentation of a U-Prove token with a highly granular expiry date, for example, may well be uniquely traceable. Even if only a limited number of attribute values may be encoded, a malicious Issuer could try to earmark U-Prove tokens by encoding invalid or unique values. Concerned Provers should inspect any information that is to be encoded so that they can boycott inappropriate choices. Alternatively, or in addition, information could be encoded in a manner that enables the Prover to selectively hide some or all of that information when presenting the token; see Section 4.3 for details.

[8] This assumes that Provers use truly random numbers in the issuance protocol. Using a high-quality pseudo-random number generator lowers the complexity to "computationally hard."

From a privacy perspective long-lived U-Prove tokens are preferable over on-demand U-Prove tokens, since the latter may be traced by matching the time of presentation to the time of issuance. This does not render the untraceability property useless for on-demand U-Prove tokens, however. Firstly, tracing on the basis of timing information may have significant error probability. Secondly, Provers need not worry that the presentation transcripts forwarded by Verifiers enable tracing by third-party recipients: presentation transcripts cannot be traced any more than the U-Prove tokens that gave rise to them.

4.2 Unlinkability

Similarly, the use of a U-Prove token cannot inherently be correlated to uses by the same Prover of other U-Prove tokens, even if the Issuer identified the Prover and issued all of the Prover's U-Prove tokens at the same time. See Figure 6. This unlinkability property holds unconditionally in the same sense as the untraceability property. On the other hand, multiple uses of the same U-Prove token are linkable; this enables Provers to establish relationships and build reputations with Verifiers in repeat visits.

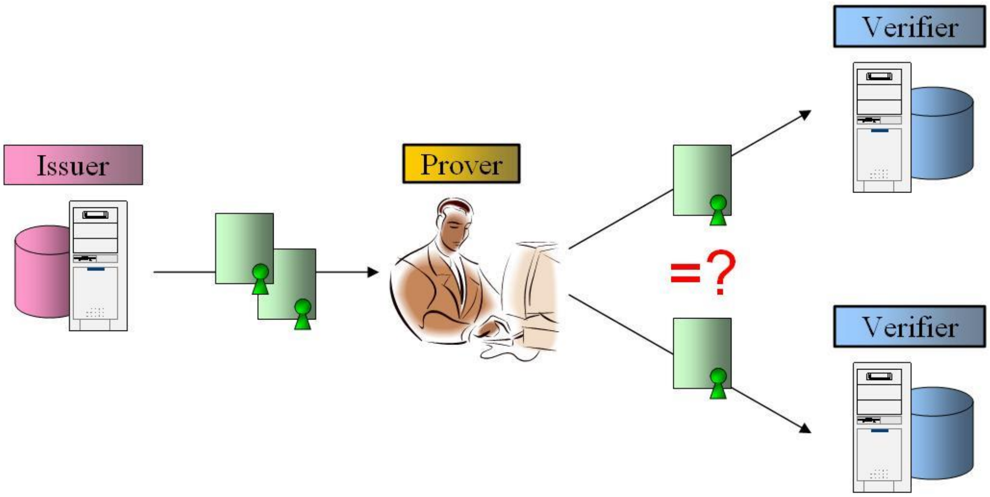


Figure 6: Unlinkability property

When the Issuer and the Verifier(s) are the same party, the untraceability and unlinkability of U-Prove tokens may be exploited by Provers for the purpose of remaining anonymous in their authenticated interactions with that party. When the Issuer and the Verifiers are different parties, the privacy benefits of U-Prove tokens may have little to do with Prover anonymity:

- Hooking up different U-Prove tokens to different accounts ensures that Verifiers and Issuers cannot compile all account information they hold on the Prover into one dossier. This does not mean that Provers are anonymous or pseudonymous vis-à-vis Verifiers; as explained in Section 3.5, token identifiers may be hooked up to meaningful "local" identifiers, such as usernames.
- Independent of any Prover interests, the untraceability and unlinkability of presentation transcripts may also be of interest to Verifiers, who may have their own privacy, autonomy, security, or competitive intelligence concerns vis-à-vis third parties. Without untraceability the Issuer and possibly other parties can infer the identities of a Verifier's clients by inspecting forwarded presentation transcripts, and without unlinkability client activities can be profiled.

By prudently using different U-Prove tokens, Provers can segment their activities into unlinkable "identity domains." Within each domain Verifiers can link and share Prover-related information; across identity domains, however, data linking and sharing are impossible unless facilitated by the Prover. This segmentation also prevents identity theft in one domain from spilling over to other domains.

Any information that Provers knowingly or unknowingly reveal outside the data flows in the U-Prove token protocols may increase the ability of Issuers, Verifiers, and others to trace or link their U-Prove tokens. In particular, the U-Prove privacy properties do not prevent tracing or linking of U-Prove tokens on the basis of the Prover's IP address. IP addresses do not contravene the privacy properties of U-Prove tokens, however:

- By way of analogy, while video cameras would make it easier to trace in-store cash payments, the resulting loss in privacy is not comparable with the privacy loss when using credit cards; in the latter case central parties automatically get to see who paid how much at which merchant at what time, whereas in the former case such tracing requires collusion with merchants and internet service providers, which involves actual work and a significant possibility of identification errors. Indeed, it cannot be assumed that one IP address maps to one user: multiple users may share a single computer; a user may not have the same IP address across different sessions, on account of dynamic address assignment; multiple users often share a small number of IP addresses on account of network address translation; and users may deploy anonymous remailers or transmit from a computer that is part of a network located behind a firewall.
- The privacy properties of the U-Prove technology are of interest even in environments where linking and tracing are easy. Namely, Verifiers may have an interest in them; presentation transcripts and Prover signatures cannot be traced and linked any more than the U-Prove tokens that gave rise to them.

4.3 Selective disclosure

Instead of, or in addition to, encoding application-specific attributes into the token information field of a U-Prove token, the Issuer can encode it into the *attribute fields* of the token. At set-up time, when specifying its Issuer parameters, the Issuer can determine how many such fields each issued token will contain. As with the token information field, any information to be encoded into any of the attribute fields must be supplied by or agreed upon by the Prover. Information can be encoded into attribute fields either directly (assuming it is sufficiently short) or indirectly (in which case it is compressed by a hash algorithm).

When using the U-Prove token, the Prover can decide for each attribute whether to hide or disclose its contents. Any field contents that the Prover does not disclose remains hidden from anyone including the Issuer, and any correlation powers that would arise when disclosing those contents are eliminated; both of these privacy properties hold in the same unconditional sense as the untraceability and unlinkability properties. See Figure 7.

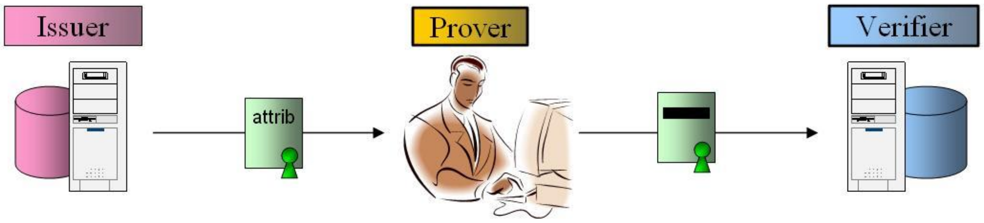


Figure 7: Selective disclosure of attributes

This selective disclosure mechanism enables the Prover to disclose attributes in a U-Prove token in a granular manner. The number of attribute fields and the manner in which the Issuer encodes attributes into these fields determine the degree of disclosure granularity available when using the token. Clearly, if the Issuer encodes each attribute into a separate attribute field then the Prover will be able to disclose any subset of those attributes.

The disclosure of the contents of some or all of the attribute fields is accomplished as part of creating a presentation proof or a signature. Since many presentation proofs may be created with the same U-Prove token, a Prover can opt to hide the contents of the attribute fields from some Verifiers while disclosing it to others. Likewise, a Prover may initially hide the information from a Verifier but disclose it later on in the same session or in another session with the Verifier. Disclosure negotiation is outside the scope of the U-Prove technology.

4.4 Prover information field

Instead of, or in addition to, the Issuer encoding information into the token information field or the attribute fields of a U-Prove token, at issuance time the Prover can encode application-specific attributes into the *Prover information field*. Any information encoded into this field remains invisible to the Issuer but is invariably disclosed when using the U-Prove token; without it the Issuer's signature in the U-Prove token cannot be verified. However, this does not enable anyone including the Issuer to correlate the presentation of the U-Prove token to its issuance.[9] See Figure 8.

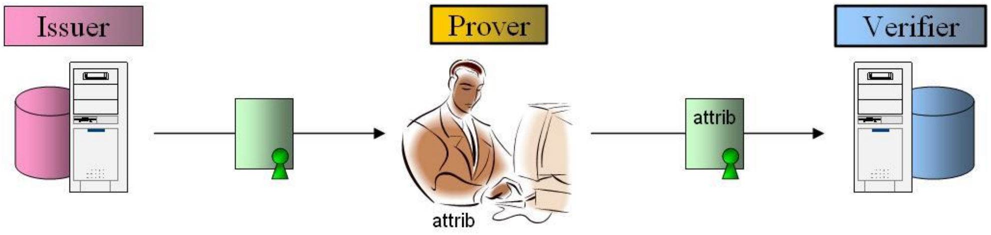


Figure 8: Prover information field

This privacy property holds unconditionally in the same sense as the untraceability and unlinkability properties.

The Prover information field enables Provers (possibly at the request of Verifiers) to bind attributes to U-Prove tokens at issuance time, without enabling the Issuer to learn the information or to trace the U-Prove tokens. For example, the Prover information field can be used to specify a human-readable username or an encryption key (to protect communications with Verifiers). The Prover information field is particularly useful for on-demand U-Prove tokens: by encoding Verifier nonces into this field, Provers can prevent these nonces from acting as correlation handles and can prevent Verifiers from covertly encoding messages for the Issuer into them.

[9] This property is not achieved by having the Issuer encode an encryption or hash digest of the attributes, since the resulting value would be a unique correlation handle.

5 Additional security measures

This section describes optional security measures for protecting U-Prove tokens. All measures are software-only and, unless otherwise mentioned, fully preserve the privacy properties described in Section 4.

5.1 Discouraging transferring of U-Prove tokens

To deter Provers from transferring (copies of) their U-Prove tokens to others, code obfuscation measures may be applied to prevent Provers from accessing the private keys of their own U-Prove tokens. Measures may also need to be taken to prevent Provers from obtaining U-Prove tokens from an Issuer while borrowing the identities of other Provers. Both of these are outside the scope of the U-Prove technology.

At the cryptography level, a technique is available for the Issuer to deter Provers from transferring their U-Prove tokens to others. Namely, by virtue of the cryptographic design of the U-Prove protocols it is impossible to use a U-Prove token without knowing the contents of all of its attribute fields, even if these contents are not disclosed. Consequently, the Issuer can discourage transferring of a U-Prove token by encoding information into the attribute fields that is confidential to its Prover, such as a credit card number, a password, or a secret key. To transfer the token, its Prover would also need to provide this confidential information.[10]

Protection from a Device also makes transferring of tokens difficult; see Section 6.

5.2 Revoking U-Prove tokens

A presented U-Prove token can be revoked by blacklisting its token identifier; as discussed in Section 3.5 this value is guaranteed to be unique even across U-Prove token issuances from different Issuers. As with X.509 certificate revocation, one can use either CRLs or an online certificate status responder (or any variation or optimization thereof). This revocation method is effective when a Prover wants to self-revoke a U-Prove token or when a Verifier stops accepting the U-Prove token (e.g., because its Prover has abused the Verifier's service).

In some cases, an Issuer may want to stop a particular Prover from using U-Prove tokens that have already been issued. For example, the Prover may no longer be qualified to use previously issued U-Prove tokens or the attributes contained in the token information fields may have become invalid. When untraceability is not a requirement, the Issuer can simply encode into the token information field of each U-Prove token a value that is unique to its Prover (e.g., an account number) or to the U-Prove token itself (e.g., a serial number); this value can be blacklisted using the same methods that are available for X.509 certificates. Alternatively, Verifiers could insist that Provers present on-demand U-Prove tokens; revoked Provers will not be issued new U-Prove tokens.

The U-Prove technology also allows long-lived U-Prove tokens to be issued in such a manner that Issuers or Verifiers can revoke them even though they are unlinkable and untraceable. As we will see in Section 6.2, a trusted security device protecting issued tokens can enable token revocation. An alternative cryptographic technique not provided in V1.1 of the cryptographic specification also provides this capability; see Appendix 1.

5.3 Preventing reuse and discarding of U-Prove tokens

A Verifier that wants to locally limit reuse of an on-demand U-Prove token can simply do so by keeping track of the number of times it has been used. To limit reuse of long-lived U-Prove tokens that do not specify a designated Verifier, Verifiers can do a real-time lookup in a central database that keeps track of the number of times each U-Prove token (possibly indexed by its token identifier) has been used. The U-Prove technology also allows long-lived U-Prove tokens to be issued in such a manner that they can be traced if and only if they are

[10] Technically, the confidential information should be encoded directly into the attribute fields, without hashing the attribute value encoding it. See [UPCS].

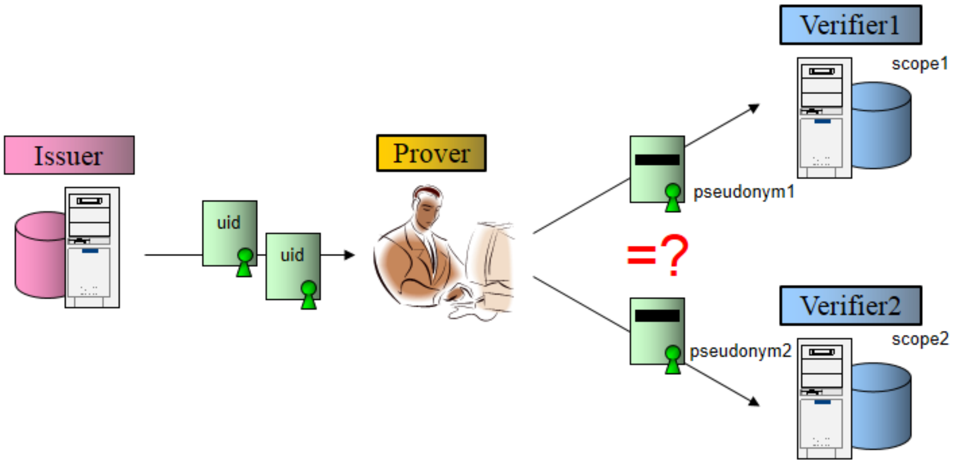
used more times than allowed, but this feature is not provided in V1.1 of the cryptographic specification; see Appendix 1.

To prevent discarding of a U-Prove token that specifies “negative” attribute information that its Prover would rather never disclose, the Issuer can encode a “positive” attribute into the same token that must always be shown to gain access to a Verifier’s service. Encoding the negative information into a token attribute field enables the Prover to disclose it only where required while at the same time being prevented from denying its existence by discarding the token altogether.

5.4 Scope-exclusive pseudonyms

It is often desirable for the Prover to present a persistent identifier to a Verifier that is derived from a permanent attribute value rather than being tied to a specific U-Prove token. This can be done by presenting a scope-exclusive pseudonym.

To achieve this, the Verifier specifies in its presentation message a unique scope identifier that will be used to derive a pseudonym that is exclusive to that scope and a designated token attribute value. Two pseudonyms derived from two different scopes are different and unlinkable, but all pseudonyms derived for the same scope and token attribute will identical.



This is useful to recognize repeat visitors when they use different tokens (from different devices, or over time, as they expire and get renewed), or to limit the number of times a resource can be used (e.g., an online poll where users can only give their opinion once per poll).

When presenting a Device-protected token (see Section 6), the Device can also be used to derive a pseudonym from its secret key. Since the issuer does not know the Device’s secret key, the resulting pseudonym is also untraceable (see Section 4.1).

When used, scope-exclusive pseudonyms are packaged into the presentation proofs.

6 Device-based security measures

Optionally, an Issuer can issue one or more U-Prove tokens to a Prover in such a manner that the private key of each U-Prove token is split between the Prover's own device and a trusted computing device. The resulting U-Prove tokens cannot be used without the assistance of the trusted device, which we will simply refer thereafter as the *Device*.

A Device may take the form of a tamper-resistant computing device (such as a smart card or a USB key with a CPU), a tamper-resistant chip (such as a TPM), a software-only emulation thereof (such as code running in a secure subsystem of the operating system or a "virtual" smart card that relies on code obfuscation techniques), a mobile smartphone, or an on-line service out of the Prover's reach (which could play the same role for many Provers). Devices may incorporate biometric access controls to provide greater security.

To create a presentation proof (defined in Section 2.5 of [UPCS]) a Device-protected U-Prove token, the Prover's computing device must engage in a compact challenge–response protocol with the Device. The Prover's device combines the Device's response with its own portion of the presentation proof to complete the creation of the presentation proof. See Figure 9.

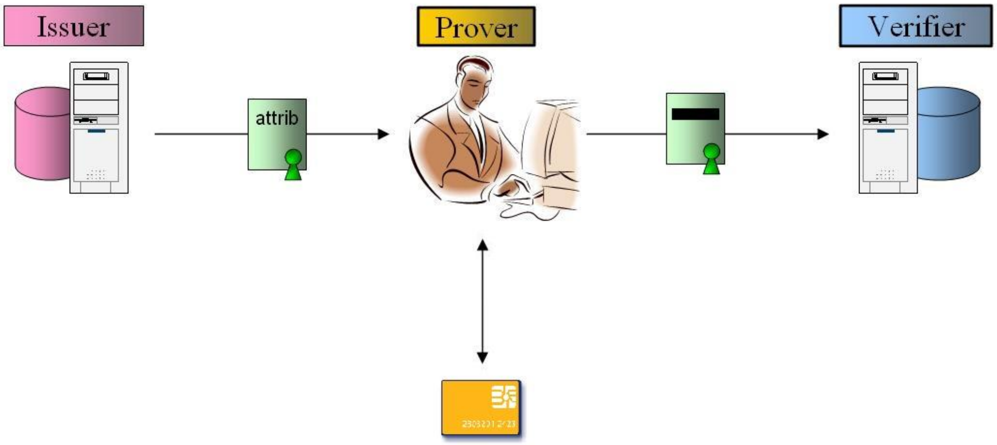


Figure 9: Device-protected U-Prove token

A resource-constrained Device suffices to achieve high performance for any number of Device-protected U-Prove tokens: computationally expensive operations can be securely offloaded from the Device. Furthermore, any U-Prove token-related information other than the Device's portion of the private key is typically stored and managed by the Prover's device.[11] These performance benefits also help to minimize the risk of side-channel attacks and leave more room on Devices for the implementation of specialized countermeasures.

Devices can be programmed so as to refuse to assist in any unauthorized uses of the U-Prove tokens they help to protect, throughout the entire U-Prove token lifecycle. At the same time, it is impossible to program a Device in such a manner that third parties (including collusions between the Issuer and Verifiers) can bypass any of the privacy properties described in Section 4. In particular, the Device cannot leak any information through its response: the Prover "randomizes" it prior to assembling it with its own portion. Furthermore, unless enabled by the Prover (see the following sections) the Device cannot learn any information about the U-Prove tokens it helps protect; notably, the Device cannot learn any attribute information they may contain nor the message signed by the Prover in the presentation protocol.

[11] It might be desirable however to use the Device as a secure roaming store for the issued U-Prove tokens.

The Device-protection feature provides many benefits over having a trusted device store and manage all of user's U-Prove tokens; see Chapter 6 of [Brands] for a complete treatise.

6.1 Preventing unauthorized Prover behavior

A Device that acts in the security interests of the Issuer (and/or Verifiers and/or Auditors) can be exploited to strengthen the software-only protections of U-Prove tokens discussed in Section 5 and to provide additional protections that cannot be achieved by software-only techniques. For example:

- The Device can enforce non-transferability of Device-protected U-Prove tokens by requiring Prover authentication prior to activation. This is particularly effective when requiring a biometric.
- The Device can prevent the Prover from discarding U-Prove tokens that contain unfavorable attribute information. An attempt to muzzle the Device could cause it to enter into suspension mode, preventing the Prover from obtaining or using any other Device-protected U-Prove tokens.
- The Device can prevent the unauthorized reuse of limited-use U-Prove tokens by maintaining an internal counter and refusing to assist the Prover in using a U-Prove token once its use limit has been reached.
- The Device can refuse to assist in using expired U-Prove tokens, assuming it has access to a synchronized clock.

6.2 Dynamic policy enforcement

The Verifier can include an application-specific message in the presentation challenge for inclusion in the Prover's own challenge to the Device. The Prover cannot omit or modify this Device message without rendering the Device's response useless. The Device message must be agreed upon by the Prover and can be inspected, enabling the Prover to boycott inappropriate messages.

The Device message is useful for any of the security measures described in Section 6.1 in situations where the Device protects multiple U-Prove tokens with different protection characteristics. By way of example, if the Prover has U-Prove tokens with different expiry dates or reuse limits, this information can be passed to the Device via the Device message.

More generally, the Device message can be exploited by the Device to dynamically enforce security policies at U-Prove token use time. In the Device message a Verifier can specify, for example, a security policy, some of the data transmitted by the Prover (e.g., attribute information encoded into the U-Prove token), U-Prove token protection characteristics (encoded, e.g., in the Token Information field), and Verifier-specific information (e.g., the Verifier's identity). The Device can inspect this data to make a policy decision as to whether or not to assist the Prover in using the U-Prove token. In case of non-compliance the Device could enter into suspension mode.

The Device message also enables the Device to do internal bookkeeping in the interest of the Issuer for other parties. For example, the Device can keep track of the identities of Verifiers, U-Prove token use times, and other U-Prove token use details. This audit log can be taken into account by the Device in future policy decisions and may be useful to settle disputes.

Assuming that a unique Device identifier is known to the Issuer, the Device message can also be exploited to enable the Issuer to revoke the Prover's untraceable long-lived U-Prove tokens on the basis of the Prover's identity with the Issuer. Namely, the Device message can be used to specify blacklisted Device identifiers.

6.3 Preventing local attacks on Provers

Rather than serving only the interests of non-Provers, a Device can also serve to protect its own Prover against local attacks on the Prover's device by malware and individuals with direct access. Indeed, Device-protected U-Prove tokens may be preferable for Provers even if Issuers, Verifiers, and Auditors do not require them. For example, a Device can protect against malware running on the Prover's computer that attempt to steal (the private keys of) the Prover’s U-Prove tokens or trick the Prover into using U-Prove tokens at untrustworthy web sites that are not whitelisted by the Device.

References

[Brands] Stefan Brands. *Rethinking Public Key Infrastructures and Digital Certificates.* The MIT Press, August 2000. http://www.credentica.com/the_mit_pressbook.html.

[UPCS] Christian Paquin and Greg Zaverucha. *U-Prove Cryptographic Specification V1.1*. Microsoft Corporation, April 2013. http://www.microsoft.com/u-prove.

Appendix 1 Capabilities not included in V1.1

This release makes available the fundamental features of the U-Prove technology for a broad range of use cases and scenarios. However, many more U-Prove features are possible, including the following ones not yet available in the U-Prove Cryptographic Specification V1.1 [UPCS]. It is possible, however, to make use of a built-in extension mechanism to extend the capabilities of the current specification. Indeed, Provers can cryptographically commit to token attributes to allow external modules to compute feature-specific proofs to complement a presentation proof.

- **Privacy-preserving revocation:** Long-lived untraceable U-Prove tokens can be issued (with the cooperation of the Prover in the issuance protocol) in such a manner that they can be revoked. The following privacy-preserving revocation techniques are available:
    - **Issuer-driven revocation:** An Issuer can revoke one or more tokens of a known Prover by blacklisting a unique attribute encoded in these tokens (even if it is never disclosed at token use time).
    - **Verifier-driven revocation:** A Verifier can revoke unlinkable tokens of an unknown Prover by blacklisting a unique attribute common to all tokens that is disclosed at token use time but was not seen by the Issuer.
    - **Community-driven revocation:** Honest Provers can prove to the Issuer they did not engage in a particular fraudulent/suspect presentation.
- **Proving attribute properties:** The Prover can prove properties of the attributes encoded in a U-Prove token, without destroying the verifiability of the Issuer's signature. Nothing more can be inferred by Verifiers, even if Issuers and Verifiers collude. Among others, the Prover can prove that an attribute does not have a particular value or that its value is contained in a specific interval or set.
- **Proving attribute properties across tokens:** The Prover can minimally disclose information derived from multiple attributes encoded in different U-Prove tokens issued by one or more Issuers. Verifiers cannot infer anything beyond what is explicitly disclosed, even if they collude with Issuers. For example, the Prover can prove that two tokens contain the same or different attributes without disclosing them.
- **Limited-use tokens:** A U-Prove token can be issued (with the cooperation of the Prover in the issuance protocol) in such a manner that it cannot be presented in an untraceable manner more than a predetermined number of times; the token can be traced back to its issuance if and only if the token is used more times than allowed.
- **Zero-knowledge token presentation:** A Prover can present a U-Prove token in such a manner that the Verifier cannot prove to anyone else that the presentation occurred.
- **Censoring and uncensoring of cryptographic proofs of token presentation:** A Verifier can do its own selective disclosure on the cryptographic proof of a token presentation prior to forwarding it to third parties, and can later on provide some or all of the censored details without being able to lie about them.
- **Hiding the Issuer's identity:** A Prover can present a U-Prove token in such a manner that the Verifier can conclude only that the token was issued by one of a set of Issuers. This privacy guarantee holds even if Verifiers and Issuers collude, modulo any conclusions they can draw on the basis of token content encoded at issuance time and disclosed at presentation time.
- **Token recertification and updating:** An Issuer can re-sign a previously issued U-Prove token without knowing its attribute content (or possibly only a property of the encoded attributes). The Issuer can add value to or subtract values from previous encoded attributes when re-issuing.
- **Verifiable attribute encryption:** When presenting a U-Prove token, the Prover can include a verifiable encryption of an encoded attribute. The primary use case of this capability is "identity escrow," meaning that only one or more designated escrow agents will be able to trace token uses.

Appendix 2 Structure of a U-Prove token

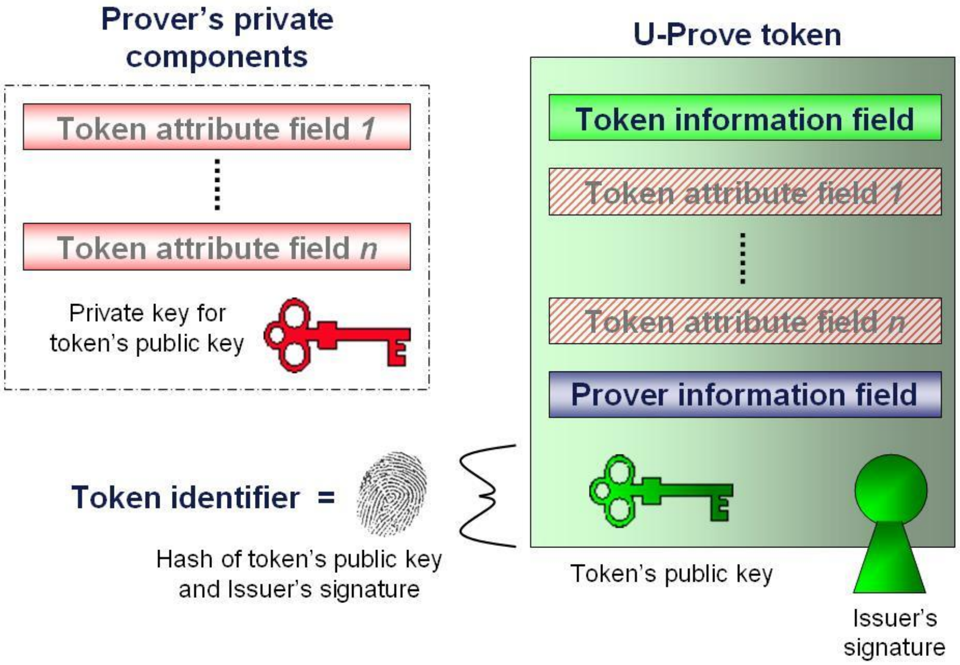


Figure 10: Structure of a U-Prove token