IRMA: practical, decentralized and privacy-friendly identity management using smartphones

Gergely Alpár[†‡], Fabian van den Broek[†‡], Brinda Hampiholi[‡],
Bart Jacobs[‡], Wouter Lueks[*‡], Sietse Ringers[‡]
[*]IMDEA Software Institute, Madrid, Spain
[†]Open University, Heerlen, The Netherlands
[‡]Radboud University, Nijmegen, The Netherlands

1. INTRODUCTION

Whenever we need to do something serious online, we need to authenticate ourselves. Not only do we tell the website who we are, we also tell Google or Facebook what we are doing if we use their convenient identity services. Often, however, we would not need to authenticate to prove that we are eligible to access a service. Hence, we unnecessarily reduce our privacy.

In theory, attribute-based credentials (ABCs) allow us to put privacy first. Using ABCs we can prove (without involving third parties) a minimal set of attributes about ourselves to access a service. In practice, however, service providers cannot easily use ABCs. Existing software focuses on the cryptography, leaving the building of the ecosystem and user-friendly applications to others.

In this talk we will introduce the IRMA project,[1] which focuses on making ABCs practical. IRMA builds an ABC ecosystem using open source software that is easy to use for users, service providers, and issuers. Users manage and control their credentials using a smartphone application: the IRMA app. Service providers and issuers can use the IRMA JavaScript library and the IRMA API server to interact with the IRMA app without needing to worry about (complicated) cryptographic operations.

2. ATTRIBUTE-BASED CREDENTIALS

Attributes, such as nationality, sex, name, social security number, and date of birth, describe individuals. An attribute-based credential is a cryptographic container of attributes that is issued and digitally signed by an issuer.

To prove properties about herself, a user can selectively disclose a subset of the attributes in a credential to a service provider. Even though some attributes might be hidden, the service provider can still verify the validity of the revealed attributes. Furthermore, disclosure proofs are unlinkable: service providers cannot distinguish different credentials with the same set of disclosed attributes.

ABCs are very privacy friendly. Users minimize the data they reveal to service providers by disclosing only relevant attributes. The unlinkability of proofs ensures that users are as anonymous as their disclosed attributes allow.

The IRMA project implements Idemix [1], an ABC scheme developed by IBM Zürich, and later refined in the ABC4Trust

[1]IRMA stands for 'I Reveal My Attributes', see `https://irmacard.org` and `https://privacybydesign.foundation/irma`. The following movie gives a quick introduction: `https://www.youtube.com/watch?v=q6IihEQFPys`

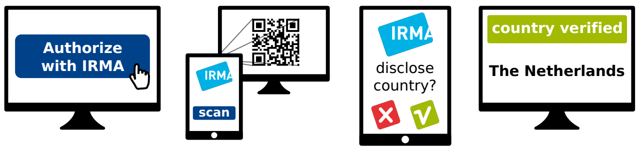

Figure 1: From left to right: example of using the IRMA app to disclose an attribute to a website.

project. In contrast to the Idemix cryptographic library, IRMA does not implement all possible cryptographic features. Instead, it focuses on the basic attribute-based credentials, preferring simplicity over complexity. In our experience, this simplicity does not restrict the system much in practice. For example, while we do not support domain specific pseudonyms or range proofs, we have found that these can often be simulated using simple attributes.

Restricting IRMA to only credentials and attributes makes the system simpler as a whole, and it has the additional benefit that users always have to make the same decision: do I want to show this attribute or not?

3. EASY FOR USERS

An attribute-based credentials ecosystem contains several attribute providers that issue credentials. These credentials can then be used at many service providers. To allow users to store, use and manage their credentials, the IRMA project provides a wallet-like smartphone application: the IRMA app (currently, only on Android). Earlier versions of IRMA used smart cards as a credential carrier [3], but we found that using them places a heavy burden on users.

Figure 1 shows how a user uses her app to show credentials to a website. To connect her IRMA app to the website, she first scans a QR code.[2] She is then asked to confirm the disclosure of attributes—to simplify the user interface, the app focuses solely on attributes rather than the credentials within which these are contained—before the result is sent back to the website. Users obtain credentials using a similar procedure. Because of the use of ABCs, showing credentials is decentralized: the application interacts directly with the service provider; the issuer is not involved.

Since all credentials are contained in the IRMA app, the user remains in control. Without access to the application, credentials cannot be shown. In fact, the user can determine

[2]When the user visits the website on her Android device, she is immediately redirected to the IRMA app.

when and where she reveals certain attributes. The application also allows the user to view credentials, remove them, and see a usage log.

4. EASY FOR SERVICE PROVIDERS AND ISSUERS

To provide strong privacy and security guarantees, ABCs require cryptographic techniques that are somewhat non-standard. The IRMA app handles the cryptography on the user's side. However, issuers and service providers need to perform similarly complicated cryptographic operations to issue new credentials and verify existing ones, respectively. While some libraries exist to handle the cryptographic details, they can be difficult to integrate into a website.

IRMA makes it easy for service providers and credential issuers to integrate ABCs into their platforms. Consider a website, i.e. service provider, that restricts access to certain resources. Using IRMA, it can specify its authorization policy in terms of attributes. The IRMA stack consisting of the irma.js JavaScript library, embedded in the website, and the IRMA API server, deployed by the service provider (or third party), takes care of all the details.

The entire interaction is depicted in Figure 2. To verify some attributes, the service provider makes a simple JSON description of these attributes and calls irma.js. The JavaScript library contacts the IRMA API server and displays the QR code (or redirects to the IRMA app directly on Android). The IRMA app obtains the request from the IRMA API server and, after user confirmation, returns the disclosure proof. Eventually, the library returns a JSON Web Token (signed by the API server) summarizing the requested attributes and their validity. Since JSON Web Tokens (JWTs) are standardized, they can easily be processed in the service provider's backend software.

Issuing credentials is equally simple. The issuer sends a JWT describing the credentials it wants to issue to the API server by making a simple irma.js call. The API server issues the requested credentials using the issuer's Idemix keys.

Issuers and service providers need to trust the API server. In particular, they trust the API server to correctly verify credentials and to issue credentials only when requested. Therefore, we made it easy for parties to configure and run their own API server. We fully expect large websites to run their own, whereas others might use verification or issuance as a (commercial) service. Having many API servers also mitigates the privacy risk that having only one API server would cause. However, to make it easy to get started with IRMA, we also run a public API server operating in a demo domain for testing purposes.

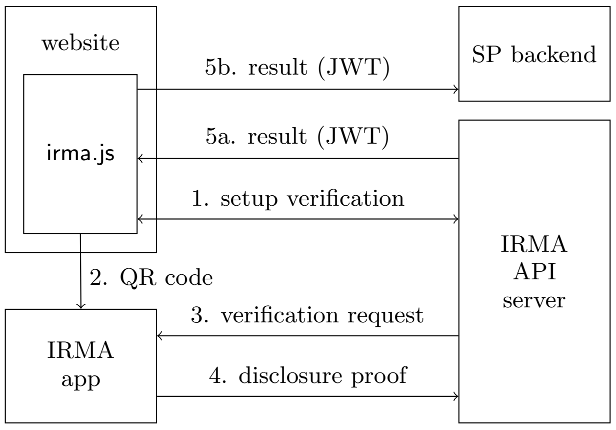

Figure 2: Example of interaction between irma.js, IRMA API server, IRMA app and service provider (SP) backend when verifying some attributes.

5. CHALLENGES

Deploying IRMA is not without its challenges. First of all, IRMA suffers from the classic chicken and egg problem: why would you want IRMA attributes when nobody uses them, and why would you issue IRMA credentials when nobody wants them? To ease bootstrapping within the Netherlands, we reissue attributes from several trusted sources: (1) Dutch passports and driver's licenses after electronically verifying them, (2) the bank identification system iDIN, and (3) the academic federated identity system SURFconext.

However, we believe that the flexibility of IRMA and the fact that everybody can issue and verify credentials easily makes it straightforward to try it out, even without bootstrapping. In particular small, self-contained projects that nevertheless require strong authentication can benefit from IRMA. In fact, our experiments show that it is simple for developers to integrate IRMA into their platforms.

Another challenge is how to offer good privacy protection. While IRMA protects the user's privacy at the application layer, the network layer may still deanonymize the user (for example because of her IP address). We plan to make the IRMA app communicate via Tor [2] in order to hide these network layer identifiers.

Finally, the IRMA app protects valuable credentials and keys. Any attacker that obtains these can easily impersonate the user. We have a preliminary solution to protect the user's keys, but for now this requires a strong assumption.

6. CONCLUSIONS

We hope that this talk inspires you to try out IRMA and use attribute-based credentials in your next project. We are looking forward to discuss what would make you use IRMA and how it could be improved. All IRMA code is open source and freely available on GitHub, see `https://github.com/credentials`. The IRMA app is also available in the Google Play store. We refer developers to `https://credentials.github.io` for an introduction to our ecosystem.

Acknowledgements.

We would like to thank Maarten Everts, Jaap-Henk Hoepman, Koen van Ingen, Ronny Wichers Schreur, and Pim Vullers for their earlier contributions to the IRMA project.

7. REFERENCES

[1] J. Camenisch and A. Lysyanskaya. A Signature Scheme with Efficient Protocols. In *SCN 2002*, volume 2576 of *LNCS*, pages 268–289. Springer, 2003.

[2] R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320. USENIX, 2004.

[3] P. Vullers and G. Alpár. Efficient selective disclosure on smart cards using idemix. In *IDMAN 2013*, volume 396 of *IFIP AICT*, pages 53–67. Springer, 2013.