Taler:
Usable, privacy-preserving payments for the Web

Jeffrey Burdges Florian Dold Christian Grothoff Marcello Stanisci

1. SYSTEM OVERVIEW

Content and services provided on the internet, such as reading a blog post or sending an email, tend to be of very small monetary value compared to traditional financial transactions. Currently the majority of online offerings are financed via advertisements. Any alternatives must reduce the mental and technical overheads of existing payment systems to handle micro-payments. Addressing this problem is urgent because advertising revenue is declining, and the Big Data business model where citizens pay with their private information in combination with the deep state hastens our society's regression towards post-democracy [3].

Taler is a new electronic online payment system that provides anonymity for customers. Here, *anonymous* simply means that the payment system does not involve any personal information from the customer, and that different transactions by the same customer are unlinkable. For strong anonymity, Taler usually needs to be used in combination with existing techniques, such as Tor and [1], to avoid circumstances leaking information about the customer's identity. The facts that the user does not need to authenticate, and that the merchant thus never learns sensitive personal information about the customer, improves usability and security: the payment process is simplified, the merchant's security requirements are dramatically reduced and the customer's risk of identity theft does not accumulate with every (micro-)payment.

Taler uses blind signatures [2] to create digital coins, and a novel "refresh" protocol to allow giving change and refunds while maintaining unlinkability. We will not go into the details of Taler's cryptographic protocols here[1] and instead focus on the high-level concepts to explain how the system works from the perspective of customers and merchants in the Taler system (Figure 1).

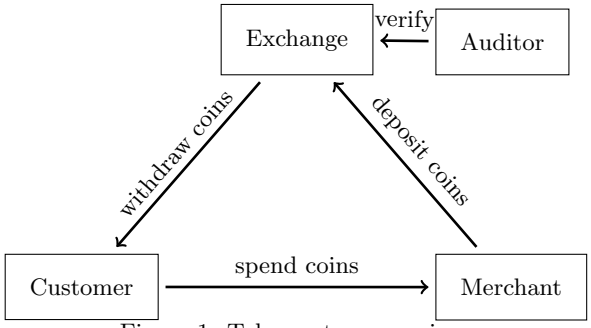

Figure 1: Taler system overview.

2. CUSTOMER PERSPECTIVE

In Taler, customers use a *wallet* to withdraw, hold, and spend coins. Withdrawing coins requires the customer to authenticate and to optionally authorize the specific transaction, e.g. via a PIN/TAN method as commonly used by banks. Afterwards, the customer can anonymously spend their coins by visiting merchants without having to authenticate for each transaction.

The wallet is implemented as a cross-platform browser extension. All cryptographic operations and access to sensitive data are executed in a component that is isolated from websites the user visits.

By necessity, the wallet leaks one bit of information to websites that the user visits, namely whether the wallet is installed and activated by the user. Websites cannot access the customer's balance or purchase history. This however also means that all cryptographic tokens of value are kept locally, and the customer is responsible for not losing them. Future versions of the wallet will provide encrypted backups and synchronization between the wallets of a user.

A common activity for online content is sharing and bookmarking. Taler specifically provides support to make this easy for the user. A resource that was purchased is identified by a unique *fulfillment URL* for each purchase of the resource.

Should the session state that allows the user to access the content be lost, visiting the fulfillment URL will transparently restore the session state by transparently replaying the payment with the same digital value

[1]Full documentation at `https://api.taler.net/`

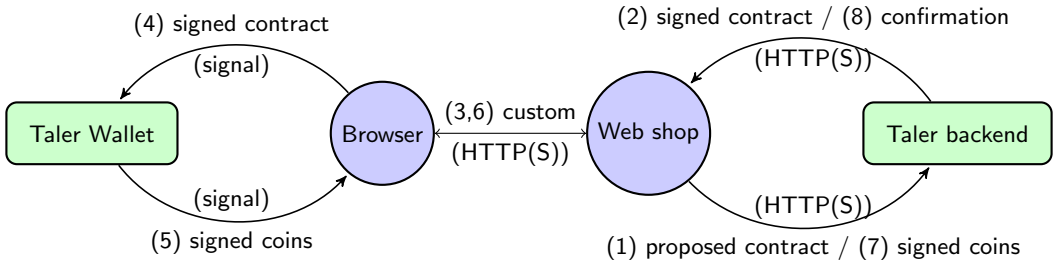

Figure 2: Both the customer's client and the merchant's server execute sensitive cryptographic operations in a secured background/backend that is protected against direct access. Interactions between the Taler components (Figure 1) are not shown. Existing system security mechanisms are used to isolate the cryptographic components (boxes) from the complex rendering logic of existing Web applications (circles).

tokens from the user's wallet. Replaying a contract is only allowed from the domain that the contract originated from, and thus does not allow arbitrary websites to obtain information about previous purchases that the customer made. Sharing the fulfillment URL with a user that did not pay for the associated digital contract will result in the expected behavior, namely that they receiving a new instance of the digital contract with the opportunity to pay for it.

The case where a user already payed for a resource and then visits the resource URL (instead of the fulfillment URL) after losing temporary session state is also handled as expected, since the wallet component will look for contracts that refer to the same resource.

While Taler is designed to work well with digital resources on the web, it can also be used for more traditional purchases. The resource that is being payed for then represents the shopping cart of items that are being purchased.

3. MERCHANT PERSPECTIVE

A new payment system must also be easy to integrate and deploy for merchants. Figure 2 shows how the security critical payment components of Taler interact with the logic of existing Web shops. First, the Web shop front-end is responsible for constructing the shopping cart. For this, the shop front-end generates the usual Web pages which are shown to the user's browser client front-end. Once the order has been constructed, the shop front-end gives a *proposed contract* in JSON format to the payment backend, which signs it and returns it to the front-end. The front-end then transfers the signed contract over the network, and passes it to the wallet. Here, the wallet operates from a secure background context on the client side, which allows the user to securely accept the payment, and to perform the cryptographic operations in a context that is protected from the Web shop. If the user accepts, the resulting signed coins are transferred from the client to the server, again by a protocol that the merchant can customize to fit the existing infrastructure.

Instead of adding any cryptographic logic to the merchant front-end, the generic Taler merchant backend allows the implementor to delegate handling of the coins to the payment backend, which validates the coins, deposits them at the exchange, and finally validates and persists the receipt from the exchange. The merchant backend then communicates the result of the transaction to the frontend, which is then responsible for executing the business logic to fulfill the order. As a result of this setup, the cryptographic details of the Taler protocol do not have to be re-implemented by each merchant. Instead, existing Web shops implemented in a multitude of programming languages can rather trivially add support for Taler by **(1)** upon request, generating a contract in JSON based on the shopping cart, **(2)** allowing the backend to sign the contract before sending it to the client, **(7)** passing coins received in payment for a contract to the backend and **(8)** executing fulfillment business logic if the backend confirms the validity of the payment.

To setup a Taler backend, the merchant only needs to configure it with the respective wire transfer routing details, such as an IBAN number. The customer's authentication of the Web shop continues to rely upon HTTPS/X.509.

4. CONCLUSION

We encourage everyone to try our prototype for Taler at `https://demo.taler.net/`.

Acknowledgements

This work benefits from the financial support of the Brittany Region (ARED 9178) and a grant from the Renewable Freedom Foundation.

5. REFERENCES

[1] E. Androulaki and S. Bellovin. Apod: Anonymous physical object delivery. In *Symposium on Privacy-Enhancing Technologies (PETS)*, 2009.

[2] D. Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer, 1983.

[3] R. Stallman. How much surveillance can democracy withstand? *WIRED*, 2013.