Demonstration of Privacy Protection Mechanisms for Document Storage Services

Maximilian Blochberger

blochberger@informatik.uni-hamburg.de

January 12, 2019

As privacy protection mechanisms are not necessarily visible to the user it is hard to demonstrate their effectiveness. We present several visualizations, which can be used for that purpose.

1 Introduction

Privacy protection mechanisms have to be integrated into applications, frameworks, and services. Many of these mechanisms work without the user noticing anything. When such mechanisms are in effect in the background, a demonstration of their effectiveness might prove difficult.

In this paper we present different visualizaitons that can be used to demonstrate the effectiveness of privacy protection mechanisms for document storage services. Document storage services can be used by multiple users in order to store their documents or files. Documents are usually tied to a user's account and are identified uniquely per user by their name and path. Lots of personal information can be inferred from data stored in such a service. Operators can abuse this information or it can unintentionally get leaked. Privacy protection mechanisms can be integrated to protect personal information.

However, demonstrating the effectiveness of such mechanisms by showing the contents of the database or log entries of the service is hard to comprehend. We therefore present several visualizations with the aim to make the effect of some protection mechanism more comprehensible.

2 Setting

In order to be able to demonstrate the effectiveness of such protection mechanisms we have developed a document storage, which can imitate a malicious service operator. This is done by collecting as much information as possible, including traffic data like IP addresses, geolocation of networks, and metadata like file sizes. The history of each document is preserved, so that even if a client issues a request to delete the document to the service, the serivce operator will still be able to access the file afterwards. He reports to the client, that the file has been removed, so from the client's perspective the service functions normally. The document storage is offered as a web service, which can be accessed using a REST-style API. Each HTTP request will be logged completely.

As the service including the visualizations or introspection APIs might be publicly accessible on the Internet, this behavior would violate privacy regulations of most countries. Therefore, the malicious behavior of the service is only enabled if the client opts in by setting the `X-AppPETs-BadProvider: 1` HTTP header. Additional HTTP headers are used to determine ground truth, e. g., if the identity of the client is protected by a specific mechanism, the identity can be provided additionally. This allows to evaluate the accuracy of heuristical inference attacks.

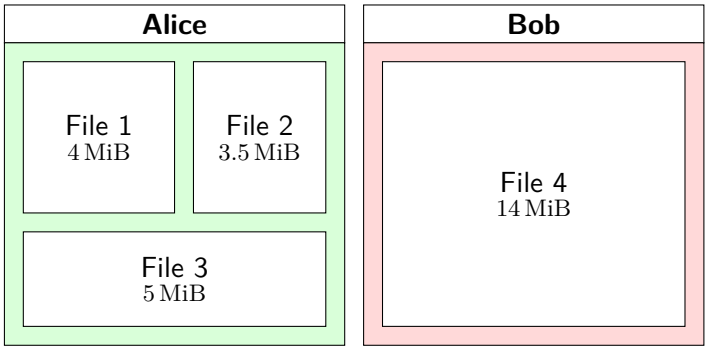

Figure 1: Visualizaton of files of different sizes belonging to two user accounts.

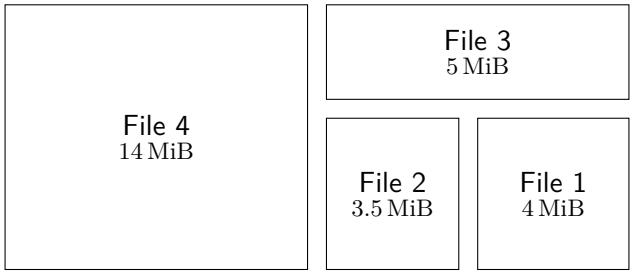

Figure 2: Visualizaton of files of different sizes, where the file's owner is not known.

The visualizations are offered as web pages by the service and supports animations as well as interactions.

3 Scenarios

We discuss some selected scenarios and present mock-ups for possible visualizations.

3.1 Content and Metadata

A document storage without privacy protection mechanisms stores the files as they are. They are tied to the account of the file owner.

The state of the document storage service can be visualized as a tree map [3]. The files are rectangular leaf nodes and their size is derived from the file's size. Files are grouped by the user account of their owner, as depicted in fig. 1. Additional metadata will be displayed when clicking on a file node, such as a preview of the content as well as a link to download the original file. Metadata of files could be used for further visualizations, in order to demonstrate what a service operator can learn about its users if they have access to the files contents. Possible visualizations with differing effects on the data subjects (file owners) are presented by Harkous et al. [4] and Krause [7].

3.2 Traffic Data

We further assume that content and metadata are protected in a way, that the service provider can not link them to specific user accounts, e. g., as presented by Blochberger [1]. The file sizes and file identifiers will be kept, so that differences can be spotted. The same previous visualization method can then not group files per user account anymore, as depicted in fig. 2.

We assume that the client communicates directly with the service. A malicious service operator can now link the files by the client's IP address, e. g., if Alice has the IP address `2001:db8::a` and Bob has the IP address `2001:db8::b`, the grouping can be restored as depicted in fig. 3. Note that the server might not know which IP address belongs to Alice or to Bob. The address can still be used to deduce which files belong to the same user.

The IP addresses can also be used to see where Alice and Bob where located when they mod-

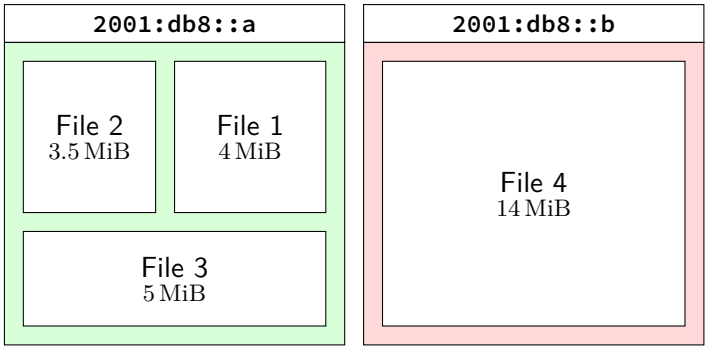

Figure 3: Visualizaton of files of different sizes, grouped by the sender's IP address.

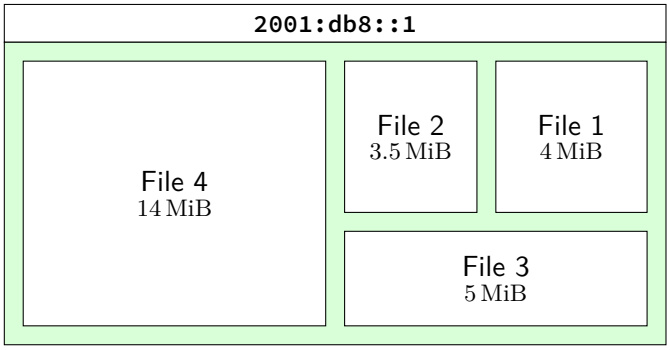

Figure 4: Visualizaton of files of different sizes, grouped by the sender's IP address. All senders use the same proxy server.

ified or retrieved files. As depicted in fig. 6, Alice interacted with a file 1 in Hamburg, then moved to Berlin and interacted with files 2 and 3. Bob is located in Nuremberg. A similar visualization based on EXIF metadata of photos has been proposed by Krause [7].

We want to protect the client's IP addresses as well and use two different methods for this purpose to highlight their differences from the service's perspective. First, the same proxy server with the IP address `2001:db8::1` is used for all clients, e. g., as presented by Panchenko et al. [8]. All users are in the same anonymity group. The service operator now only sees a single sender IP address, as depicted in fig. 4, which is quite similar to fig. 2. Second, an anonymous communication network is used, e. g., Tor or JAP. It is used in a way, that each request is sent through a different IP address with the prefix `2001:db8:1`. The information derived as depicted in fig. 5 cannot be used to reliably link files to either Alice or Bob.

The problem is not solved however, as different methods can be used to link multiple requests to the same user. Proxies usually attach additional traffic data that allows linkage, as presented by Perino, Soriente, and Varvello [9]. Eckersley [2] present fingerprinting techniques based on HTTP headers, and Husák et al. [5] present such techniques for TLS connections.

4 Demonstrator

An open source key-value storage service has been implemented in Python, which can be used to store documents[2]. Visualizations have been implemented with *D3.js* JavaScript library and are accessible through the service directly. Not all the proposed visualizations have been implemented as described at the time of writing. In addition to visualizations an introspection API was added that can be used to retrieve the data collected by the service. This can be used to realize visualizations on the user's device.

[1] Map taken from `https://pixelmap.amcharts.com`

[2] AppPETs/PrivacyService: Implementation of privacy-friendly services: `https://github.com/AppPETs/PrivacyService`

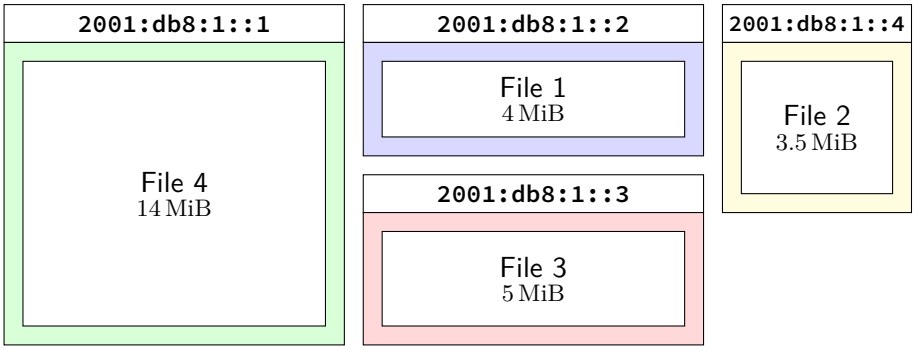

Figure 5: Visualizaton of files of different sizes, grouped by the sender's IP address. The senders use an anonymization service that uses a different IP address for each request.

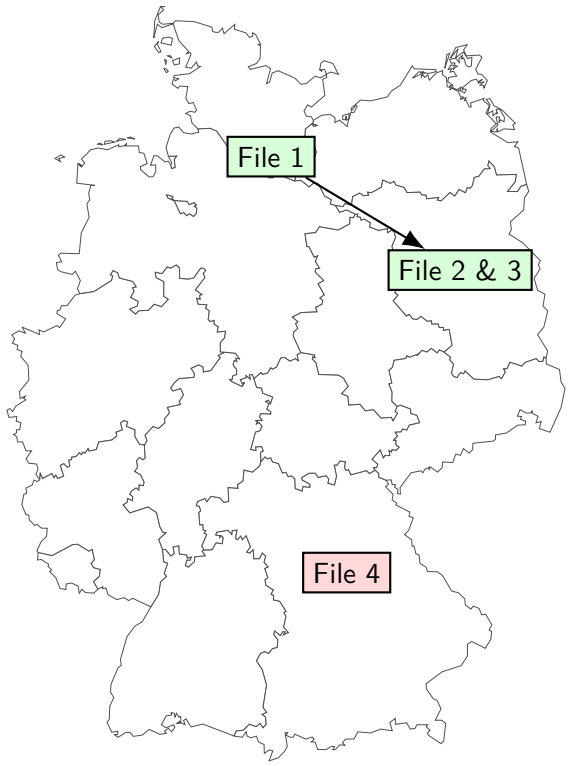

Figure 6: A map[1] showing the locations where files were uploaded.

5 Conclusion

Several visualizations have been proposed and a demo service was developed which implement some of these visualizations. A user study should be performed in order to validate that the visualizations are helpful in understanding effects of privacy protection mechanisms on document storage services.

Additional information could be collected from the client as well, which would allow creating even more helpful visualizations of data processing and data protection mechanisms. One example of visualizing data flows for an application was proposed by Kani-Zabihi and Helmhout [6].

The visualizations presented in this paper as well as visualizations from related work could also be used for interactive privacy policies as they might be able to explain complex data processing and protection mechanisms in a comprehensive manner.

Acknowledgements

This work was done in the AppPETs project[3] and supported by the BMBF.

References

[1] Maximilian Blochberger. *Key-value Storage with Cryptographic Client-separation*. White paper. Jan. 2019. URL: https://github.com/AppPETs/SecureKeyValueStorage-Whitepaper.

[2] Peter Eckersley. "How Unique Is Your Web Browser?" In: *Privacy Enhancing Technologies*. Vol. 6205. Lecture Notes in Computer Science. Springer, 2010, pp. 1–18.

[3] Martin Graham and Jessie B. Kennedy. "A survey of multiple tree visualisation." In: *Information Visualization* 9.4 (2010), pp. 235–252.

[3] AppPETs – Datenschutzfreundliche Smartphone Anwendungen ohne Kompromisse: https://app-pets.org

[4] Hamza Harkous et al. “The Curious Case of the PDF Converter that Likes Mozart: Dissecting and Mitigating the Privacy Risk of Personal Cloud Apps.” In: *PoPETs* 2016.4 (2016), pp. 123–143.

[5] Martin Husák et al. “HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting.” In: *EURASIP J. Information Security* 2016 (2016), p. 6.

[6] Elahe Kani-Zabihi and Martin Helmhout. “Increasing Service Users’ Privacy Awareness by Introducing On-Line Interactive Privacy Features.” In: *NordSec*. Vol. 7161. Lecture Notes in Computer Science. Springer, 2011, pp. 131–148.

[7] Felix Krause. *iOS Privacy: detect.location - An easy way to access the user’s iOS location data without actually having access*. Oct. 2017. URL: https://krausefx.com/blog/ios-privacy-detectlocation-an-easy-way-to-access-the-users-ios-location-data-without-actually-having-access.

[8] Andriy Panchenko et al. “SHALON: Lightweight Anonymization Based on Open Standards.” In: *ICCCN*. IEEE Computer Society, 2009, pp. 1–7.

[9] Diego Perino, Claudio Soriente, and Matteo Varvello. “Your Are ‘Proxy’ And I Know It.” In: *CoRR* abs/1612.06126 (2016).