Data-Leashing: Towards a Characterization of The Problem and Its Solution

Thowayba M. Elkaffash and Armstrong M. Nhlabatsi*
KINDI Center for Computing Research, Doha, Qatar
te2206792@qu.edu.qa, Armstrong.Nhlabatsi@qu.edu.qa,
*corresponding author

Abstract—Once sensitive digital content is shared or leaked, it is not feasible for the content owner to control it. This has led to issues whereby the privacy of leaked information is compromised. There is a need for a solution to allow digital content creators to maintain control of their digital assets even after they have been shared, downloaded, and saved by the receiver. In this paper, we propose a novel approach called *Data-Leashing* - inspired by the common practice of putting a *leash* on a *pet*. The proposed approach solves the *post-sharing control of data* problem by encapsulating digital assets (such as personal pictures) in a special container program that protects the digital assets. The specialized container program allows only authorized users to access protected digital assets. Should the receiver of the digital asset decide to (re)share the protected digital assets with unauthorized receivers, the container program notifies the asset's owner and gives them the option to either grant or deny access to the previously unauthorized subjects. In this paper, we outline the key features and properties of the proposed Data-Leashing approach.

Keywords–data leashing, post-sharing control, digital assets

1. Introduction and Motivation

Various security concerns arise with the increasing number of unprotected shared files on social media. Once someone shares a file on any digital platform, particularly on social media platforms (e.g. WhatsApp, Facebook, and Instagram), they lose control over who can use and share these data. To demonstrate the magnitude of this problem, consider the following: After someone shares a picture with another user, the receiver can download it to use and share without the owner's consent, and the owner cannot remove access from them after they have saved the content on their device.

Although it is possible on most cloud storage services (e.g. Google Drive) to deny access to a user after a file is shared, this revocation of access is only possible within the cloud platform. If the file has been downloaded to the receiver's device before access is revoked, it will remain stored on the receiver's device, and they will still be able to access it. As a result, users cannot preserve their right to control the content they own after sharing it. This is because the existing conventional approaches to controlling access to digital assets allow downloading the content in its original form, thus giving unlimited full access to the receiver.

In this paper, we propose a novel approach called *Data-Leashing* - inspired by the common practice of putting a *leash* on a *pet*. The Data-Leashing problem is a security issue that occurs when data is transferred between two or more parties without any mechanisms in place on how the owner of the data can control it after it is shared. This can leave the data vulnerable to theft, tampering, or other malicious activities. In conventional systems, Data-Leashing can be prevented by implementing access control measures such as authentication, authorization, and encryption. We urge that these approaches are insufficient for addressing the Data-Leashing problem.

Our proposed approach solves the control of *data control after sharing problem (Data-Leashing)* by encapsulating digital assets in a special container program that protects the digital assets (such as personal pictures). The specialized container program allows only authorized users to access protected digital assets. Should the receiver of the digital asset decide to (re)share the protected digital assets with unauthorized receivers, the container program notifies the assets owner and gives them the option to either grant or deny access to a *novel subject*. We use the term *novel subject* to refer to any new receiver of a shared digital asset whom the original owner of the digital asset did not authorize.

In essence, the container program turns *passive* file objects into *active* entities with inbuilt functionality to actively report on their status and security environment. One key factor that exacerbates the problem of controlling digital assets once they are shared is that digital files are passive objects that can be copied, manipulated, stored, and shared without much limitations of functionality built onto the digital objects themselves. This is the fundamental premise behind our motivation to turn passive objects into active self-protecting objects. Even if a receiver of protected digital assets downloads and saves a protected asset, the protection mechanism remains active. This allows the assets' owner to *command* the assets to perform a limited set of actions, such as *self-destruction*. We present a high-level architecture illustrating the key components of our approach.

The rest of the paper is structured as follows: Section II presents a motivating scenario to illustrate the key properties and aspects of the data post-sharing control problem. Existing approaches and their limitation are discussed in section III. The high-level architecture of the Data-Leashing approach and its key components is presented in section IV. We conclude the paper and discuss further work in section V.

2. Motivating Scenario

A graphic designer, Samar, has discovered that one of the designs she made for her client has been stolen and used without her permission. She used to send her designs to

2767-6684/23/$31.00 ©2023 IEEE
DOI 10.1109/DSA59317.2023.00042

customers for feedback and suggestions, allowing them to save her designs without her knowing. However, she couldn't prevent her clients from viewing their designs, as she constantly needed their feedback.

Her friend, Hana, recommended she use a new Data-Leashing method. Instead of sending her designs unprotected, she started sending her designs inside a program (a virtual machine) that could protect her work. The next time, Samar began testing this method when sending a logo she has designed for her new client.

Her client could download her design to access it offline, which was convenient for them. After they edited the logo, they decided to save a copy of it instead of having to pay Samar for her services. First, they tried saving it as a PNG, but they could not find a way to convert it to any format. Thus, they decided to choose a more straightforward option: take a screenshot of the logo, which did not work because the picture turned black when they tried it.

Since their attempt failed, they thought of taking pictures using their phone. However, the program was able to detect their phone's camera in front of the screen, so it hid the design again. Meanwhile, the Data-Leashing server kept reporting the client's suspicious behaviors to Samar. When the client tries calling someone to see what is happening, the camera detects another figure in the room, one of which is an unrecognized and authorized face. However, since Samar didn't see a threat in a client showing her designs to others, maybe to receive different opinions from friends or family, she decided to allow unrecognized faces to view her designs through the client's device.

Ultimately, her client could not figure out a way to save Samar's design, so they disconnected their computer from the internet, hoping it would help them find a way to steal the logo. However, that did not work, as Samar had already set the content's availability to the default 24-hour time limit. After time had passed, her client couldn't access the design anymore. Therefore, they had to either purchase Samar's plans or give up the idea of using her work without her permission. Overall, this data-leashing method could help Samar and many other digital creators preserve their rights without sacrificing customer satisfaction.

The problem presented in the scenarios has two main aspects. (1): Content owners can lose track of their data and whom it is being shared with by the receivers. (2): Content owners cannot control their data after being shared; even if they can detect that someone is using their data without permission, there is no technical solution they can use to prevent such unauthorized usage. The only recourse available is the legal route where a victim can seek redress from a court of law on the abuse of their data by an offender.

3. Problem Definition

Digital files shared on social media can be easily accessed, downloaded, and shared without the owners' knowledge or consent. That is because users cannot maintain full control over their shared files, which may lead to various privacy violations, cyberbullying, intellectual property theft, and other threats that could cause harm to individuals and organizations, making it a significant concern for users who share their data on digital platforms.

For example, when someone shares personal pictures on social media, they become more vulnerable to cyberattacks, such as identity theft or blackmail, which may cause severe financial and personal harm. While some platforms may allow content owners to deny access to their files being shared, these solutions are insufficient since users who receive these files can still download and store them on their device and may share them with others, making it difficult for owners to monitor their shared data and maintain control over it.

Therefore, there is a need for a novel solution that allows users to fully protect and control their data after sharing. The Data-Leashing problem has the following components: *Owner*, *Receiver*, *Physical Data Transfer*, *Electronic Data Transfer*, *Digital Asset*, and *Unauthorized Users*. The components are depicted in Figure 2 and explained in detail in the rest of this section.

Digital Asset (DA): Any digital information, media, or file stored, transmitted, or accessed electronically. Examples of digital assets can include documents, images, software, databases, websites, and more. Access control is important in order to protect these digital assets from unauthorized access or modification. For example, a company may have a database containing customer information that they want to protect from unauthorizzed access. The company can employ access control mechanisms such as authentication, authorization, and encryption to ensure that only authorized users can access the database.

Asset Owner (OW): The user with authority over a file and the right to control its distribution and access. The person, user, or entity that has legal rights over the data. This can include copyright holders, authors, publishers, and other persons and organizations with ownership rights over the distribution or access to the data. The owner may also be responsible for granting permission for others to access and use the data.

Authorized Recipient (AR): The person, user, or entity that receives the data from the owner or sender. This person or entity is responsible for using the data in accordance with the terms and conditions set by the owner. The recipient may also be required to agree to a data usage agreement or other legal agreement as part of the transfer process. In the context of the Data-Leashing problem, the receiver obtains access to the data with the data owner's permission. However, the receiver may misuse or share the data with unauthorized users without the owner's consent.

Sharing Medium: Refers to the means by which a digital asset is being shared. We identity three sharing mediums, namely: *electronic, physical*, and *screen view*:

- *Physical Data Transfer:* The process of transferring files from one device to another using physical media such as optical disks, USB drives, external hard drives, and more. This is usually done in order to quickly transfer large amounts of data between two devices that are not connected

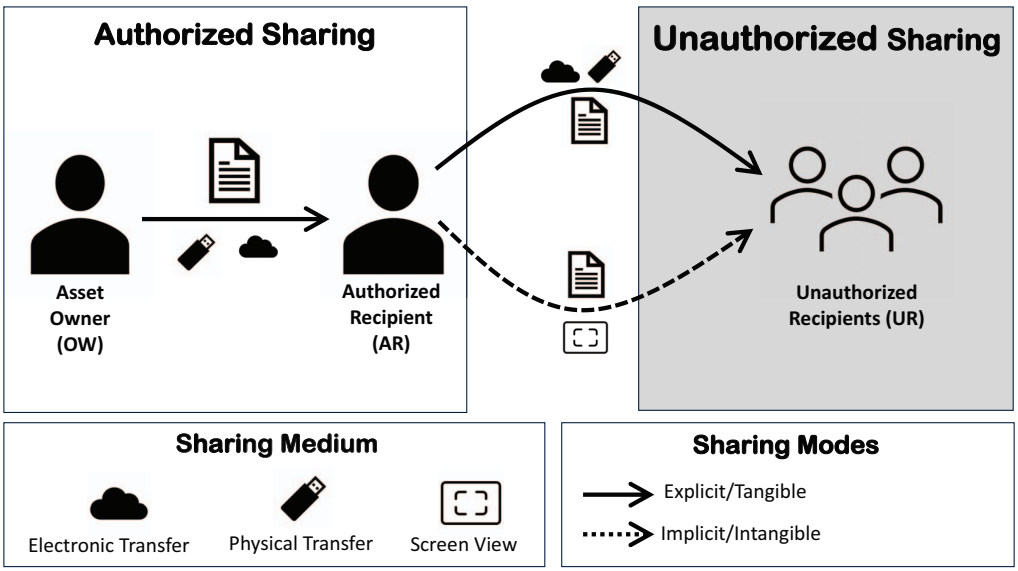

Figure 1. Depiction of the Components of the Data-Leashing Problem

via a network.

- *Electronic Data Transfer:* The process of sending digital files from one device to another. This can be done through an electronic communication network such as the internet, email, cloud storage systems, or through a direct connection between two devices. This is a common way of sending and receiving digital files such as documents, images, videos, and more.
- *Screen View*: This is when a digital asset is shared by showing an unauthorized recipient the display of the asset on a screen. In this case, although the authorized recipient has not been explicitly sent the digital asset, he has had access to it by virtue of having seen it on the display. Such sighting of the digital asset breaches it confidentiality if the owner intended for that the asset should be kept confidential.

Unauthorized Recipient (UR): Someone/user who has been explicitly or implicitly denied access to a system, resource, or service due to inadequate access control measures. They may not have the necessary credentials, privileges, or permissions to access the resource, or they may have been blocked explicitly by an administrator. For example, if a company has a secure database with sensitive customer information, an unauthorized user would be someone who does not have the necessary permission to access the database. The administrator may have blocked the user's access to the database to prevent any unauthorized access. An unauthorized user may further share the digital asset with other unauthorized users.

This issue also refers to a user who has been shared a confidential digital asset letting unauthorized users read the digital asset on the user's device. This entails indirect or implicit sharing of the assets. For example, a user A shared a confidential photo with another user B. Being the authorized user, B opens the photo and then allows unauthorized user C to view the photo on B's device. This breaches the confidentiality of the photo as user C is unauthorized to see it - it is meant for the eyes of user B only.

Sharing Modes: We distinguish between *explicit* and *implicit* sharing modes. In the explicit mode when the asset is shared there is evidence of such sharing. For example, if the asset has been shared by email there will be tangible evidence that it was sent to and received by the recipient. On the hand, with implicit sharing there is no physical evidence that such sharing of the digital asset took happened. For example, if the authorized recipient (AR) showed a sensitive photo received from the owner to an unauthorized subject (UR).

4. Key Security Requirements or Properties of an Ideal Solution to the Data-Leashing Problem

The key security requirements of a solution to the Data-Leashing problem are: *Remote Asset Access Control*, *Self-Destruction*, *Inseparability*, and *Indestructibility*. The requirements are explained in more detail below:

1) *Remote Asset Access Control*: The ability to control the asset even after it has been shared. This can be used to revoke access to a digital asset that has been shared without the permission of the owner. Alternatively, asset access control can be used to grant access to a previously unauthorized subject.

   To achieve effective access control, the protection mechanism should be able to detect and distinguish between authorized and unauthorized users' devices. Including a server that can connect the owner with the digital asset while keeping a record of authorized and non-authorized devices and can be utilized to implement a remote access control mechanism.

However, one limitation to this approach is when data is being accessed offline. In such scenarios, the program would refer to the pre-established permissions given by the owner, while maintaining the data's security.

2) *Inseparability and Indestructibility*: It should not be possible to destroy, separate, or isolate the digital asset from its protection mechanism. The binding of the digital asset to its protection mechanism should be a *one-way function* i.e. it should be easy to bind the two but not feasible to separate or unbind them, similar to the idea of a one-way hash function in cryptography. If such an attempt is made by an attacker, the asset should *self-destruct*. Isolation of digital assets from the protection mechanism will leave them vulnerable to attacks. Inseparability could be achieved by encoding the protected data within the carrier program using encryption algorithms.
3) *Self-Destruction*: Refers to the digital assets having the functionality to destroy themselves either on demand (upon instruction from its owner) or in case the protection mechanism might fail in fully protecting the data.
4) *For Your Eyes Only (FYEO)*: An authorized user may share the content of sensitive information by inviting unauthorized users to view the content on the authorized user's device, or by capturing using an external camera and sharing it with unauthorized users. This is a violation of the confidentiality of the information as it is viewed by unauthorized users. The protection mechanism should ensure the sensitive information is only viewed by the authorized user by detecting the presence of authorized users and blocking the viewing if such users are detected.

While these key requirements are fundamental, there are additional specifications that are highly preferred for an optimal solution, including: *Platform Independence*, *Privacy Preserving Asset Tracking*, *Transparency*, and *Portability*.

1) *Platform Independence*: The solution should ideally be platform-independent; that is, it should function consistently across all platforms (Email, WhatsApp, OS copy command, etc.) and without specific dependencies.
Platform independence can be achieved in various ways. One way to achieve it is by encoding the program in a mini *virtual machine (VM)*, a software that behaves as a separate environment and allows a program or an operating system to run inside another. Using a mini VM would be beneficial in providing independence as it will allow the program to run consistently across different operating systems. However, this approach might face challenges regarding compatibility and performance. If this option becomes inapplicable, an alternative approach to ensure platform independence would be creating a cross-platform. Eventually, both options would provide platform independence to the solution.
2) *Privacy-Preserving Tracking*: Even after a file has been downloaded, it should be possible for the owner of the file to be notified when it is being shared without violating the privacy of recipients, including unauthorized users.
Privacy-preserving tracking would be essential for the owner to be able to take appropriate actions towards any potential security risks. That is without violating the privacy of other users.
3) *Transparency*: This security requirement implies that the Data-Leashing mechanism while providing good protection to the digital asset, should not change the way that the digital asset is accessed/manipulated. This means that its presence in the digital asset should be transparent; that is, it should be as if it were not there.
4) *Portability*: It should be possible to open the digital asset (file) through the reader normally used for opening a file of that type. For example, if a photo in PNG format is protected through the data-leashing approach, viewing the file with its normal viewer should still be possible.

5. Related Work

This section discusses different existing methods that can be used in protecting shared digital content. Which includes *authorization, time-limited, screen-shooting prevention, facial authentication*, and *password-based authentication*.

5.1 Authorization Approach

Authorization is one of the most common methods used for controlling access to digital content on cloud-based platforms [1], [2], [3], [4], [5], [6], [7];as it enables the owner to share their files with many users simultaneously while keeping track of who can view and share the content of these files. In addition, it allows the owner to assign specific *privileges* to every user. The *privileges* are a set of actions that a user is allowed to perform on the resource such as the ability to read or write a file.
Some digital platforms that uses this approach are: Google Drive, Dropbox, and One Drive.
In the context of protecting a shared image, the effective implementation of an authorization approach relies on using appropriate authentication methods. Some common authentication methods include:

5.1.1 Face Authentication

Face authentication[8] is a technology that uses artificial intelligence (AI) to verify people's identities by recognizing their faces to give them access to different digital services/resources. Many technology companies, such as Apple, use this authentication method widely as a standard feature in their products. For example, FR allows owners of iPhones to unlock their devices using their Face ID, which eliminates the need to remember passwords. This technology can be helpful in our Data-Leashing method to prevent different types of attacks. For example, it could be used to prevent third-party members from accessing content through an authorized user's device.

5.1.2 Password-based Authentication

In this method, digital assets are protected by a program that encrypts their content and decrypts it only when a user types in the correct password. This method is found in many websites and applications such as Adobe Acrobat.[9], [10], [11] The main concern when using this method is that although the content itself might be protected, the password is not; because it can be hacked, leaked, or intentionally shared with unauthorized users. However, if the authentication process was based on a more reliable option that cannot be exchanged (such as a digital ID or a MAC address) this problems would be resolved.

5.2 CP-ABE-based Access

CP-ABE stands for *Ciphertext-Policy Attribute-Based Encryption*[12], [13], [14]. It is an encryption scheme where users can decrypt and encrypt data based on their specific sets of attributes. CP-ABE is widely used with sensitive data that requires access restrictions.

5.3 Temporary Access Approach

This approach is often seen on social media platforms such as Whatsapp, Snapchat, and Instagram. It gives the receiver access to view digital content for a limited time, or *one-time access*, which expires after opening a photo once. This approach is very straightforward and effective. However, it does not give the owner enough options to control the accessibility of their digital content, as it gives viewers full access within the allocated time, which does not protect content until time expires. In addition, this solution is only used to protect digital content online. However, it has more potential and can be a beneficial way of protecting offline content while giving owners more options to manage the accessibility of their data.

5.4 Screenshot prevention

Screenshot prevention is a good way of protecting visual content by hiding the content once the screenshot attempt has been detected [15]. The same concept usually applies to screen recording prevention methods [16]. This method is common in many social media and streaming platforms. One of the technologies used for this approach is the *Encrypted Media Extensions (EME)*, which was first used by Netflix. This method is very useful as it enables the user to view the digital content while it remains protected. However, it can only be a part of the solution since it only protects the digital content but cannot give the owner post-sharing control of the content.

5.5 Physical Attacks Prevention Methods

To ensure sensitive data are only accessible by authorized users, the solution should prevent any physical attacks where authorized or unauthorized recipients may capture shared data to save or share with others.

5.5.1 Camera recognition Approach

This approach relies on the program's ability to detect camera lenses in its sight using machine learning technologies, which have been implemented in similar solutions before [17] Once a camera has been detected, the program hides the protected image by either blurring or blocking the screen.
However, this method has many limitations and risks. For example, a well-hidden camera may be able to deceive the camera detection mechanism, making the data vulnerable to attacks. Instead, another possible approach would be using a passive foolproof distortion mechanism that will prevent cameras from capturing images.

5.5.2 Passive Camera deactivation Approaches

These approaches employ the differences between cameras and human eyes. They passively prevent any physical attacks by distorting images captured by cameras without affecting how human eyes would observe, which ensures images will remain protected.
One example of these approaches is the one used in Kaleido [18]. In this approach, the original video frames are re-encoded and subjected to chromatic frame decomposition, illuminance frame pollution, and spatial deformation effects, causing significant quality degradation imperceptible to human eyes. The method leverages the rolling-shutter effect found in most cameras (where the image is captured line by line sequentially) to disrupt the video recording process. While this technique may not be effective with high-end cameras equipped with global shutters, adopting a similar approach could benefit the Data-Leashing solution by stopping unauthorized video recording without needing additional hardware.

5.6 The Use of blockchains in Non-Fungible Tokens (NFTs)

NFT stands for *non-fungible tokens*, unique digital assets representing ownership over a non-replaceable item such as an image, video, or any other form of digital media. NFTs are built on blockchain[19], [20] technology, allowing them to be traded like physical assets while protecting them by proving a digital asset's ownership and authenticity.

5.7 Steganography

Steganography[21], [22] is a technique of hiding messages or information inside another. For example, it could be hiding a photo or a program inside an image by attaching it to the *EXIF* files. EXIF (Exchangeable Image File Format) are those files attached to images to carry important data about the image, for instance, the time when the photo was taken. These files could be compromised to execute code, a widely used method with self-executing viruses. However, it could also be used for ethical purposes, such as protecting data. There are different ways to use steganography to protect photos, including:

- *A photo inside a photo*[23], [24] In this method, we can hide the image we want to protect inside another by slightly changing the RGB values of specific pixels in a way that is not visible to the bare eye. Another way is by attaching the photo to be protected at the end of another photo's code. However, this solution cannot be independent as it requires a program to decrypt the data (photo) that has been encrypted. In addition, it cannot prevent screenshots and other possible attacks. Therefore, an alternative method to hide a self-executing program that could decrypt the hidden photo and protect it from different attacks is needed.
- *A program inside a photo* In this method, a code is embedded inside a picture allowing it to carry a self-executing program that could run independently as long as the usage is let to load. Nevertheless, this technique could lose effectiveness once the image is stopped from loading, as the program protecting the photo would not execute. Hence, the carried photo would be vulnerable to all kinds of attacks. Moreover, in both methods, a knowledgeable attacker can access the hidden pictures or code using *steganalysis*, the study of decrypting messages hidden using steganography.

5.8 Device Fingerprinting

Device fingerprinting[25], [26] is a process that involves collecting and analyzing various unique characteristics of a device. That is to create a *digital fingerprint* that serves as a distinct identifier for that device, similar to human fingerprints. These characteristics may include hardware or software configurations, MAC address, screen resolution, and other attributes specific to the device that would be hard to replicate.
Device fingerprinting is often used for various purposes. In the context of Data-Leashing, a program can determine whether it has been transferred from one device to the other by creating a unique device fingerprint using insensitive data obtained from that device. It can then apply consecutive checks to ensure the program carrying the digital asset remains on the same device. If the current fingerprint differs from the previously recorded one, the system can assume the program might have been shared or transferred. It then takes an appropriate action to keep the digital asset protected. To conclude, a device fingerprinting technique will equip the solution with privacy-preserving asset tracking and assure data safety.

6. The Data-Leashing Approach

The high-level architecture of the Data-Leashing approach and its key components is depicted in Figure 2. A Data-Leashing solution consists of a mini virtual machine, access control functions, access control data, an interface, a digital asset, and a server. These components of Data-Leashing are explained in the following subsections:

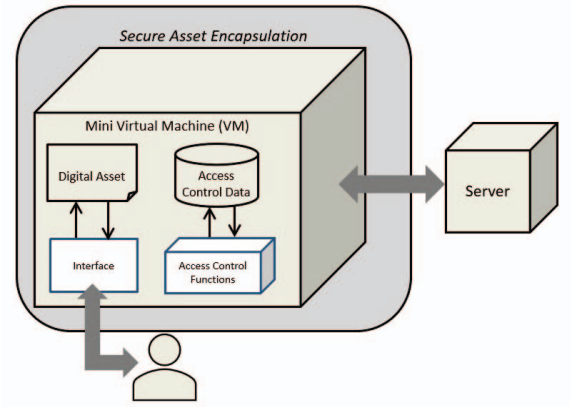

Figure 2. High-level architecture of the solution

6.1 Mini Virtual Machine

Using a virtual machine (VM) provides platform independence, meaning the Data-Leashing approach will work on any operating system.
This mini virtual machine (VM) contains the components needed for the *Data-Leashing approach*, including: *an interface* (2), a *digital asset* (3), *access control functions* (4), and *access control data* (5).

6.2 Interface

The *interface* will allow users to interact with the virtual machine and view its content while keeping it protected.

6.3 Digital Asset

The digital asset refers to the image or data to be protected.

6.4 Server

The *server* is a client application that connects the mini VM and the digital asset owner. Any communication between the owner and the mini VM will be done through the server. It will also provide the owner with a dashboard for monitoring and controlling their data. In addition, it will save a copy of the owner's information so they can access their dashboard from other devices.

6.5 Access Control Functions

Access Control Functions are a set of functions that to protects the digital asset by limiting the user's abilities to view, save or edit the digital asset. These functions are responsible for giving user's their privileges based on the roles assigned to them.
The access control functions component enforces strict access control over the hidden data (image) stored inside the mini VM. It employs encryption algorithms to conceal the data from users, ensuring users will not be able to directly access or view protected data unless it is explicitly displayed by the program.

The access control functions utilize a combination of MAC address and device fingerprinting to detect unauthorized transfers of the program. When the program is first installed, it records the device's MAC address and creates a unique fingerprint based on its attributes. During subsequent checks, it compares the current MAC address and fingerprint with the recorded ones. If either changes, the program infers that it has been moved to another device, triggering access restrictions where it immediately hides the image until the device reconnects to the server for further access verification.

In the access verification process, the program communicates with the server to confirm the device's authorization status from the access control data. It sends the device's identification details (i.e. MAC address and fingerprint) to the server. The server then compares this information with its records to decide whether this device has been authorized by the owner or not, and whether it should grant or deny access from it. The server then sends confirmation to the program, allowing it to display the image.

This process ensures that only authorized devices can be used to access protected data, whether it is online or offline. Devices need to connect to the server once for verification, which guarantees the digital asset will remain secure. Additionally, it allows users to conveniently share the data container (VM) with others, while waiting for owner's approval before giving access to new users.

Furthermore, as long as the image is on display, it remains secured by a mechanism similar to the one discussed in subsection 5.5.2.This protection ensures that unauthorized recording or capture attempts would result in degraded or unusable versions, preserving the image's security during its display.

In case the program fails to protect the digital asset for any reason, the Access Control Functions keeps the data safe by either hiding it or initiating a self-destruct mechanism to make sure the data will never be leaked.

7. Conclusion and Future Work

In this paper, we have proposed the idea of Data-Leashing, a novel approach to the problem of controlling data after it has been shared online. There is currently no method of protecting data after it has been shared. The Data-Leashing approach has the potential to solve this issue by providing a mini virtual machine (VM) that carries and protects its data content. This virtual machine works like a file viewer that is portable and platform-independent to ensure it can be shared easily across a variety of devices. It comes with a set of access control functions, access control data, and an interface connecting the content viewer with the mini VM. In addition, it is linked to a server that keeps it linked to the owner, where the VM can receive commands and send activity updates regularly. It also gives full protection over its content by *encapsulation* techniques similar to *Encrypted Media Extensions (EME)* that are used in *Digital Rights Management (DRM)* along with some camera deactivation technologies, while allowing viewers to edit and save data offline under the owner's authority as long as the digital assets are being protected inside the VM under the owner's control. In future work, we will implement the proposed architecture of the solution to evaluate its feasibility and effectiveness. Once the feasibility is confirmed, we will develop the proof-of-concept into a product and evaluate its practicality across different platforms.

References

[1] J. Li, Y. Ye, Y. Zhou, and J. Ma, "Codetracker: A lightweight approach to track and protect authorization codes in sms messages," *IEEE Access*, vol. 6, pp. 10 107–10 120, 2018.

[2] M. Joshi, K. P. Joshi, and T. Finin, "Delegated authorization framework for ehr services using attribute-based encryption," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 1612–1623, 2021.

[3] D. Patil and N. Mahajan, "An analytical survey for improving authentication levels in cloud computing," in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2021, pp. 6–8.

[4] N. A. Patel, "A survey on security techniques used for confidentiality in cloud computing," in *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, 2018, pp. 1–6.

[5] V. R and V. Lavanya, "An survey analysis of security issues in the cloud data storage," in *2022 8th International Conference on Smart Structures and Systems (ICSSS)*, 2022, pp. 1–8.

[6] P. J. Sun, "Privacy protection and data security in cloud computing: A survey, challenges, and solutions," *IEEE Access*, vol. 7, pp. 147 420–147 452, 2019.

[7] A. Ahadipour and M. Schanzenbach, "A survey on authorization in distributed systems: Information storage, data retrieval and trust evaluation," in *2017 IEEE Trustcom/BigDataSE/ICESS*, 2017, pp. 1016–1023.

[8] L. Li, X. Mu, S. Li, and H. Peng, "A review of face recognition technology," *IEEE Access*, vol. 8, pp. 139 110–139 120, 2020.

[9] A. Sharma, R. C. Belwal, V. Ojha, and G. Agarwal, "Password based authentication: Philosophical survey," in *2010 IEEE International Conference on Intelligent Computing and Intelligent Systems*, vol. 3, 2010, pp. 619–622.

[10] A. Abraheem, K. Bozed, and W. Eltarhouni, "Survey of various graphical password techniques and their schemes," in *2022 IEEE 2nd International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, 2022, pp. 105–110.

[11] C. Doğanay and A. Küpçü, "Comparative survey on single password authentication techniques," in *2020*

International Conference on Information Security and Cryptology (ISCTURKEY), 2020, pp. 5–10.

[12] C. Zhao, L. Xu, J. Li, H. Fang, and Y. Zhang, "Toward secure and privacy-preserving cloud data sharing: Online/offline multiauthority cp-abe with hidden policy," *IEEE Systems Journal*, vol. 16, no. 3, pp. 4804–4815, 2022.

[13] G. A. Thushara and S. M. S. Bhanu, "A survey on secured data sharing using ciphertext policy attribute based encryption in cloud," in *2021 8th International Conference on Smart Computing and Communications (ICSCC)*, 2021, pp. 170–177.

[14] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[15] T. Kawamura, T. Ebihara, N. Wakatsuki, and K. Zempo, "Eyedi: Graphical authentication scheme of estimating your encodable distorted images to prevent screenshot attacks," *IEEE Access*, vol. 10, pp. 2256–2268, 2022.

[16] B. V. V. R. Kumar, B. A. Vardhan, C. H. R. Gupta, and P. Surekha, "Reduction of movie piracy using an automated anti-piracy screen recording system: Anti-piracy screen recording system," in *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, 2019, pp. 301–304.

[17] K. Truong, S. Patel, J. Summet, and G. Abowd, "Preventing camera recording by designing a capture-resistant environment," 09 2005, pp. 73–86.

[18] L. Zhang, C. Bo, J. Hou, X.-Y. Li, Y. Wang, K. Liu, and Y. Liu, "Kaleido: You can watch it but cannot record it," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 372–385. [Online]. Available: https://doi.org/10.1145/2789168.2790106

[19] S. Balasubramanium, K. Sivasankar, and M. P. Rajasekaran, "A survey on data privacy and preservation using blockchain in healthcare organization," in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2021, pp. 956–962.

[20] T. Salman, M. Zolanvari, A. Erbad, R. Jain, and M. Samaka, "Security services using blockchains: A state of the art survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 858–880, 2019.

[21] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, "Image steganography: A review of the recent advances," *IEEE Access*, vol. 9, pp. 23 409–23 423, 2021.

[22] R. Anderson and F. Petitcolas, "On the limits of steganography," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 474–481, 1998.

[23] N. Kanwal, M. N. Asghar, M. S. Ansari, M. Fleury, B. Lee, M. Herbst, and Y. Qiao, "Preserving chain-of-evidence in surveillance videos for authentication and trust-enabled sharing," *IEEE Access*, vol. 8, pp. 153 413–153 424, 2020.

[24] M. Swanson, M. Kobayashi, and A. Tewfik, "Multimedia data-embedding and watermarking technologies," *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1064–1087, 1998.

[25] K. Takeda, "User identification and tracking with online device fingerprints fusion," in *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*, 2012, pp. 163–167.

[26] D. Wei, Y. Gu, and Y. Du, "Mobile device fingerprinting recognition using insensitive information," in *2022 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML)*, 2022, pp. 1–6.