

**Technical Report
1210**

Charting a Security Landscape in the Clouds: Data Protection and Collaboration in Cloud Storage

G. Itkis
B.H. Kaiser
J.E. Coll
W.W. Smith
R.K. Cunningham

7 July 2016

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



This material is based on work supported by the Department of Homeland Security
under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001.

Approved for public release: distribution unlimited.

This report is the result of studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology. This material is based on work supported by the Department of Homeland Security under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Department of Homeland Security.

© 2016 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Delivered to the U.S. Government with Unlimited Rights, as defined in DFARS Part 252.227-7013 or 7014 (Feb 2014). Notwithstanding any copyright notice, U.S. Government rights in this work are defined by DFARS 252.227-7013 or DFARS 252.227-7014 as detailed above. Use of this work other than as specifically authorized by the U.S. Government may violate any copyrights that exist in this work.

Massachusetts Institute of Technology
Lincoln Laboratory

Charting a Security Landscape in the Clouds: Data Protection and
Collaboration in Cloud Storage

G. Itkis
B. Kaiser
J. Coll
W. Smith
R. Cunningham
Group 53

Technical Report 1210

7 July 2016

Approved for public release: distribution unlimited.

Lexington

Massachusetts

This page intentionally left blank.

EXECUTIVE SUMMARY

This report surveys different approaches to securely storing and sharing data in the cloud based on traditional notions of security: confidentiality, integrity, and availability, with the main focus on confidentiality. An appendix discusses the related notion of how users can securely authenticate to cloud providers.

We propose a metric for comparing secure storage approaches based on their *residual vulnerabilities*: attack surfaces against which an approach cannot protect. Our categorization therefore ranks approaches from the weakest (the most residual vulnerabilities) to the strongest (the fewest residual vulnerabilities). In addition to the security provided by each approach, we also consider their inherent costs and limitations. This report can therefore help an organization select a cloud data protection approach that satisfies their enterprise infrastructure, security specifications, and functionality requirements.

This page intentionally left blank.

TABLE OF CONTENTS

	Page
EXECUTIVE SUMMARY	iii
List of Illustrations	vii
List of Tables	ix
1. INTRODUCTION	1
1.1 Scope	2
1.2 Categorization Criteria	4
2. PRIMARY CATEGORIZATION: CONFIDENTIALITY	7
2.1 Category CO-1: No Protection	8
2.2 Category CO-2: Guarded Gate	8
2.3 Category CO-3: Combined Storage & Security Provider	10
2.4 Category CO-4: Client-side Encryption, Outsourced Security Provider	11
2.5 Category CO-5: Client-side DIY Security Provider	12
2.6 Category CO-6: Client-side Automated Security Provider	14
2.7 Performance and Storage Costs	15
2.8 Categorization and Attack Vectors Summary	16
3. INTEGRITY	21
4. AVAILABILITY AND DENIAL-OF-SERVICE	23
5. CONCLUSION	25
APPENDIX A	
AUTHENTICATION	27
Glossary	33
References	35

This page intentionally left blank.

LIST OF ILLUSTRATIONS

Figure No.		Page
1	Possible attack vectors for an adversary in the cloud storage setting.	5
2	CO-1: No protection.	8
3	Probability of system compromise within one year.	8
4	CO-2: Guarded gate.	8
5	CO-3: Combined storage and security provider.	10
6	CO-4: Client-side encryption with an outsourced security provider.	11
7	CO-5: Client-side DIY security provider.	12
8	CO-6: Client-side automated security provider.	14
9	Storage space saved by data deduplication.	16
10	A single provider duplicates (or encodes) files across multiple storage servers.	23
11	Files are duplicated (or encoded) and stored across multiple providers.	24
12	Files are duplicated (or encoded) and stored across multiple providers in a way that is transparent to the user.	24
13	Authentication with a verifier. The verifier checks that the user knows a particular secret; if they do, the verifier (acting here as a reference monitor) grants the user access to the resources.	27
14	Authentication without a verifier. The user locally derives a key which can be used to directly access cryptographically protected resources.	28

This page intentionally left blank.

LIST OF TABLES

Table No.		Page
1	A Summary of the Security Features Offered by Each Category	18
2	A Summary of How Each Major Attack Vector Discussed in This Report is Addressed by Each Approach	19
3	Authentication Modalities	29
4	Authentication Categories. In this context, “computational” protection refers to the entropy of the information in question. Low entropy information is easier for an attacker to compromise; high entropy information is more difficult.	31

This page intentionally left blank.

1. INTRODUCTION

The advent of cloud computing is perhaps the most revolutionary force in the information technology industry today. This field encompasses many different domains, including data storage, data processing, and services for the full stack of enterprise computing needs including infrastructure, platforms, and applications.

According to industry analyst firm International Data Corporation (IDC), the cloud accounted for one-third of global IT infrastructure spending in 2015, and that share is expected to increase to 45 percent in the next five years [1].

The U.S. government, and DoD specifically, are also moving to the cloud to reduce costs and increase flexibility. Many commercial cloud service providers have gone through the Federal Risk and Authorization Program (FedRAMP) [2] accreditation process and have products approved for government use. In addition, government organizations, such as Defense Information Systems Agency, are deploying clouds for use by their government customers.

Ensuring the confidentiality, integrity, and availability of code and data that are located on hardware owned and managed by a potentially untrusted third party is a significant challenge. This is especially true for organizations such as DoD that handle extremely sensitive information; there is no margin for error. These problems are complex; there are solutions that address some of them, but there are also some open problems which are subject to a great deal of ongoing research.

One characteristic shared by all cloud computing tasks is that they involve storing data in the cloud. In this report, we therefore aim to describe and rank the different approaches that are used in practice to securely store and share data in the cloud.

To further focus the scope of this report, we will provide a motivating use case. An organization comprised of many, potentially geographically disparate users wishes to allow those users to store, share, and collaborate¹ on files using the cloud. The owner of a file—that is, the user who created it in the cloud—must be able to specify who has access to that file, and preferably they should be able to specify independent access protections of different types (e.g., read vs. write).

This report surveys and categorizes approaches to the problem described in this section. Specific companies and products are mentioned as examples of the application of these approaches, but this report

¹Collaboration in this setting refers to the simple scenario in which multiple users are able to read and write to the same file. In the rest of this report, such simple collaboration will be understood as a part of sharing and not mentioned explicitly.

is not intended as a comprehensive survey of cloud data protection vendors. Information regarding individual products was obtained from publicly available white papers and articles, as well as conversations with some vendors (we are very grateful to those vendors who kindly provided us with information).

Included with the report is an appendix that overviews the related notion of how users can authenticate to cloud storage services—an important aspect of cloud security.

1.1 SCOPE

The main information security property we address in this report is *data confidentiality*, a property that ensures that only authorized users have access to the contents of data (i.e., the data in unencrypted, cleartext form). While there are a variety of approaches to protect confidentiality, most are based on two techniques. The first is a *reference monitor*, which is a software guard that verifies users' authorizations and mediates access to data. The second is cryptography; specifically, *data encryption*, which ensures data confidentiality against any parties who do not possess the requisite cryptographic key. Different approaches apply these two techniques separately or together in different configurations.

Cryptography in particular can be challenging to apply correctly, and the largest challenge of using cryptography is usually *key management*. When secure sharing is not supported, this task becomes trivial—each user can use a single key² to protect all of their data. For this reason, in this report we focus on secure storage approaches that enable secure sharing, and ignore the sometimes stronger protections available for non-shared data.³

The primary focus of this report is data confidentiality, but we will also briefly address data *integrity*, which is the property that ensures that an unauthorized party has not modified data, and *authenticity*, which allow users to verify that data truly originates from where it claims to originate. The two properties are essentially the same—data modified by an unauthorized party now comes from that

² A common practice of deriving this key from the user password—typically using functions such as PBKDF2—is often insecure. Essentially, it presents attackers with a problem fundamentally similar to that of discovering passwords *after* the password file (containing hashes of the passwords) has been stolen. Functions such as PBKDF2 provide some improvement in security by increasing the effort required for a successful attack, but the cloud itself provides the attackers with a powerful tool to defeat these defenses.

³ It is of course possible that several users share a single account, thus providing a kind of sharing, but we do not consider this here, as this approach simply punts the question of how the users share the account's key. Furthermore, ideally sharing should provide dynamic and flexible access controls that would enable revocation of permissions and support of security beyond confidentiality protection.

party, rather than the original source, and an impersonator can be viewed as an unauthorized party modifying the data. So, in this report, the term “data integrity” will be used to refer to both of these properties.

Data *availability*, a property that guarantees the ability of authorized users to access data, is our third consideration. Although it is impossible to prevent the loss of data stored in the cloud if the Internet is unavailable, or if the cloud destroys the data, there are techniques that help mitigate threats to availability. For example, it is possible to enable data retrieval over unreliable connections or in situations where a subset of cloud providers is unavailable or corrupted.⁴ Similarly, for data integrity, we cannot prevent the cloud service provider(s) storing the data from modifying it. Therefore, the best we can hope for is detection of data corruption and the application of techniques to make data corruption more difficult for attackers.

These three data security properties—confidentiality, integrity and availability (often abbreviated as CIA)—are the most commonly considered. However, these are not the only security properties one might desire. For example, confidentiality of not just the data but of the data *access patterns* is extremely important in some cases. Different privacy considerations may impose various information flow restrictions. For example, it might be desired that while in general, no one should be able to find out who made the last change to the document (only that the change was authorized), in some cases, the identity of the last author might need to be discoverable (with non-repudiation). These and other more advanced security features are outside the scope of this report, and in general represent future research directions rather than features of existing commercial tools.

Also out of scope for this report are the following notions: analysis of the cryptographic strength of specific cryptographic primitives used such as Advanced Encryption Standard (AES); protection of keys and key materials beyond the protocol level, e.g., fulfillment of Federal Information Processing Standards (FIPS) 140 requirements; protection of data-in-use on the client device; and satisfaction of FedRAMP requirements. Data-in-transit protections are also not considered, since Transport Layer Security/Security Socket Layer (TLS/SSL) can be trivially applied to any of the approaches covered in this report, and for some approaches data-at-rest protection also protects data in motion. Finally, all approaches rely on some form of identity management or public key infrastructure (PKI). As the implementation of this service affects all approaches essentially in the same way, it is not considered in this report.

For a thorough analysis of some of the notions mentioned in the previous paragraph, the authors recommend the 2012 Security as a Service (SecAAS) Encryption Implementation Guide [3] published by the Cloud Security Alliance (CSA). In addition to examining some similar ideas to those discussed in this report, the CSA guide covers algorithm strength analysis, key material protection (in accordance with FIPS requirements), public key infrastructure configuration, and other general standards and guidelines.

⁴ For more information, see the Availability and Denial-of-service section.

1.2 CATEGORIZATION CRITERIA

To categorize cloud data protection approaches, we believe it is important to first take a bird's-eye view and abstract away trivial technical details like choice of algorithms, modes, or key size. Claiming that one protection scheme is stronger than another because it uses Advanced Encryption Standard (AES) with a 256-bit key instead of a 128-bit key for example, is not a particularly insightful observation. Rather, this report seeks to provide thoughtful analysis of each approach by determining what assumptions it relies on and what attack surfaces it exposes. This information is particularly useful because it enables the determination of what types of adversaries each approach can protect against.

Each approach to data protection defends against certain attack vectors while leaving others undefended. Undefended attack vectors are called *residual vulnerabilities*. It is important to recognize that *any* approach will have some residual vulnerabilities and to be aware of these vulnerabilities, carefully evaluate the likelihood and impact of attackers exploiting them⁵, and identify potential mitigation techniques.

Specifically, Figure 1 shows the general attack vectors available to an adversary in the cloud storage setting. In the diagram, the *security provider* encapsulates whatever mechanisms are used to protect data stored in the cloud and enforce the access control policies defined by the data owners. Typically it includes authentication and authorization mechanisms.

An attacker can attempt to compromise data at the endpoints by attacking either *data in use* on the client device (AT-1) or *data at rest* on the cloud storage server (AT-2). They can also attack *data in transit* en route between the two, but as mentioned in the introduction, this attack vector is out of scope for this report. An attacker can also attempt to *subvert* some mechanisms by forcing them to deviate from their proper behavior or by changing their output. For example, if a cloud storage provider uses disk encryption to protect data at rest, an attacker may subvert the cloud storage service itself and gain access to the data that way (AT-3). Finally, an attacker may try to subvert the security provider itself (AT-4). This is a broad category of attack vectors that includes attacks on authentication and authorization mechanisms and their interfaces. Clearly, the security of an approach depends significantly on where and how the security provider is implemented; this notion is at the root of how many of the approaches discussed below differ.

⁵ Estimating the likelihood and impact of exploits is a notoriously difficult task, as one is prone to overlooking unexpected attack vectors and some implications and unanticipated consequences of the compromises. However, explicitly tackling such tasks is much better than leaving them unaddressed or handwaving them away until the disaster hits.

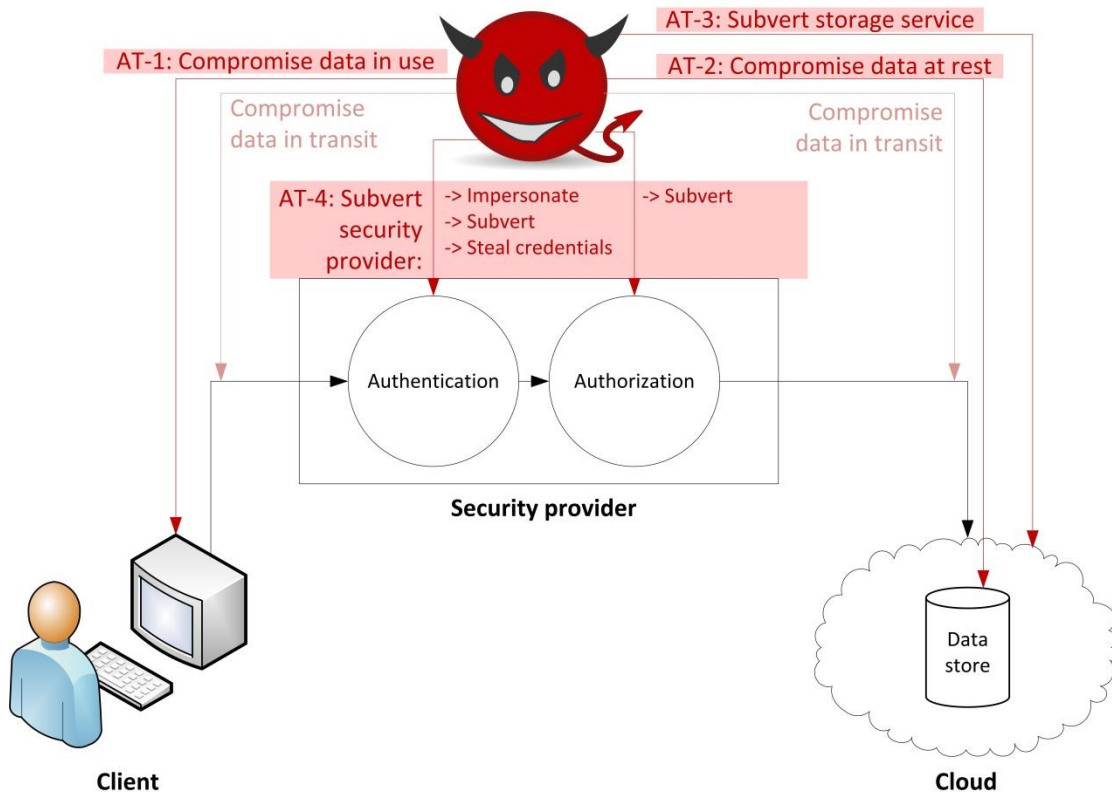


Figure 1. Possible attack vectors for an adversary in the cloud storage setting.

Specifically, this report will categorize cloud data protection approaches based primarily on which of the attack vectors they can protect against and which they leave as residual vulnerabilities. This categorization is discussed in Section 2, and a summary can be found in Section 2.8. Schemes that minimize residual vulnerabilities can generally be thought of as stronger or more secure. By this metric, ideal approaches would only expose data unprotected when absolutely necessary, i.e., while it is being created or consumed by the user. This is necessary because protection usually gets in the way of data consumption—e.g., data must be decrypted before it can be displayed for the user to view⁶.

⁶ Some system security approaches, such as memory encryption or other operating system capabilities, do attempt to protect data being consumed by the user. However, this attack vector is out of scope for this survey as it is inherently vulnerable to a sufficiently motivated attacker.

With that in mind, the fundamental questions that this report seeks to answer about each approach are:

- How is data protected at rest?
- How are keys managed?
- How is sharing supported?
- How do the above impact the attack surfaces available to adversaries?

2. PRIMARY CATEGORIZATION: CONFIDENTIALITY

Cloud data protection approaches are organized in this section from weakest security guarantees to strongest. The weaker approaches have more residual vulnerabilities and a wider attack surface; in the stronger approaches, these drawbacks are minimized. Our categorization represents the full spectrum of data protection: categories CO-1 and CO-2 store data without cryptographic protection, leaving stored data vulnerable to an attacker who can find it; in category CO-3 cryptographic protection (e.g., encryption) is applied to the data by the cloud storage provider before storage; categories CO-4 through CO-6 apply cryptographic protection to data *client-side* (i.e., on the client device), before it is sent to the cloud, providing data protection everywhere except the client device.

Pre-cloud security paradigms generally involve *centralizing* security provisions; in the cloud setting, such approaches are covered by categories CO-1 through CO-4 and are very popular in practice. There are some benefits to designing a system in this way: centralized system administration, user monitoring, and other useful tools are enabled. However, these approaches inherently suffer from having a *single point of failure*, the consequences of which have been made evident by the global scale of security failures when they occur. Examples abound, from the Wikileaks publications to the recent OPM breaches in the U.S. government to the millions of accounts compromised in various commercial organizations. The approaches described in categories CO-5 and CO-6 enable infrastructures with much finer-grained compartmentalization, which could potentially drastically reduce the scale of the impact of security failures when they do occur.

Each category is accompanied by a simple diagram depicting the principle parties. As data-in-transit protections are not considered in this report, they are ignored in the diagrams. The data transferred between the user and the cloud can be thought of as being Transport Layer Security/Security Socket Layer (TLS/SSL) protected whether they have a lock icon or not. That icon indicates client-side protection (that is, whether the content is protected *at the endpoint* of the connection) and will be explained when used.

Two particular residual vulnerabilities deserve a brief mention here. First, observe that in order for a user to create or consume a file, the file must be visible to them in cleartext. This means that no matter what protections are applied to data, the client device can be thought of as the minimal unavoidable exposure surface for unencrypted data. This attack vector is referred to as a *data in use compromise* in this report. Second, there is a class of attack that cannot be prevented through purely cryptographic methods: *denial-of-service attacks*. Because data is stored outside the control of the enterprise or user, an adversary accessing or controlling the cloud can destroy or corrupt it. It may be possible to mitigate this threat through other, non-cryptographic techniques; we mention these briefly in the Availability section, but they are not the focus of this report.

2.1 CATEGORY CO-1: NO PROTECTION

In a secure, closed environment such as an office LAN, home network, or disconnected military base, an enterprise may decide that it does not require data protection in addition to, say, physical protections such as armed guards at the gates. In this case, all attack vectors are left available to adversaries. The only reason to use this approach is if the organization is willing to accept the significant risks implied. Here, data at rest is unprotected, there are no keys to manage, and sharing is enabled trivially and without restrictions because data is accessible in cleartext to all users.

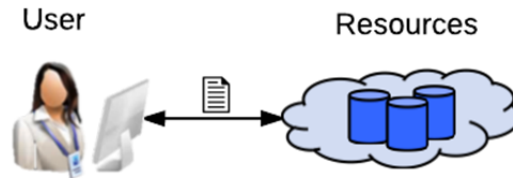


Figure 2. CO-1: No protection.

The weakness of this approach is obvious: it fails to maintain a basic tenet of security known as the *least privilege principle*. In the context of cloud storage, this doctrine advises that a given user should only be able to access data necessary for their legitimate purpose. In this scheme, any user with access to the system can access any piece of data; there is no enforcement of the least privilege principle. This opens up a variety of attack vectors; in particular, it makes possible the trivial compromise of data confidentiality by a malicious insider.

This approach also does not scale well—as the closed environment grows, the probability of compromise increases exponentially. For example, as shown in Figure 3, if each component of a system has a 0.1% chance of being compromised per month, a system with 1000 components has a 99.999% chance of being compromised within one year.

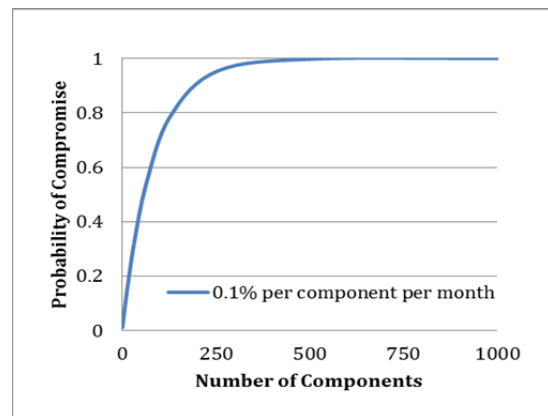


Figure 3. Probability of system compromise within one year.

2.2 CATEGORY CO-2: GUARDED GATE

The addition of a *reference monitor* mechanism to the unsecured environment defined above allows for the enforcement of access control policies. A reference monitor enforces access controls by intercepting and validating all user requests according to a predefined policy. According to its original conception [4], a reference monitor must have three properties: it must be tamper proof, impossible to bypass,



Figure 4. CO-2: Guarded gate.

and small enough to test for correctness. In modern environments, fulfilling each of these requirements poses a serious challenge.

In this model, when a user wishes to access a resource, they send the request to the monitor instead of directly to the storage server. The monitor then determines whether the user has the permissions required to execute the requested operation. If the request is accepted, it is executed on behalf of the user; otherwise, it is terminated.

This model maintains the benefits of the “no protection” option—no keys to manage, easy sharing—while allowing for the administration of user authentication and access controls. Data protection—including data confidentiality and integrity—is enforced by the reference monitor. This is convenient but extremely risky, because if the monitor is subverted or circumvented, data security is completely compromised. And because data is unprotected at rest, some risks from the prior model remain, such as the vulnerability to storage server administrators. Additionally, attackers who can subvert or circumvent the monitor can gain full access to data on disk.

In practice, the reference monitor can take several forms. It could be a gateway inside the enterprise network, a proxy server in between the enterprise network and an external cloud storage service, or a gatekeeper service running directly on the storage servers. Data-in-transit protections aside, these settings are functionally very similar when data is unencrypted at rest, so they have been combined into one category. However, it is worthwhile to understand the distinction between internal monitors, which operate inside of the enterprise firewall, and external monitors, which do not.

Due in particular to the attack vector available to administrators of the storage server, this approach is not well suited for use cases that require protecting data from storage service providers. Products that do provide a guarded gate are typically designed to manage *on-premises* storage services, although this does not alleviate other threats such as the risk of insider threats or Trojans. Acronis Access Advanced [5] is an example of such a product: the storage server and reference monitor all reside inside of the enterprise firewall, and users can conveniently access and share data from a variety of clients without concern for encryption or key management.

That being said, some vendors do protect data stored on an external cloud using a guarded gate. SpiderOak [6], a vendor that operates its own hardware and data centers for storage, can protect data stored on its servers with a strong protection scheme that prevents anyone but the data owner from decrypting the data (see category CO-5). However, this protection method is only used for data that is *not shared* with other users. To enable sharing with other SpiderOak users or external parties, the data is stored unencrypted and made available via a reference monitor-protected URL.⁷ Users who wish to access

⁷ On 1/28/2016, SpiderOak issued a press release announcing a forthcoming tool for “team collaboration privacy” [50]. However, it is not included in this report as technical details were not available at the time of this writing.

the data are provided with the URL and a password. This offers convenience, but sacrifices security. IDrive [7] and nCryptedCloud [8] are two other vendors that utilize this protection approach in order to enable file sharing.

2.3 CATEGORY CO-3: COMBINED STORAGE & SECURITY PROVIDER

Next, we will introduce cryptographic protection into the model. As discussed in the introduction, we will focus on providing confidentiality via encryption both for the sake of simplicity and because it is the most common cryptographic tool used. The simplest way to apply encryption to data and manage the associated encryption keys is to outsource these functionalities to the reference monitor. In addition to handling authentication and access control as is typical of reference monitors, the centralized service used in this model maintains encryption keys for all data stored on the server. This allows it to perform encryption and decryption operations transparently to the user. Centralizing these operations also enables convenient, powerful administrator oversight over features such as logging, auditing, and access management.

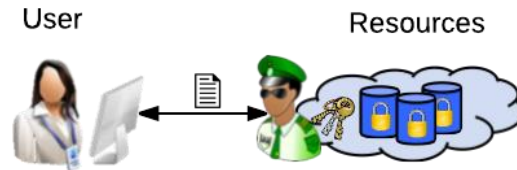


Figure 5. CO-3: Combined storage and security provider.

In contrast to the previous model, this approach offers protection from malicious administrators of the storage servers, and because data is encrypted it also prevents other adversaries who may bypass the reference monitor from viewing data in clear. Protection still relies on the monitor, but the introduction of cryptography enables protection of the stored data against an adversary who is able to access the storage directly (e.g., without going through the monitor). However, the effectiveness of this protection depends on the protection of the corresponding keys. If the keys are stored on the same disks as the data or are otherwise accessible by attackers, then little is achieved by this approach.

Since the monitor views all data in cleartext during cryptographic operations and manages all of the keys, it represents a significant residual vulnerability: a *single point of failure* for the whole system. The compromise or coercion of the monitor would fully break data confidentiality and integrity. This threat is magnified when the reference monitor is external—that is, outside of the enterprise network—but it is still present even when the monitor is internal and managed by enterprise-trusted administrators.

This single point of failure is exploitable not only by the organization controlling the monitor but also by other groups that have sufficient leverage over the organization. These groups could include government agencies of the country where the system is hosted, or criminal enterprises. The bottom line is that when the provider holds both the encrypted data and the keys, they can be compelled to decrypt and provide user data, and they may also be prevented from notifying the user that their data was accessed. If the user holds onto their own keys, they could still be compelled to decrypt their data, but they will at least know that their information has been accessed.

Many cloud data protection products adhere to this scheme, with DropBox [9] being perhaps the canonical example. Other products that provide centralized encryption and key management include Barracuda CudaDrive [10], BlueCoat Data Protection Gateway [11], Citrix ShareFile [12], Cleversafe [12]⁸, CTERA [13] [14], Egnyte Enterprise File Sharing [16], Google Drive [17], Nasuni [18], Panzura [19], Sookasa CASB [20], Sophos SafeGuard [21], SugarSync [22], and Vaultive [23].

Some vendors provide stronger, client-based encryption schemes as described in following categories but also offer centralized encryption and key management as a more convenient option. The best example is Amazon Web Services' Simple Storage Service [24], which offers automatic, provider-handled encryption but also includes an optional encryption tool in its software development kit (SDK) that allows customers to manage their own keys and perform their own encryption (as in category CO-5).

2.4 CATEGORY CO-4: CLIENT-SIDE ENCRYPTION, OUTSOURCED SECURITY PROVIDER

In the field of data security, separation of concerns and design modularity are extremely important doctrines. Adhering to these principles helps with prevention of security breaches and their containment. The next iteration of our cloud data protection model applies these notions by reducing the responsibility of the storage server to a single function—data storage—while moving encryption to the client device and security provisions to a third party. This party is a *security provider* that authenticates users and distributes authorizations that allow them to access data. Typically, the security provider is implemented as a key manager service that generates and provisions keys and provides a centralized interface for access management enforcement and oversight.

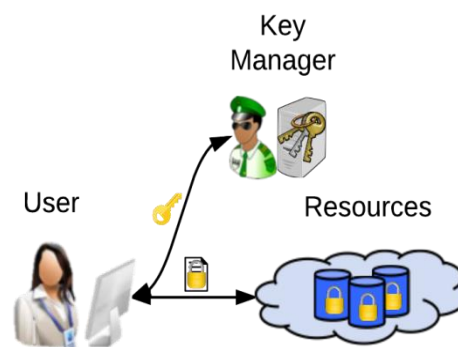


Figure 6. CO-4: Client-side encryption with an outsourced security provider.

⁸ Cleversafe uses encryption to implement an *all-or-nothing transform*, which splits data across multiple storage servers and ensures that an adversary who captures only some pieces of the data learns nothing. This technique packages the keys alongside the encrypted data, trading the challenge of key management for the challenge of assuring that authorized users are able to obtain sufficiently many pieces of the data while attackers or unauthorized users cannot. So while Cleversafe does not provide key management as other vendors in this category do, it does enforce its access controls (and authentication) centrally using reference monitors, so we believe it is a close (albeit not perfect) fit.

In this model, a user who wishes to access an object stored in the cloud first authenticates to the key manager, which then determines whether the user has a permission corresponding to the object they wish to access (and the access type they seek). If the user is authorized, they are provided with the necessary keys. To store an object in the cloud, the user again requests keys from the key manager and applies encryption and integrity protections locally before storing.

Note that in Figure 5, the file now shows a lock icon to demonstrate that it is protected by the user's key prior to transfer. No one, not even system administrators at the storage server, can access the protected contents. This is not to say that previous models transferred data in clear; it would likely be TLS/SSL-protected in transit. However, TLS/SSL does not prevent the recipient of the file from reading the file in cleartext.

The key manager can be implemented as a hardware or virtual software appliance that operates inside of the enterprise network or as an external, third-party service such as Amazon Web Services Key Management Service, which generates and stores keys using offsite hardware security modules (HSMs). For obvious reasons, it should not be collocated with the storage server.

The most substantial security benefit provided by this model is the relocation of encryption and decryption operations to the client device. This means that data is never visible in cleartext to any party other than the client device where the data is created or consumed. In order to compromise data confidentiality, an intruder must either corrupt the client or steal both the encrypted data from the storage server and the pertinent decryption key(s) from the key manager. The difficulty of this task demonstrates the value of adhering to the principle of separation of concerns. That being said, an insider with access to the reference monitor still has the ability to modify permissions and access keys. The danger of the single point of failure has been relocated but not ameliorated.

Products in this category include Accellion Kitemworks [25], Box [26] [27], Bitcasa Turnkey Drive [28], Credeon Cloud Data Protection [29], TitanFile [30], Viivo [31], and Vormetric [32]. These products typically provide client applications that encrypt data in preparation for storage on a public, commercial cloud. All users connect to a centralized server to retrieve their keys, then upload their encrypted files either to a commercial cloud file storage service such as DropBox or to servers managed by the provider.

2.5 CATEGORY CO-5: CLIENT-SIDE DIY SECURITY PROVIDER

As mentioned in the introduction, the client device can be thought of as the minimal required exposure surface for unencrypted data. Therefore, as we seek to minimize residual vulnerabilities, the natural next step is to migrate as much functionality as possible to client devices. This model does just that by removing the key management service and reference monitor and

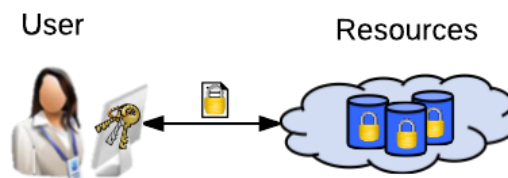


Figure 7. CO-5: Client-side DIY security provider.

forcing users to manage their own cryptographic keys—hence the name “do-it-yourself key management”. Here, each client device can be thought of as implementing an individual security provider that provides protections for just that user’s data.

Eliminating the centralized security provider that is used in the previous categories removes the need for *centralized* system administration. Instead of empowering a single party with absolute administrative authority, an enterprise can employ a distributed and compartmentalized infrastructure in which each user is responsible for the data they own.

This compartmentalization helps to contain the effects of security breaches; if one provider is compromised, only the corresponding user’s data is exposed. There is no reference monitor to corrupt and no centralized key store to attack. No party outside of the client, not even the cloud storage service provider, has the ability to decrypt files.

While the benefits of eliminating these centralized parties are substantial, these parties exist in other schemes for good reasons; they provide many convenient features. In this model, a user that wishes to share a file must manually transmit decryption and verification keys to the recipient of the permission. And if a user loses their decryption key, their data is unrecoverable unless it was shared with another user or an escrow agent.

Because of these complications, this protection scheme is not particularly user-friendly, but it still offers better security and has applications, particularly in cases where data security is the highest priority or when sharing is not a required feature.

The scheme can be implemented in one of two ways. One option is for the users of the enterprise to manually generate and store their keys locally. This is required by Amazon Web Services Simple Storage Service configurations that utilize Amazon’s Encryption Client. The other option is for the cloud data protection provider to store the keys in an encrypted form (typically using a password). When a user needs their keys, they can request them on-demand from this key service and decrypt them locally. This methodology is espoused by SpiderOak.

Other products that offer manually managed keying options include ElephantDrive [33], IDrive, and OpenDrive [34]. It is worth noting that the few vendors that offer these services typically make them available as an option alongside other forms of cloud data protection. Often they do not highly publicize their manual key management offerings due to the loss of functionality entailed.

Outside of these products, using a disk encryption service such as BitLocker [35] to encrypt files before uploading them to cloud storage provides comparable security guarantees (and functionality limitations).

2.6 CATEGORY CO-6: CLIENT-SIDE AUTOMATED SECURITY PROVIDER

The final class of cloud data protection techniques provides the functionalities typically enabled by automated key management techniques with the benefits of the client-side security provider. Here, sharing is implemented through the use of cryptographic permissions that are robust and self-enforcing. No reference monitor or other trusted third party is necessary for sharing because protection is provided by cryptographic hardness assumptions rather than by a reference monitor.

In this model, when a user wishes to access a file, they first check whether there is a cryptographic permission for them associated with that file. For read access, this permission is simple to implement. A symmetric content key encrypts the file, then that content key is encrypted using the user's public key and attached to the encrypted file as a permission. Exercising this permission does not require a reference monitor: if your private key decrypts a permission, then you have access; otherwise, you do not. For an adversary to view a file, they must break the encryption—there is no way for them to illicitly gain a permission or access the file contents without a permission.

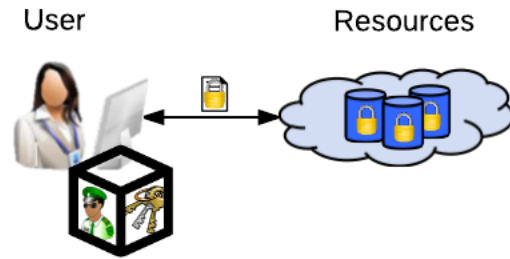


Figure 8. CO-6: Client-side automated security provider.

Users are responsible for storing their own private keys; this can be done via a variety of mechanisms including smartcards or operating system features such as OS X's Keychain. Content keys are never stored anywhere unencrypted. Therefore, the client devices are the only location in which files are unprotected and susceptible to theft and/or corruption. This is essentially the minimum possible attack surface—the data must be visible in clear to the client since that is where the user creates or consumes it. But no other parties in this scheme have the ability to access private keys or see files in cleartext. Because clients are the only fully trusted parties, this scheme eschews centralized management of any sort. By design, system administrators cannot view, add, or revoke permissions. And if a user loses their keys, they cannot be restored and their data may be unrecoverable⁹.

As in CO-5, eliminating this centralized management removes the need for centralized system administration, enabling more secure, compartmentalized architectures. In general, this approach improves on the previous category not by offering stronger security guarantees, but by facilitating

⁹ One way to mitigate this risk is by escrowing keys with special parties called *escrow agents*. This additionally provides the benefit of better data compartmentalization. If one agent is compromised, only the data to which it stores keys is revealed.

functionality that was otherwise cumbersome (to the point of being infeasible) through the use of sophisticated cryptographic techniques.

Very few cloud data protection products offer fully automated access control enforcement. Tresorit [36], a Swiss secure cloud storage firm that operates its own storage servers, is one. Tresorit users generate and store their own private keys and use a public key agreement scheme called Tree-based Group Diffie-Hellman (TGDH) to establish keys for file sharing. Sharing is initiated via an email invitation that contains a key; the permission recipient authenticates with a certificate and presents the invitation key in order to establish their identity.

MIT Lincoln Laboratory (the author of this report) has developed an application, called Self-enforcing Security for the Cloud (SENSECL), for seamless cryptography and key management providing flexible, cryptographically enforced access-control policies ensuring data confidentiality, integrity, and authenticity.¹⁰

In SENSECL, each data item is encrypted on the client device using a unique, randomly generated content key that is itself encrypted by the public key of the authorized user and embedded in a cryptographic *permission*. Only the owner of the associated private key can exercise the permission and access the content. In the simplest example, an individual user owns the public-private key pair.

To access data stored through SENSECL, an authorized user would retrieve the protected content from the cloud and exercise the embedded cryptographic permission to remove the protection on the client device. Only authorized parties with the appropriate permissions can access the content. Unauthorized parties cannot extract the content key from the permission because they do not possess the necessary private key. Therefore, the provider and other unauthorized parties never have access to the unprotected content. Using this and other cryptographic methods, access control policies are defined and enforced without having to rely on other parties for the enforcement. These policies ensure data confidentiality, integrity and authenticity while enabling secure sharing with users and groups that have a need-to-know.

2.7 PERFORMANCE AND STORAGE COSTS

Data protection approaches that hide as much information as possible from the cloud service provider are without a doubt stronger and more secure than approaches that leak information. However, there is a tradeoff between information leakage to the cloud service providers and the functionalities that they can provide.

¹⁰ This work was sponsored in part by the Defense Information Systems Agency.

For example, searching through stored data is reasonably easy for cloud service providers to implement when users are not trying to protect their data from them. However, to attempt to implement this capability securely, vendors must turn to searchable encryption schemes, which by definition leak information about the data encrypted. Recent leakage-abuse attacks have demonstrated the difficulty of managing this leakage [37].

Storage costs can also rise when strong encryption is applied to data in storage. *File deduplication*—a process that removes identical copies of data, usually at the file or block level—requires the provider to be able to recognize when two pieces of data are identical. However, this property is directly in conflict with basic notions of the security of encryption. Therefore, if we only allow the provider to view data in a securely encrypted form, they cannot perform deduplication.

The effects of this limitation on total storage space are typically not severe, but they are also not negligible. Results from a large-scale study analyzing file data from 15 globally distributed file servers hosting data for over 2000 users [38] are shown in Figure 9. The results for the SharePoint server can be thought to represent enterprise storage; the group file server approximates group or team file storage, and the home file server models personal storage. Based on this data, for the U.S. government or another large organization, we anticipate that an additional 3–7% of storage will be required for any system that uses categories CO-4, CO-5, or CO-6.

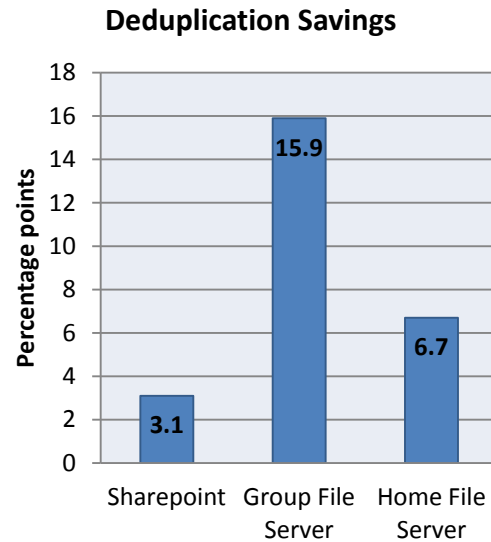


Figure 9. Storage space saved by data deduplication.

2.8 CATEGORIZATION AND ATTACK VECTORS SUMMARY

In this section, we summarize our confidentiality classification using two tables.

In Table 1, we reiterate our categorization and the security features offered by each approach. Included are lists of which products correspond to each category.

In Table 2, we organize the attack vectors discussed in this report and summarize how each approach addresses them. Attacks are left either as residual vulnerabilities (meaning they are not defended against), mitigated, or prevented.

The attack vectors in the table are defined as follows (see Figure 1 for a graphical representation):

- AT-1. *Data in use*: while data is being created or consumed on a client device, it is vulnerable to being accessed by an attacker using malware, surveillance software, side channel attacks, or other methods. This attack vector includes attacks by malicious *local* system administrators.
- AT-2. *Data at rest*: an attacker who obtains direct access to the storage server can compromise unprotected data while it is in storage.
- AT-3. *Subvert storage service*: the storage provider can directly provide data at rest protections, but if they can access the data then an attacker that is able to subvert the storage provider can as well.
- AT-4. *Subvert security provider*: an attacker can subvert the authentication or authorization services offered by the security provider. In Table 2 we show the location of the security provider for each approach. Some locations make the provider more difficult to subvert, but as a rule this class of attacks is a residual vulnerability for all categories. Subverting the security provider may result in
- a. *Complete compromise of data*: the attacker may impersonate a legitimate user, subvert the authenticator and become authenticated without valid credentials, or subvert the authorizer and receive access to data that they are not authorized to access.
 - b. *User compromise*: an authorized user's credentials may be stolen from a variety of locations: upon entry into the system via a keylogger, in transit to the authenticator, or from the authenticator itself. With these credentials, an attacker can access only the data that user was authorized to access.

Although it is not a separate attack vector per se, there is another notion that is extremely important in evaluating the security of each approach. If the security provider is centralized, it is a *single point of failure* for this system, and an adversary who compromises it can gain access to all of the stored data. Removing this single point of failure by decentralizing the security provider enables finer-grained data compartmentalization, which may not help to prevent security breaches but certainly helps to contain them and reduce their scale. As shown in Table 2, categories CO-5 and CO-6 achieve this goal to the highest standard by enabling each client device to act as the security provider for its own data.

Table 1: A Summary of the Security Features Offered by Each Category

Cat #	User authentication by server	Data protection at rest	Key management	Vendors
CO-1	No	No	N/A	N/A (nothing to sell)
CO-2	Yes, but a compromised monitor leaks user info	No	N/A	Acronis Access Advanced, iDrive, nCryptedCloud, Spideroak
CO-3	Yes	Yes, but vulnerable to insiders	By the cloud service provider	AWS S3, Barracuda CudaDrive, BlueCoat Data Protection Gateway, Citrix ShareFile, Cleversafe, CTERA, DropBox, Egnyte Enterprise File Sharing, Google Drive, Nasuni, Panzura, Sookasa CASB, Sophos SafeGuard, SugarSync, and Vaultive
CO-4	Yes, by the key management provider	Yes, but vulnerable to insiders	By the key manager service provider	Acellion Kiteworks, Box, Bitcasa Turnkey Drive, Credeon Cloud Data Protection, TitanFile, Viivo, and Vormetric
CO-5	Optional ¹¹	Yes	By the users or enterprise	AWS Encryption Client, ElephantDrive, IDrive, and OpenDrive. Also Bitlocker and other full-disk encryption tools
CO-6	Optional ¹¹	Yes	Automatically on the client	SENSECL, Tresorit

¹¹ Categories CO-5 and CO-6 do not rely on the security features provided by the server and hence user authentication by the server is not essential. While such authentication can be a useful feature in some settings, it can also have certain downsides. For example, it may unnecessarily leak identity and access pattern information to the server. Also, users might be tempted to reuse authentication with the server (e.g., if password authentication is used).

Table 2: A Summary of How Each Major Attack Vector Discussed in This Report is Addressed by Each Approach

Cat #	AT-1: Data in use	AT-2: Data at rest	AT-3: Subvert storage service	AT-4: Subvert security provider
CO-1	Residual vulnerability	Residual vulnerability	Residual vulnerability	No security provider, making the storage server a single point of failure
CO-2	Residual vulnerability	Slightly mitigated by the reference monitor, but still largely a residual vulnerability	Residual vulnerability	Storage service provides security, which does not mitigate the single point of failure
CO-3	Residual vulnerability	Prevented	Residual vulnerability	Storage service provides security, which does not mitigate the single point of failure
CO-4	Residual vulnerability	Prevented	Mitigated—attacker must also compromise key manager	Third party is the security provider, relocating but not mitigating the single point of failure
CO-5	Residual vulnerability	Prevented	Prevented	Client device provides security, containing user compromises (<i>b</i>) to just that user's data and eliminating the single point of failure
CO-6	Residual vulnerability	Prevented	Prevented	Client device provides security, containing user compromises (<i>b</i>) to just that user's data and eliminating the single point of failure

3. INTEGRITY

Even vendors that detail their mechanisms for ensuring data confidentiality and availability often fail to discuss or publicize their data integrity protections. These protections, when they exist, tend to focus on accidental data corruption rather than malicious modification of the data by a devious attacker. Therefore, we are unable to address data integrity in depth in this report. However, there is one attack, closely related to the family of data integrity attacks, that we do wish to address as its mitigation ties nicely into our existing categorization.

In a *replay* or *rollback attack*, data in storage is reverted to a previous state by an attacker or malicious administrator. If *all* of the data is stored in the cloud, where attackers can rollback all of it to a previous state, then this attack is impossible to defend against. If at least some state data can be protected from rollback— e.g., by being stored on a user’s smartcard or in some centralized database assumed to be incorruptible—then it is possible to protect against the rollback attack. In our categorization, approaches that maintain persistent storage that they assume cannot be rolled back use this assumption to defend against these attacks.¹² Categories without persistent storage (specifically, CO-5 and CO-6) are inherently vulnerable to this attack.

¹² Note that we do not distinguish which, if any, products in those categories provide such defense, as it is based solely on the assumption that the persistent storage service cannot be compromised. This assumption is fragile, particularly when the service in question is external to the enterprise. In general, we discount defense mechanisms that are founded upon the assumption that a certain resource is inaccessible to adversaries.

4. AVAILABILITY AND DENIAL-OF-SERVICE

In a *denial-of-service (DoS)* attack, an adversary attempts to attack data *availability*, which is described in the introduction of this report. Namely, an adversary tries to prevent users from performing authorized actions on data (e.g., retrieving the data, recognizing it as valid, and properly changing and storing data). In the cloud storage setting, adversaries mounting a DoS attack have no shortage of attack vectors—they can attack communications between the user and the storage server, attempt to disable the network connection entirely, or delete or modify the data in storage itself.

One particular DoS attack vector is made accessible in categories CO-3 and higher. When data is encrypted, the encryption keys are susceptible to corruption or deletion; this renders data impossible to decrypt and thus inaccessible to users. In category CO-6, when permissions are cryptographic objects, they can be similarly targeted by attackers. This means that in these approaches, keys must be protected not only from attacks against their confidentiality but also their availability. The keys used for authentication in these schemes can be protected by careful *key escrow* mechanisms in which keys are shared with escrow agents. Preferably, multiple different agents should be used to avoid creating a single point of failure. Improving the availability of the keys does require increasing their attack surface, but escrow policies can be defined to mitigate these risks. For example, a policy may require a set of agents to cooperate and use data usually inaccessible from the network in order to recover the keys. Furthermore, the encoding techniques described in this appendix can also be used to protect stored keys and permissions from DoS attacks.

While DoS attacks that target resources outside of the storage server are not in scope for this report, preventing data in storage from being modified or deleted is an important and relevant consideration. However, it is clearly impossible to prevent certain types of attacks, such as a cloud provider that tampers with data stored on its servers. Therefore, rather than focusing on the *prevention* of DoS attacks, we will instead briefly detail techniques used to *mitigate* threats to availability. In the diagrams that accompany each technique, “SM” indicates where storage management is handled

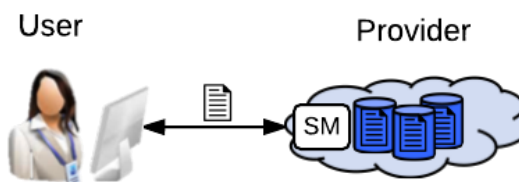


Figure 10. A single provider duplicates (or encodes) files across multiple storage servers.

The simplest technique to help maintain users’ access to their data in the face of unreliable network connections or unavailable cloud providers is to duplicate stored files across multiple storage locations (see Figure 10). Storage providers often use this technique: Amazon Web Services’ Simple Storage Service, Microsoft Azure [39], and Google Cloud Platform Cloud Storage [40] [41] are a few examples. If a subset of a provider’s storage centers are unavailable, users can simply access copies of their objects stored elsewhere.

Many of the vendors surveyed for this report do not have in-house storage servers; they offer data protection mechanisms but outsource storage to a third-party provider. In this case, data can be similarly duplicated across multiple storage providers (see Figure 11).

Ideally, this duplication should be transparent to the user: when they go to retrieve a file, they should not need to know which providers are available, which provider the file was retrieved from, or what credentials were used to authenticate to that provider (see Figure 12). Although many vendors allow for interchangeable back-end storage, to our knowledge only SENSECL and Avere [42] enable data to be redundantly stored across multiple providers in a way that allows for transparent, automatic failover.

The problem with fully duplicating all user data is that it consumes significant additional storage space. To improve on this technique, a type of error correcting code known as an *erasure code* can be used to split each file into pieces. These pieces are then stored across different providers; a file can be recovered from a subset of those pieces, allowing for data to be accessed even if some providers are unavailable. The number of pieces stored and the number of these pieces that can become unavailable without making the encoded data unavailable are configurable, allowing for an enterprise to determine the ratio between storage space consumed and availability protection enabled. By way of example, the default settings for Tahoe-LAFS [43] spread data over ten disks and can recover the data using any three pieces. The storage cost of their erasure coding is just 10/3 or 3.3 times the normal storage cost of the file, and they note that this encoding is more reliable than comparable Redundant Array of Independent Disks (RAID) arrangements [44]. Reducing the erasure threshold so that all but three disks must be available for a successful recovery would reduce that storage cost to a factor of 1.7. In addition to Tahoe-LAFS, this technique is also used by SENSECL and Cleversafe to maintain data availability.

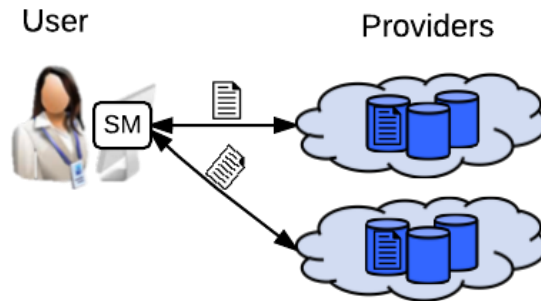


Figure 11. Files are duplicated (or encoded) and stored across multiple providers.

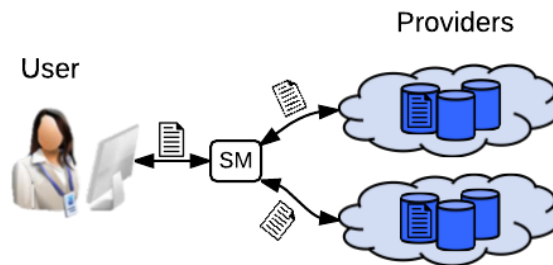


Figure 12. Files are duplicated (or encoded) and stored across multiple providers in a way that is transparent to the user.

5. CONCLUSION

The advent of cloud computing has caused government organizations to rethink their IT architectures so that they can take advantage of the lower costs and increased flexibility provided by clouds. DoD is participating in this effort and is both outsourcing infrastructure and services to commercial providers and insourcing to shared internal clouds. The difficulty with this model is ensuring confidentiality, integrity, and availability of data stored on systems managed by external organizations. This report provides a framework for analyzing the tradeoffs between different methods of supporting these protections and indicates where certain cloud storage products fall within this framework.

Our main contribution is the organization of cloud data confidentiality mechanisms into six categories. These categories are differentiated based on the fundamental notion of which adversaries they are capable of protecting against and which they are not. This approach allows us to arrive at a clean understanding of the different methods by which data and keys can be protected and distributed to users and what tradeoffs must be made between security and functionality. This report also discusses the limitations of maintaining the integrity of data stored in the cloud and describes methods for improving data availability. From this report, an approach to cloud data protection can be chosen depending on enterprise infrastructure, security specifications, and functionality requirements.

This page intentionally left blank.

APPENDIX A

AUTHENTICATION

How users authenticate to a cloud storage system is a significant factor contributing to its security; this section will briefly detail approaches to authentication. As in the body of the report, this appendix is primarily concerned with the relationships between the parties in the protocol rather than the precise details of how cryptographic primitives are used or implemented.

In order to be authenticated, a user must possess some unique, distinguishing features or information. In some authentication systems, the user must demonstrate this uniqueness to a *verifier* who can establish that it corresponds to the identity the user is claiming. If the verifier is satisfied, then the authorizations corresponding to the authenticated identity are granted to the user.

Similarly to how confidentiality categories CO-5 and CO-6 eschew a reference monitor, other authentication systems seek to avoid the risks of a mediator. Instead, they assure directly that users can only exercise *permitted* capabilities without the use of any third party. This is enabled by the derivation of a cryptographic authentication key from the user's uniqueness. This key can be used to exercise cryptographic permissions.

There are a variety of ways that authentication mechanisms can enable each user to uniquely distinguish themselves. These mechanisms can be analyzed, similarly to how we categorize confidentiality protection approaches in the body of this report, based on the inherent residual vulnerabilities.

Most traditional authentication mechanisms have the user, acting as the *prover*, interacting with an authentication service acting as the *verifier*. Authentication is successful when the verifier *accepts*. A depiction of what this looks like in the context of cloud storage can be seen in Figure 13.

We will refer to authentication mechanisms that use a verifier as *binary*, since their result is a binary decision (accept or fail). Each binary mechanism provides different information to the verifier, which we call verifier information (VI). This VI can be public or private, and the impact of its compromise varies depending on the approach. We organize binary approaches later in this appendix according to this VI, and the attack vectors enabled by its compromise.

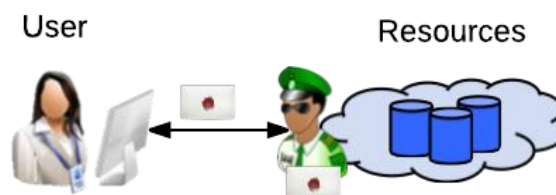


Figure 13. Authentication with a verifier. The verifier checks that the user knows a particular secret; if they do, the verifier (acting here as a reference monitor) grants the user access to the resources.

As we have argued in the body of this report, relying on trusted, centralized guards such as a verifier can be a liability. These trusted parties are a popular target for subversion, and in particular when they are located on remote servers (i.e. in the cloud), they may not be trustworthy. For this reason, as a general rule we say that binary authentication mechanisms have the inherent weakness of being vulnerable to malicious or corrupted verifiers accepting incorrect secrets.

The second approach to authentication eliminates the need for a verifier, thus mitigating this vulnerability. Specifically, *resilient* authentication mechanisms allow the user to locally generate a cryptographic key that can be used to access cryptographically protected resources (see Figure 14). Resilient authentication can be thought of as “self-enforcing” — a user with the ability to generate the necessary cryptographic key can authenticate, but a user without the key cannot, nor can they simply flip a few bits to subvert a binary authenticating guard. Of course, even in resilient authentication schemes, the device generating and/or storing the key remains a residual vulnerability

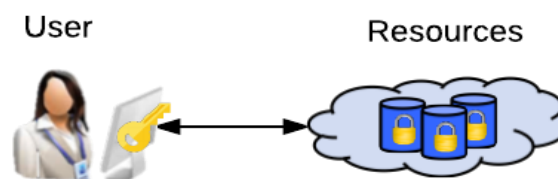


Figure 14. Authentication without a verifier. The user locally derives a key which can be used to directly access cryptographically protected resources.




The distinction between these two approaches forms the basis of our classification; other factors include the type of VI (public or private), whether the verifier learns the VI, whether successive authentication attempts reuse proof information or vary it, and how much of that proof information the verifier is able to learn. The full classification can be seen at the end of this appendix in Table 4.

Another distinguishing factor between authentication mechanisms is the *modality* used (see Table 3). These modalities are generally organized into three categories: (i) what you have, such as a hardware token like a smartcard, (ii) what you are, or biometrics such as iris scans or fingerprints, and (iii) what you know—e.g., a password. Each of these modalities has strengths and weaknesses. It is also possible to combine multiple modalities; this is known as *multi-factor authentication*. Ideally, this should be implemented in a way that will make the result stronger than its individual components, but this requires some expertise and care as it is possible for the composition of multiple modalities to compound their weaknesses rather than their strengths.

Category AU-1

In simpler binary schemes, verifiers are trusted fully with all of the authentication information and thus form the most vulnerable and least secure authentication mechanisms. In this category is the password matching approach, where the VI is the correct password for each user. When authenticating, the users provide their IDs and passwords and the verifier checks for a match. In addition to the inherent weakness of binary authentication described above, this mechanism also makes the passwords susceptible to theft by anyone with access to the verifier’s storage.

Table 3: Authentication Modalities

	What you have	What you are	What you know
	 <p>Hardware tokens</p>	 <p>Biometrics</p>	 <p>Passwords</p>
Entropy	High	Medium	Low in practice
Acquisition Fidelity	Perfect	Low	High
Theft	Token stolen or lost	Stolen from <i>local</i> user	Sharing, reuse, key-loggers
Interface	Device reader	Custom	Any
Convenience	Carry around token	Convenient if works	Annoying, but common

Biometric authentication mechanisms mirror the password matching technique: they use a *reference template* generated from a biometric reading during enrollment instead of the stored password or its hash. To authenticate, a new biometric reading is provided to the verifier, which would then compute the new template and match it against the reference template. The match is successful if the two templates are sufficiently close by the appropriate metric. It has been demonstrated, however, that given a reference template it is easy to manufacture corresponding biometrics that would result in a match [45] [46]. While there are ways to derive reference templates that make it difficult to extract the original reading, the readings provided during authentication are still available for capture by an adversary. The theft of biometric data is a particularly grave threat because biometrics are permanently associated with a user, so the theft of a particular biometric reading such as fingerprints, forever compromises that user's ability to use that form of authentication for any system.

Category AU-2

A slightly more secure approach would be for the user to provide to the verifier a one-way function of the password such as a standard password hash. This is how password authentication is typically implemented in practice; virtually all cloud storage providers surveyed offer this mechanism. The problem with this approach is that the low entropy of human-usable passwords make them vulnerable to exhaustive search attacks (*password cracking*). Furthermore, during the authentication process, the users provide their passwords to the verifier so that the password hashes can be computed and matched. Therefore a malicious or corrupted verifier can steal users' passwords at that point.

Category AU-3

Another weakness of password authentication becomes important if the verifier itself is not properly authenticated: an attacker may pretend to be the verifier and steal passwords this way. This is a *man-in-the-middle attack*, and it occurs in a variety of ways from wi-fi honeypots[47] to credit card skimmers [48]. To combat this attack, *challenge-response* protocols can be used. In these protocols, the user does not provide a password directly but rather provides some function of the password in conjunction with a (random) challenge issued by the verifier. This mechanism prevents an attacker from learning credentials by impersonating the verifier but does not prevent attacks available to an attacker who obtains access to VI.

In this same vein, if both parties know a secret password, they can use a *password authenticated key exchange (PAKE)* protocol to mutually compute a cryptographic key. The advantage of these schemes is that weak (low entropy) passwords can be used securely because they are not vulnerable to offline brute-force attacks (in contrast to the less sophisticated password and challenge-response protocols). This mitigates password cracking attacks, but does not solve the fundamental problem associated with the verifier storing a shared secret as VI.

Category AU-4

Better approaches use more sophisticated cryptography that allow for the use of *public* VI. In other words, the security of these approaches does not rely on VI remaining secret. For example, using public key signatures, a user can authenticate to the verifier by signing some random messages provided by the verifier (as a kind of an improved challenge-response protocol). While such a protocol could be implemented securely, this approach still might be risky. One particular risk occurs if a user's public signature key is used for other purposes in addition to authentication; malicious verifiers can trick a user into signing documents (such as a check or contract) that the user did not intend to sign.

Category AU-5

A stronger cryptographic option is to use *zero-knowledge identification protocols* or *zero-knowledge proofs of knowledge (ZPoK)*. These protocols allow the verifier to verify that the prover knows

the right secret without learning anything else. This is a strong mechanism for implementing verifier-based authentication; however, these protocols cannot be performed by humans and require some device, such as a smartcard, to store the secret and perform the necessary computations. Therefore this mechanism is only implementable by the “what you have” modality shown in Table 3.

These protocols are still vulnerable to an attacker impersonating the verifier to the user or the user to the verifier. Preventing such an attack requires authenticating the verifier as well as the user, even if the verifier cannot learn anything from the protocol itself as in the ZPoK case. Most modern systems implement the connection between the user and verifier using TLS/SSL, which does perform authentication of the verifier. However, one should carefully analyze each specific implementation of this approach and carefully evaluate the residual vulnerabilities, such as corrupted Certificates of Authority (CA) or the users ignoring or bypassing TLS warnings.

Category AU-6

In this category are authentication mechanisms that do not rely on any verifiers. Instead, these mechanisms directly derive the authentication key for each user and then use this key to exercise cryptographic permissions. SENSECL and Tresorit are in this category. Some other tools derive authentication keys, but still rely on a reference monitor approach (Kerberos [49]) or use the keys only to access non-shared data (SpiderOak).

Table 4: Authentication Categories

(In this context, “computational” protection refers to the entropy of the information in question. Low entropy information is easier for an attacker to compromise; high entropy information is more difficult.)

	Category					
	AU-1	AU-2	AU-3	AU-4	AU-5	AU-6
Authentication type	Verifier-based (binary)					Resilient
VI type	Private			Public		N/A
VI leaked to verifier	Full	Computational (low entropy)	Full	Computational (high entropy)		N/A
Proof info¹³	Fixed	Variable				N/A
Proof info leaked to verifier	Full		Partial (computational or hard to abuse)		None	N/A

¹³ This row refers to whether the information exchanged during the proof changes across successive authentication attempts. If it does not, we say the proof is *fixed*. If it does, it is *variable*.

Conclusion

While each of these forms of authentication is widely used across products in a variety of categories, the majority of cloud providers surveyed offer authentication solely via category AU-2. Many of these also offer optional *multi-factor authentication*; that is, authentication using multiple modalities in tandem, usually passwords and hardware tokens.

A few tools distinguish themselves from the pack. SpiderOak, Tresorit, and Kerberos, among others, derive keys from passwords using methods such as Password-Based Key Derivation Function 2 (PBKDF2). This has advantages over password matching; specifically, it eliminates the single point of failure presented by a verifier. However, brute force techniques are fairly effective against popular password-based key derivation algorithms, and in the cloud environment where computational power can be enormous, this is a serious threat. SENSECL provides several authentication mechanisms: primarily smartcard authentication (using DoD CACs) but also password and biometric key derivation techniques. SENSECL authentication is non-binary (does not require a verifier), and to our knowledge it is also the first demonstration of the viability of iris biometrics as a non-binary authentication technique in a software product.

GLOSSARY

RAID	Redundant Array of Independent Disks
VI	Verifier Information
PAKE	Password Authenticated Key Exchange
ZPOK	Zero-Knowledge Proofs of Knowledge
CA	Certificate of Authority
PBKDF2	Password-Based Key Derivation Function 2
FIPS	Federal Information Processing Standards
SecAAS	Security as a Service
HSMs	Hardware Security Modules
SDK	Software Development Kit
TGDH	Tree-Based Group Diffie-Helman
SENSECL	Self-Enforcing Security for the Cloud
DoS	Denial-of-Service
IDC	International Data Corporation
FedRAMP	Federal Risk and Authorization Program
PKI	Public Key Infrastructure
CSA	Cloud Security Alliance
AES	Advanced Encryption Standard
TLS/SSL	Transport Layer Security/Security Socket Layer

This page intentionally left blank.

REFERENCES

- [1] International Data Corporation. (2015, October) IDC Forecasts Worldwide Cloud IT Infrastructure Market to Grow 24% Year Over Year in 2015, Driven by Public Cloud Datacenter Expansion. Available at: www.idc.com.
- [2] FedRAMP. Federal Risk and Authorization Management Program. Accessed on February 12, 2016. <https://www.fedramp.gov>.
- [3] Cloud Security Alliance, "SecaaS Implementation Guidance, Category 8: Encryption ," 2012. Available at: <https://cloudsecurityalliance.org/>.
- [4] James P. Anderson, Computer Security Technology Planning Study Volume II, ESD-TR-73-51, Vol. II, Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, MA (Oct. 1972). Available at: <http://csrc.nist.gov>.
- [5] Acronis. Access Advanced. Accessed on February 12, 2016. <http://www.acronis.com/en-us/mobility/access-advanced/>.
- [6] SpiderOak. Features. Accessed on February 12, 2016. <https://spideroak.com/features/zero-knowledge>.
- [7] IDrive. IDrive Inc. Web site. Accessed on February 12, 2016. <https://www.idrive.com/online-backup-security>.
- [8] nCryptedCloud. (Retrieved January, 20) nCryptedCloud Technical Overview of Encryption and Key Management. Available at: <https://www.ncryptedcloud.com>.
- [9] Dropbox. (Retrieved 2016, February 18) Dropbox business security, a dropbox whitepaper. Available at: <https://cf.dropboxstatic.com>.
- [10] Barracuda. (Retrieved 2016, February 18) CudaDrive Data Sheet. Available at: <https://www.barracuda.com/support/documentation/whitepapers>.
- [11] BlueCoat. (Retrieved 2016, February 16) Cloud Data Protection Gateway Encryption - Solution Brief. Available at: <https://www.bluecoat.com/resources/cloud-data-protection/encryption>.
- [12] Cleversafe. (Retrieved 2016, February 12) Secure Data Storage in A Cloud Infrastructure. <https://www.cleversafe.com/resources/>.

- [13] CTERA Networks. (Retrieved 2016, January 25) CTERA End-to-End Security. Available at: <https://kb.ctera.com/category/documentation/>.
- [14] CTERA Networks. (Retrieved 2016, January 25) CTERA Security Frequently Asked Questions.
- [15] Citrix. (Retrieved 2016, February 12) ShareFile Enterprise: security white paper. Available at <https://www.citrix.com>.
- [16] Egnyte Inc. (2014, January 25) (Retrieved 2016, January 25). Egnyte Security Architecture White Paper.
- [17] Sookasa. Resources. Accessed on January 25, 2016.. <https://www.sookasa.com/resources/google-drive-encryption/>.
- [18] Nasuni. (Retrieved 2016, February 22) Technical Brief: Nasuni Security Model. Available at: <https://nasuni.com>.
- [19] Panzura. (Retrieved 2016, February 22) White Paper: Panzura Global Cloud Storage System. Available at: <http://panzura.com/resources/white-papers/>.
- [20] Sookasa. (Retrieved 2016, February 16) Sookasa Whitepaper: Security. Available at: <https://sookasa.com>.
- [21] Sophos. (Retrieved 2016, February 22) SafeGuard Encryption for Cloud Storage. Available at <https://www.sophos.com>.
- [22] SugarSync. Features. Accessed on January 25, 2016. <https://www.sugarsync.com/en/features>.
- [23] Vaultive. Solutions.. Accessed on February 12, 2016. <http://vaultive.com/>.
- [24] Amazon Web Services. (2015, August) Amazon Web Services: Overview of Security Processes. Available at: <https://aws.amazon.com/whitepapers/overview-of-security-processes/>.
- [25] Accellion. (Retrieved 2016, February 12) Security Overview: kiteworks. Available at: <https://www.accellion.com>.
- [26] Box. Box Web site. Accessed on February 12, 2016. <https://www.box.com/business/keysafe/>.
- [27] Jon Brodtkin. (2015, February 12) Box hands cloud encryption keys over to its customers. Available at: <http://arstechnica.com/information-technology/2015/02/box-hands-cloud-encryption-keys-over-to-its-customers/>.

- [28] Bitcasa. (2015, July) At Bitcasa, even we can't see your data. And we prefer it that way. <https://blog.bitcasa.com/2015/07/31/at-bitcasa-were-blind-and-we-prefer-it-that-way/>.
- [29] Hitachi. (Retrieved 2016, February 12) Credeon Cloud Data Protection White Paper. Available at: <https://psg.hitachi-solutions.com/credeon/download-white-paper>.
- [30] TitanFile. Features. Accessed on February 12, 2016. <https://www.titanfile.com/features/>.
- [31] Viivo. Viivo Web Site. Accessed on February 12, 2016. <https://viivo.com/how-our-security-works>.
- [32] Vormetric. (Retrieved 2016, February 22) Vormetric Data Security Platform Data Sheet. Available at: <http://www.vormetric.com>.
- [33] ElephantDrive. How is my data secured at ElephantDrive? Accessed on February 12, 2016. <http://support.elephantdrive.com/hc/en-us/articles/205537248-How-is-my-data-secured-at-ElephantDrive>.
- [34] OpenDrive. OpenDrive Web site. Accessed on February 12, 2016. <https://www.opendrive.com/security>.
- [35] Microsoft. Bitlocker Drive Encryption Overview. Accessed on February 12, 2016. <http://windows.microsoft.com/en-us/windows-vista/bitlocker-drive-encryption-overview>.
- [36] Tresorit. (Retrieved 2016, February 12) End-to-End Encrypted Cloud Storage. [Online]. <https://www.tresorit.com>.
- [37] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart, "Leakage-Abuse Attacks Against Searchable Encryption ," in *22nd ACM SIGSAC Conference on Computer and Communications Security* , 2015, pp. 668-679.
- [38] Ahmed El-Shimi et al., "Primary Data Deduplication—Large Scale Study and System Design," Proceedings of the USENIX conference on Annual Technical Conference, p.26–26, June 13–15, 2012.
- [39] Microsoft. (Retrieved 2016, February 12.) Microsoft Trust Center // Security. Available at: <https://www.microsoft.com/en-us/TrustCenter/Security/>.
- [40] Google. (Retrieved 2016, February 12) Google Cloud Platform Security. Available at: <https://cloud.google.com/security/whitepaper>.
- [41] Dave Barth. (2013, August) Google Cloud Storage now provides server-side encryption. <https://cloudplatform.googleblog.com/2013/08/google-cloud-storage-now-provides.html>.

- [42] Avere. (Retrieved 2016, February) Hybrid Cloud NAS Architecture. Available at: <http://lp.averesystems.com/avere-cloud-nas-architecture-white-paper>.
- [43] Zooko Wilcox-O'Hearn and Brian Warner, "Tahoe: the least-authority filesystem," in *4th ACM international workshop on Storage security and survivability*, 2008, pp. 21-26.
- [44] Tahoe-LAFS. Tahoe-LAFS FAQ. Accessed on February 12, 2016. <https://tahoe-lafs.org/trac/tahoe-lafs/wiki/FAQ>.
- [45] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia, "From the iriscodes to the iris: A new vulnerability of iris recognition systems," in *Black Hat USA*, 2012.
- [46] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia, "Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1512-1525, October 2013.
- [47] Laurent Oudot. (2004, February) Symantec Security Articles. <http://www.symantec.com/connect/articles/wireless-honeypot-countermeasures>.
- [48] Brian Krebs. (2010, January) Krebs on Security. <http://krebsonsecurity.com/all-about-skimmers/>.
- [49] Massachusetts Institute of Technology. Kerberos: The Network Authentication Protocol. Accessed on February 12, 2016. <http://web.mit.edu/kerberos/>.
- [50] Mike McCammon. (2016, January) Press Release. SpiderOak Semaphore to Give Team Collaboration what It's Missing: Privacy. Available at: <http://www.reuters.com/article/mo-spideroak-idUSnBw285747a+100+BSW20160128>.
- [51] Alex Wilhelm, "Box Introduces New Client-Controlled Encryption Tool To Bring Stragglers To The Cloud," *TechCrunch*, February 2015.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 07-07-2016		2. REPORT TYPE Technical Report		3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE Charting a Security Landscape in the Clouds: Data Protection and Collaboration in Cloud Storage				5a. CONTRACT NUMBER FA8721-05-C-0002 and/or FA8702-15-D-0001	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) G. Itkis, B. Kaiser, J. Coll, W. Smith, R. Cunningham, Group 53				5d. PROJECT NUMBER 10235-271	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) MIT Lincoln Laboratory 244 Wood Street Lexington, MA 02420-9108				8. PERFORMING ORGANIZATION REPORT NUMBER TR-1210	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Department of Homeland Security				10. SPONSOR/MONITOR'S ACRONYM(S) DHS	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement A: Approved for public release, distribution unlimited.					
13. SUPPLEMENTARY NOTES					
<p>This report surveys different approaches to securely storing and sharing data in the cloud based on traditional notions of security: confidentiality, integrity and availability, with the main focus on confidentiality. An appendix discusses the related notion of how users can securely authenticate to cloud providers.</p> <p>We propose a metric for comparing secure storage approaches based on their <i>residual vulnerabilities</i>: attack surfaces against which an approach cannot protect. Our categorization therefore ranks approaches from the weakest (the most residual vulnerabilities) to the strongest (the fewest residual vulnerabilities). In addition to the security provided by each approach, we also consider their inherent costs and limitations. This report can therefore help an organization select a cloud data protection approach that satisfies their enterprise infrastructure, security specifications, and functionality requirements.</p>					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as report	18. NUMBER OF PAGES 50	19a. NAME OF RESPONSIBLE PERSON
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (include area code)

This page intentionally left blank.