Chapter 10

Cloud Security Mechanisms

10.1 Encryption

10.2 Hashing

10.3 Digital Signature

10.4 Public Key Infrastructure (PKI)

10.5 Identity and Access Management (IAM)

10.6 Single Sign-On (SSO)

10.7 Cloud-Based Security Groups

10.8 Hardened Virtual Server Images

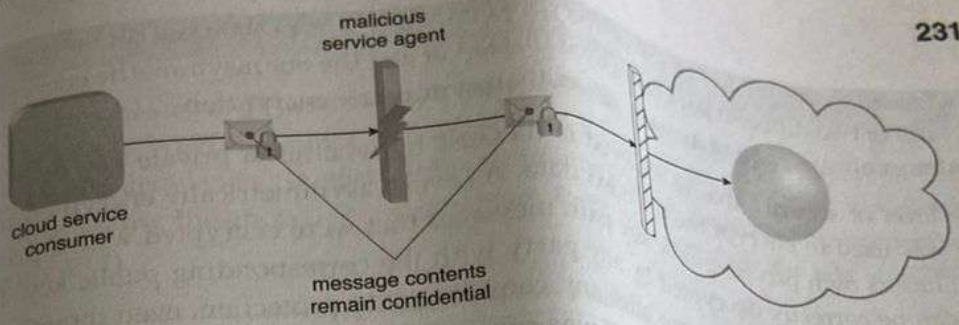

Figure 10.1
A malicious service agent is unable to retrieve data from an encrypted message. The retrieval attempt may furthermore be revealed to the cloud service consumer. (Note the use of the lock symbol to indicate that a security mechanism has been applied to the message contents.)

There are two common forms of encryption known as *symmetric encryption* and *asymmetric encryption*.

Symmetric Encryption

Symmetric encryption uses the same key for both encryption and decryption, both of which are performed by authorized parties that use the one shared key. Also known as *secret key cryptography*, messages that are encrypted with a specific key can be decrypted by only that same key. Parties that rightfully decrypt the data are provided with evidence that the original encryption was performed by parties that rightfully possess the key. A basic authentication check is always performed, because only authorized parties that own the key can create messages. This maintains and verifies data confidentiality.

Note that symmetrical encryption does not have the characteristic of non-repudiation, since determining exactly which party performed the message encryption or decryption is not possible if more than one party is in possession of the key.

Asymmetric Encryption

Asymmetric encryption relies on the use of two different keys, namely a private k and a public key. With asymmetric encryption (which is also referred to as *public* *cryptography*), the private key is known only to its owner while the public key is co monly available. A document that was encrypted with a private key can only be rectly decrypted with the corresponding public key. Conversely, a document that

Chapter 10

This chapter establishes a set of fundamental cloud security mechanisms, several of which can be used to counter the security threats described in Chapter 6.

10.1 Encryption

Data, by default, is coded in a readable format known as *plaintext*. When transmitted over a network, plaintext is vulnerable to unauthorized and potentially malicious access. The *encryption* mechanism is a digital coding system dedicated to preserving the confidentiality and integrity of data. It is used for encoding plaintext data into a protected and unreadable format.

Encryption technology commonly relies on a standardized algorithm called a *cipher* to transform original plaintext data into encrypted data, referred to as *ciphertext*. Access to ciphertext does not divulge the original plaintext data, apart from some forms of metadata, such as message length and creation date. When encryption is applied to plaintext data, the data is paired with a string of characters called an *encryption key*, a secret message that is established by and shared among authorized parties. The encryption key is used to decrypt the ciphertext back into its original plaintext format.

The encryption mechanism can help counter the traffic eavesdropping, malicious intermediary, insufficient authorization, and overlapping trust boundaries security threats. For example, malicious service agents that attempt traffic eavesdropping are unable to decrypt messages in transit if they do not have the encryption key (Figure 10.1).

CASE STUDY EXAMPLE

Innovartus has recently learned that users who access their User Registration Portal via public Wi-Fi hot zones and unsecured LANs may be transmitting personal user profile details via plaintext. Innovartus immediately remedies this vulnerability by applying the encryption mechanism to its Web portal via the use of HTTPS (Figure 10.2).

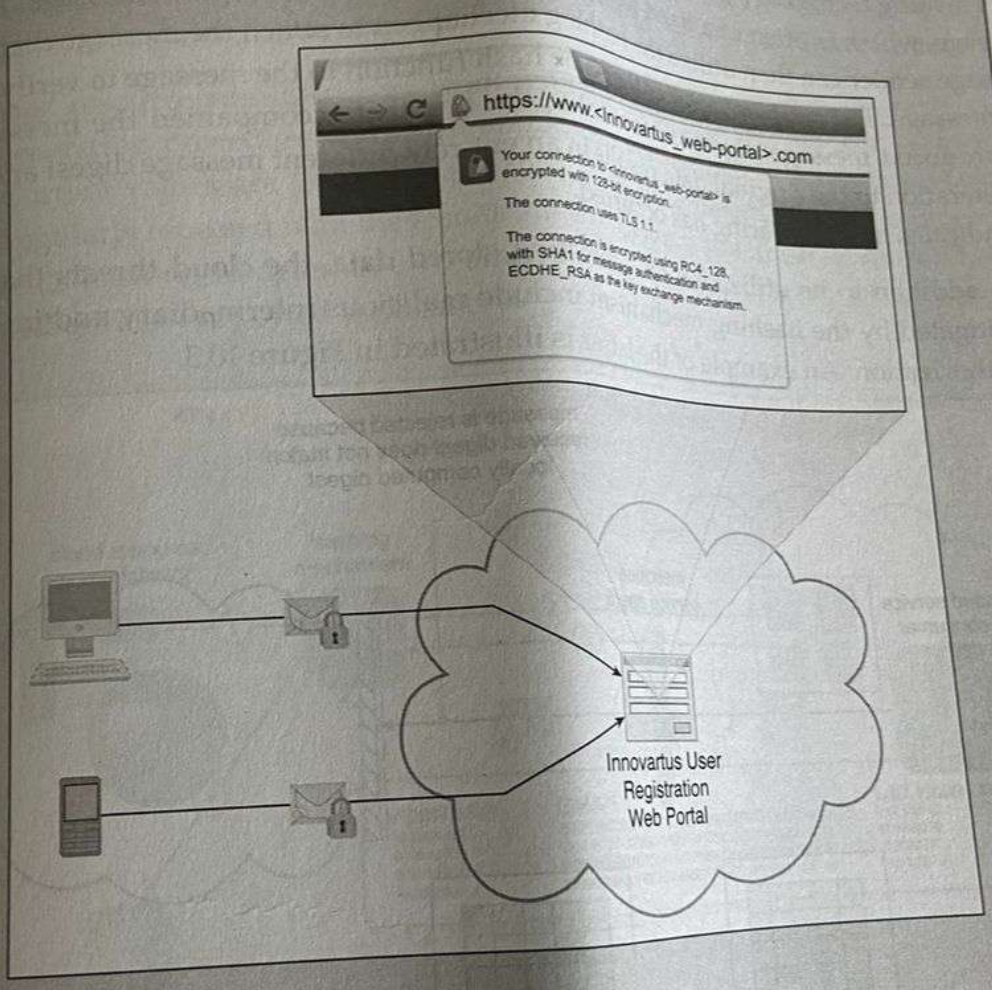

Figure 10.2
The encryption mechanism is added to the communication channel between outside users and Innovartus' User Registration Portal. This safeguards message confidentiality via the use of HTTPS.

encrypted with a public key can be decrypted only using its private key counterpart. As a result of two different keys being used instead of just the one, asymmetric encryption is almost always computationally slower than symmetric encryption.

The level of security that is achieved is dictated by whether a private key or public key was used to encrypt the plaintext data. As every asymmetrically encrypted message has its own private-public key pair, messages that were encrypted with a private key can be correctly decrypted by any party with the corresponding public key. This method of encryption does not offer any confidentiality protection, even though successful decryption proves that the text was encrypted by the rightful public key owner. Private key encryption therefore offers integrity protection in addition to authenticity and non-repudiation. A message that was encrypted with a public key can only be decrypted by the rightful private key owner, which provides confidentiality protection. However, any party that has the public key can generate the ciphertext, meaning this method provides neither message integrity nor authenticity protection due to the communal nature of the public key.

NOTE

The encryption mechanism, when used to secure Web-based data transmissions, is most commonly applied via HTTPS, which refers to the use of SSL/TLS as an underlying encryption protocol for HTTP. TLS (transport layer security) is the successor to the SSL (secure sockets layer) technology. Because asymmetric encryption is usually more time-consuming than symmetric encryption, TLS uses the former only for its key exchange method. TLS systems then switch to symmetric encryption once the keys have been exchanged.

Most TLS implementations primarily support RSA as the chief asymmetrical encryption cipher, while ciphers such as RC4, Triple-DES, and AES are supported for symmetrical encryption.

CASE STUDY EXAMPLE

A subset of the applications that have been selected to be ported to ATN's PaaS platform allows users to access and alter highly sensitive corporate data. This information is being hosted on a cloud to enable access by trusted partners who may use it for critical calculation and assessment purposes. Concerned that the data could be tampered with, ATN decides to apply the hashing mechanism as a means of protecting and preserving the data's integrity.

ATN cloud resource administrators work with the cloud provider to incorporate a digest-generating procedure with each application version that is deployed in the cloud. Current values are logged to a secure on-premise database and the procedure is regularly repeated with the results analyzed. Figure 10.4 illustrates how ATN implements hashing to determine whether any non-authorized actions have been performed against the ported applications.

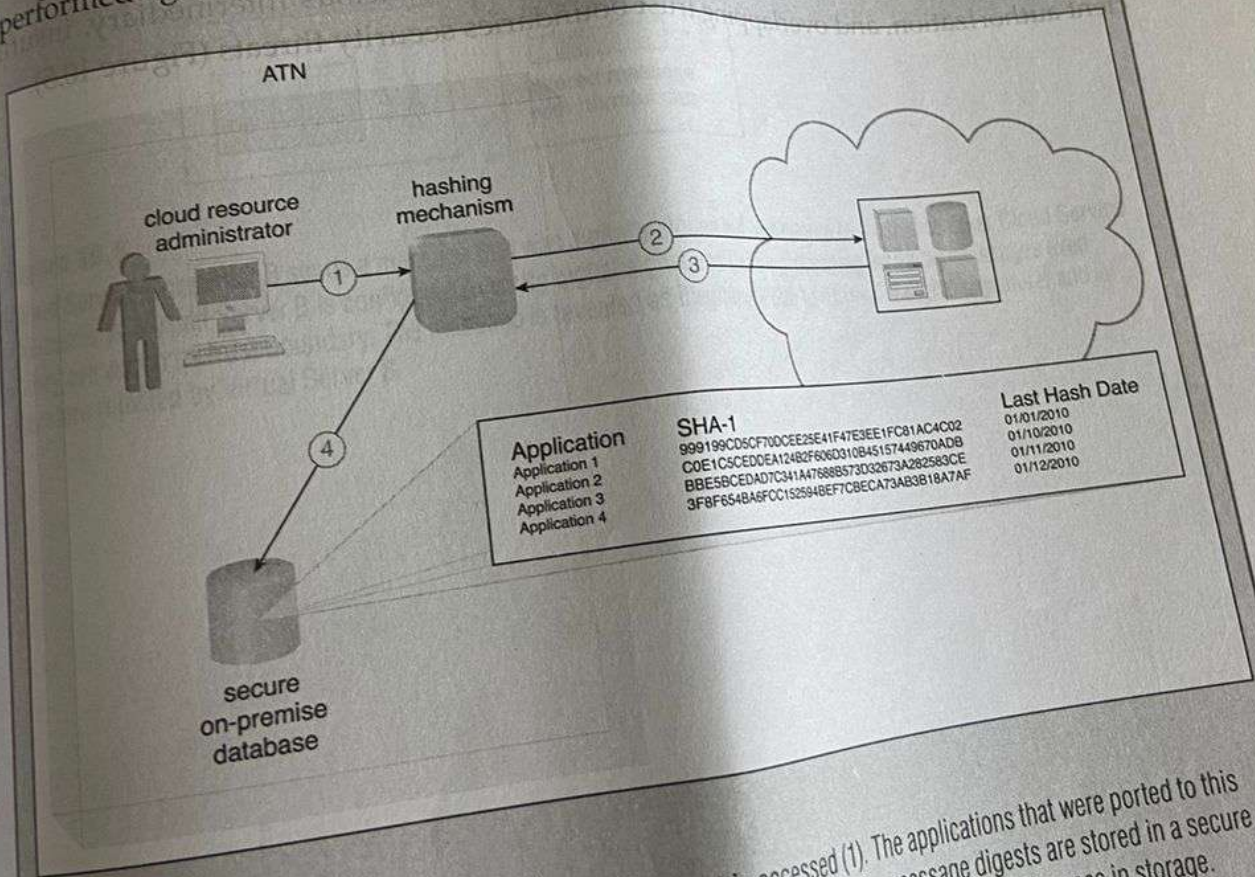

Figure 10.4
A hashing procedure is invoked when the PaaS environment is accessed (1). The applications that were ported to this environment are checked (2) and their message digests are calculated (3). The message digests are stored in a secure on-premise database (4), and a notification is issued if any of their values are not identical to the ones in storage.

10.2 Hashing

The *hashing* mechanism is used when a one-way, non-reversible form of data protection is required. Once hashing has been applied to a message, it is locked and no key is provided for the message to be unlocked. A common application of this mechanism is the storage of passwords.

Hashing technology can be used to derive a hashing code or *message digest* from a message, which is often of a fixed length and smaller than the original message. The message sender can then utilize the hashing mechanism to attach the message digest to the message. The recipient applies the same hash function to the message to verify that the produced message digest is identical to the one that accompanied the message. Any alteration to the original data results in an entirely different message digest and clearly indicates that tampering has occurred.

In addition to its utilization for protecting stored data, the cloud threats that can be mitigated by the hashing mechanism include malicious intermediary and insufficient authorization. An example of the latter is illustrated in Figure 10.3.

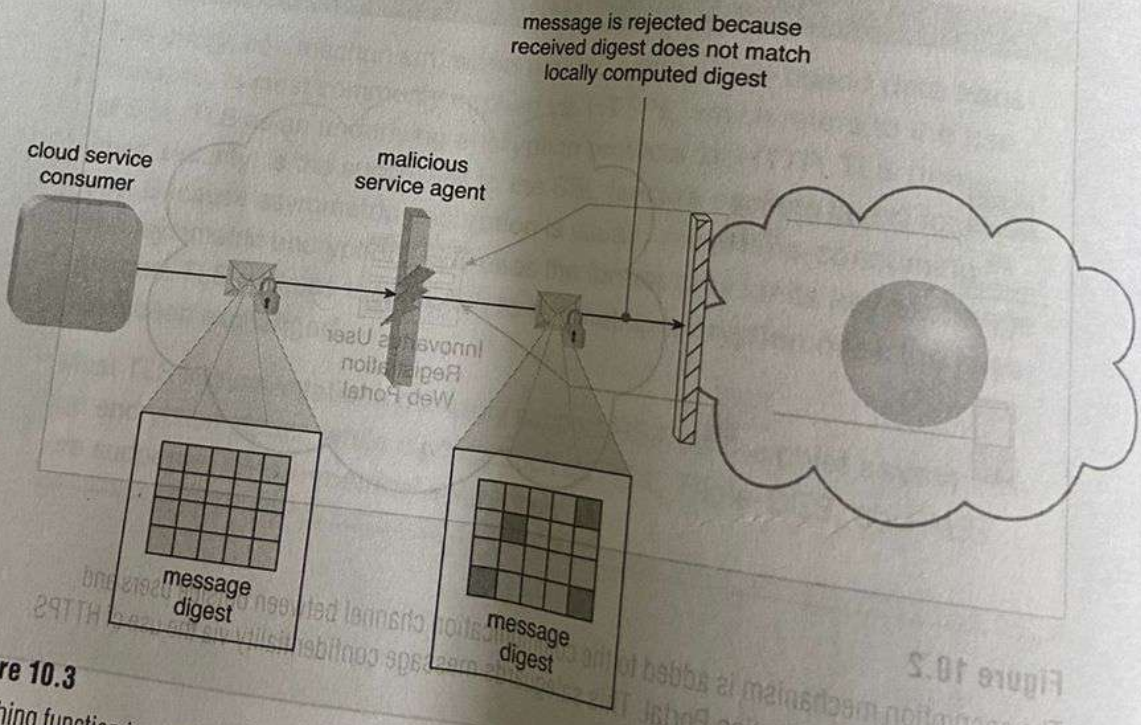

Figure 10.3

A hashing function is applied to protect the integrity of a message that is intercepted and altered by a malicious service agent, before it is forwarded. The firewall can be configured to determine that the message has been altered, thereby enabling it to reject the message before it can proceed to the cloud service.

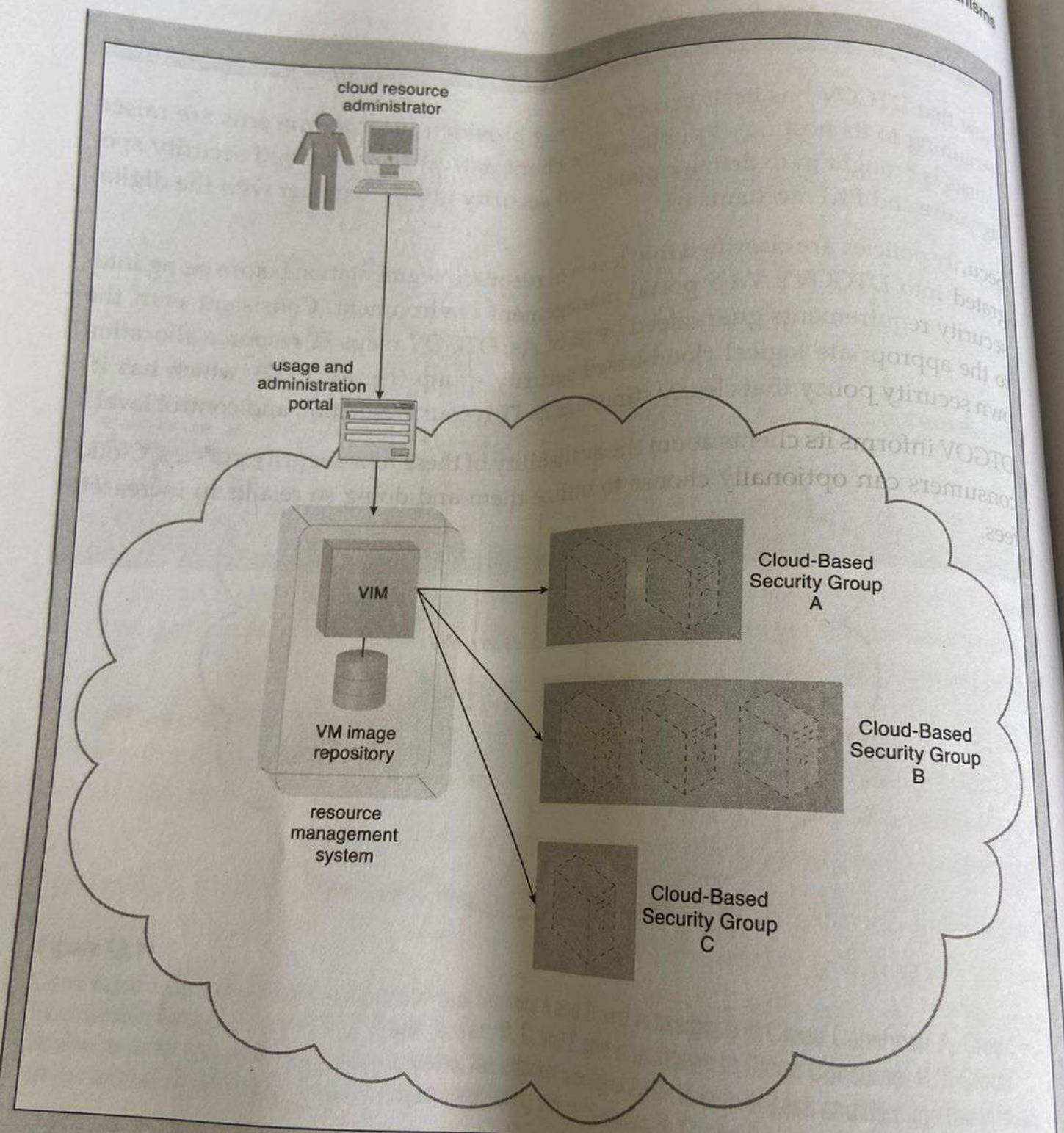

Figure 10.12

When an external cloud resource administrator accesses the Web portal to allocate a virtual server, the requested security credentials are assessed and mapped to an internal security policy that assigns a corresponding cloud-based security group to the new virtual server.

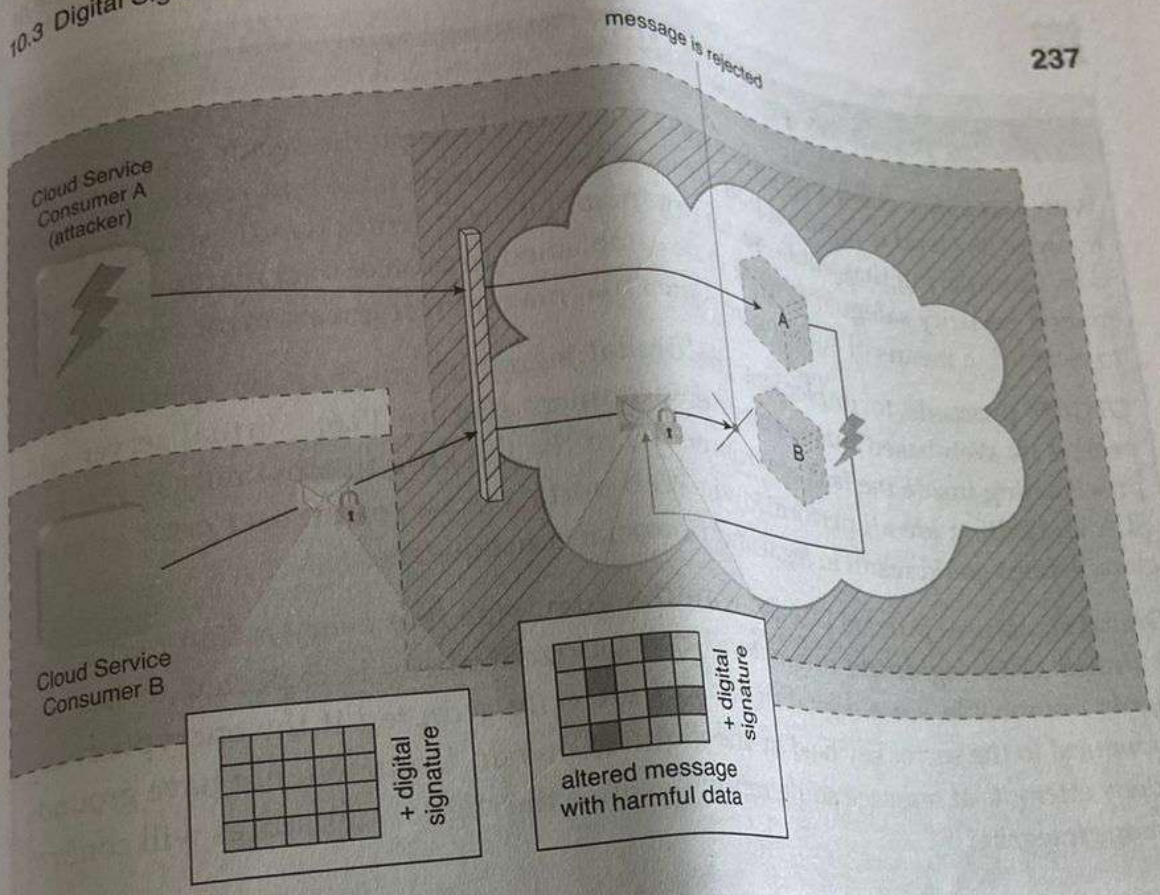

Figure 10.5
Cloud Service Consumer B sends a message that was digitally signed but was altered by trusted attacker Cloud Service Consumer A. Virtual Server B is configured to verify digital signatures before processing incoming messages even if they are within its trust boundary. The message is revealed as illegitimate due to its invalid digital signature, and is therefore rejected by Virtual Server B.

10.3 Digital Signature

The *digital signature* mechanism is a means of providing data authenticity and integrity through authentication and non-repudiation. A message is assigned a digital signature prior to transmission, which is then rendered invalid if the message experiences any subsequent, unauthorized modifications. A digital signature provides evidence that the message received is the same as the one created by its rightful sender.

Both hashing and asymmetrical encryption are involved in the creation of a digital signature, which essentially exists as a message digest that was encrypted by a private key and appended to the original message. The recipient verifies the signature validity and uses the corresponding public key to decrypt the digital signature, which produces the message digest. The hashing mechanism can also be applied to the original message to produce this message digest. Identical results from the two different processes indicate that the message maintained its integrity.

The digital signature mechanism helps mitigate the malicious intermediary, insufficient authorization, and overlapping trust boundaries security threats (Figure 10.5).

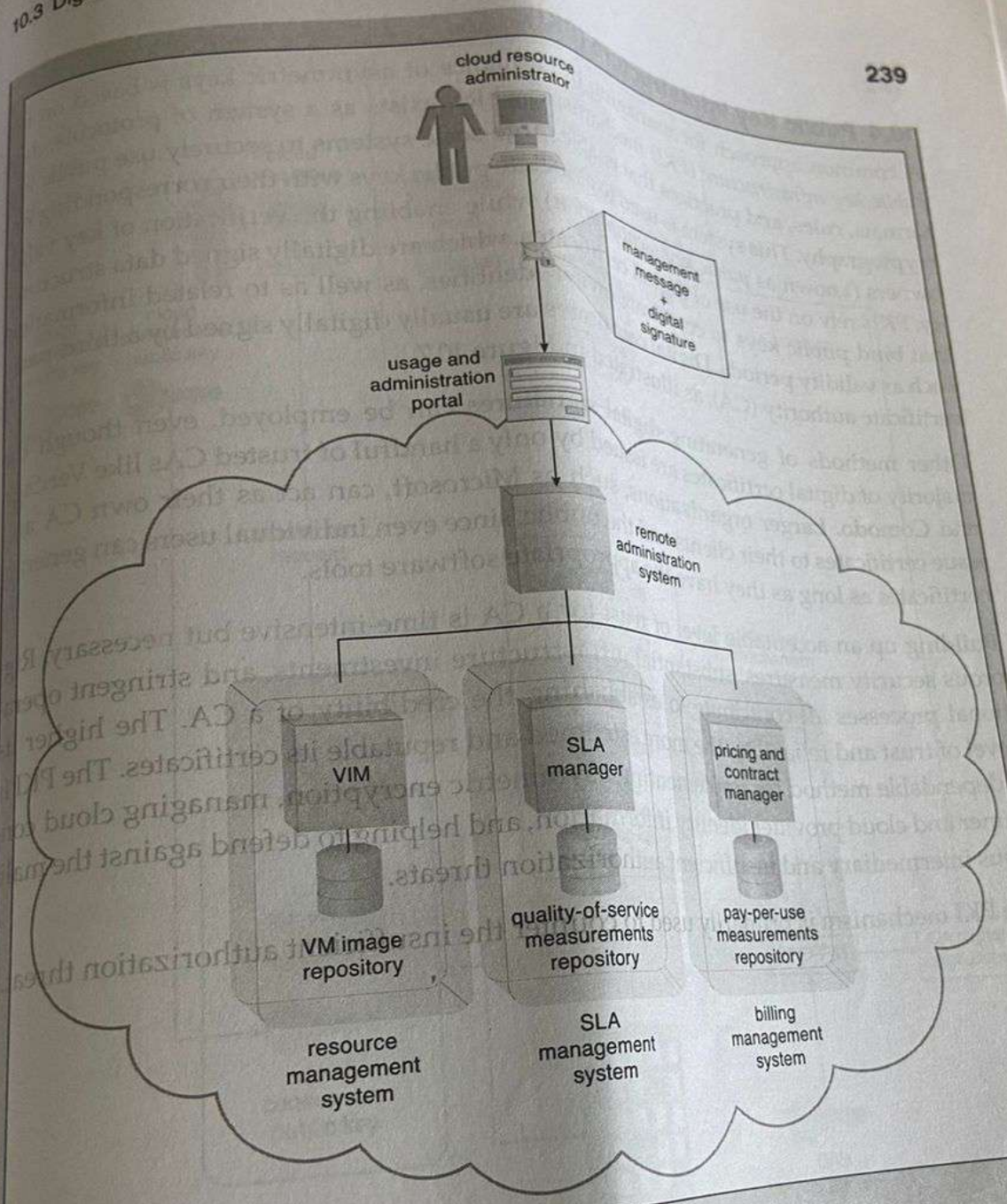

Figure 10.6
Whenever a cloud consumer performs a management action that is related to IT resources provisioned by DTGOV, the cloud service consumer program must include a digital signature in the message request to prove the legitimacy of its user.

CASE STUDY EXAMPLE

With DTGOV's client portfolio expanding to include public-sector organizations, many of its cloud computing policies have become unsuitable and require modification. Considering that public-sector organizations frequently handle strategic information, security safeguards need to be established to protect data manipulation and to establish a means of auditing activities that may impact government operations.

DTGOV proceeds to implement the digital signature mechanism specifically to protect its Web-based management environment (Figure 10.6). Virtual server provisioning inside the IaaS environment and the tracking functionality of real-time SLA and billing are all performed via Web portals. As a result, user error or malicious actions could result in legal and financial consequences.

Digital signatures provide DTGOV with the guarantee that every action performed is linked to its legitimate originator. Unauthorized access is expected to become highly improbable, since digital signatures are only accepted if the encryption key is identical to the secret key held by the rightful owner. Users will not have grounds to deny attempts at message adulteration because the digital signatures will confirm message integrity.

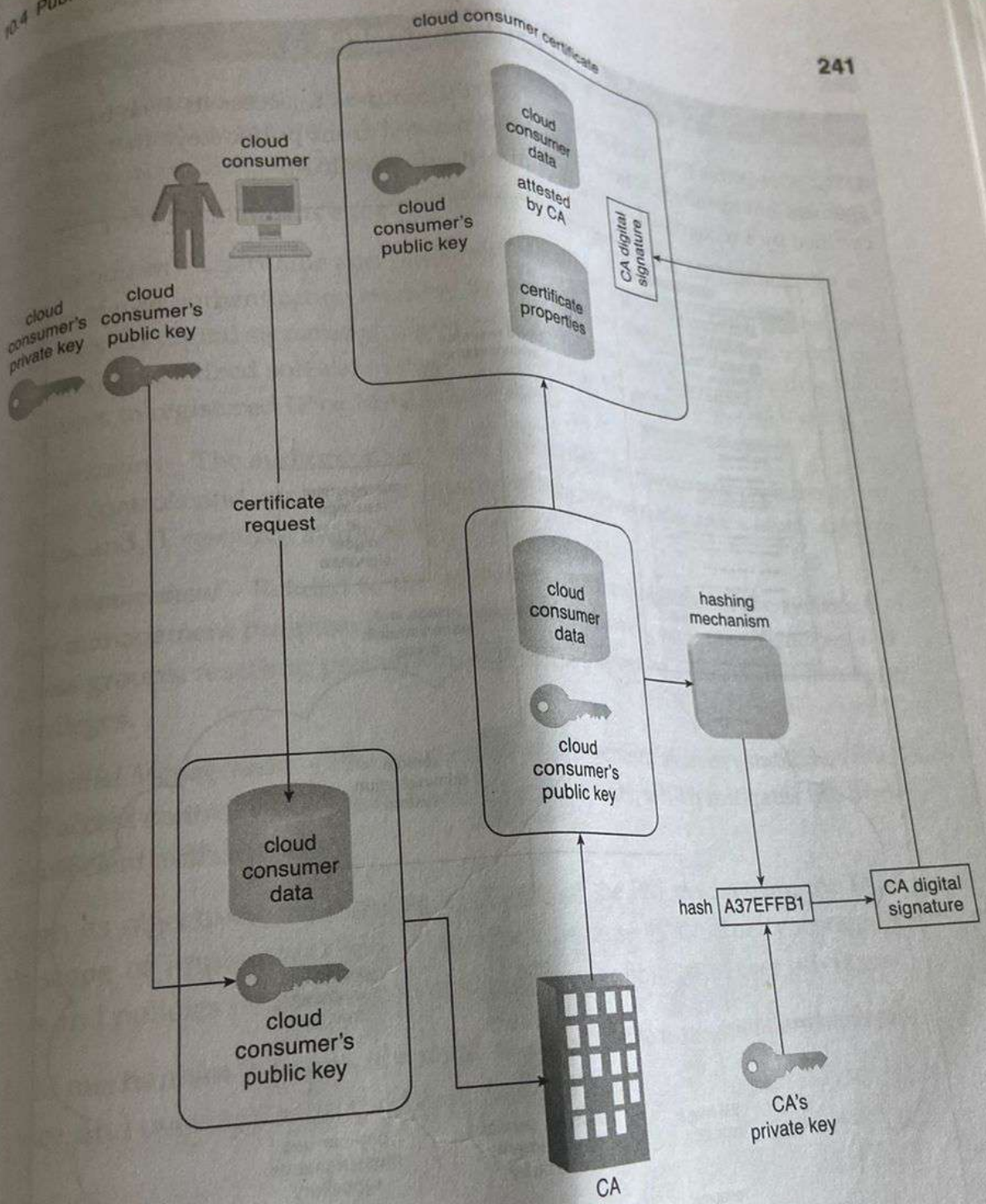

Figure 10.7
The common steps involved during the generation of certificates by a certificate authority.

10.4 Public Key Infrastructure (PKI)

A common approach for managing the issuance of asymmetric keys is based on the *public key infrastructure (PKI)* mechanism, which exists as a system of protocols, data formats, rules, and practices that enable large-scale systems to securely use public key cryptography. This system is used to associate public keys with their corresponding key owners (known as *public key identification*) while enabling the verification of key validity. PKIs rely on the use of digital certificates, which are digitally signed data structures that bind public keys to certificate owner identities, as well as to related information, such as validity periods. Digital certificates are usually digitally signed by a third-party certificate authority (CA), as illustrated in Figure 10.7.

Other methods of generating digital signatures can be employed, even though the majority of digital certificates are issued by only a handful of trusted CAs like VeriSign and Comodo. Larger organizations, such as Microsoft, can act as their own CA and issue certificates to their clients and the public, since even individual users can generate certificates as long as they have the appropriate software tools.

Building up an acceptable level of trust for a CA is time-intensive but necessary. Rigorous security measures, substantial infrastructure investments, and stringent operational processes all contribute to establishing the credibility of a CA. The higher the level of trust and reliability, the more esteemed and reputable its certificates. The PKI is a dependable method for implementing asymmetric encryption, managing cloud consumer and cloud provider identity information, and helping to defend against the malicious intermediary and insufficient authorization threats.

The PKI mechanism is primarily used to counter the insufficient authorization thre

10.5 Identity and Access Management (IAM)

The identity and access management (IAM) mechanism encompasses the components and policies necessary to control and track user identities and access privileges for IT resources, environments, and systems.

Specifically, IAM mechanisms exist as systems comprised of four main components:

- *Authentication* – Username and password combinations remain the most common forms of user authentication credentials managed by the IAM system, which also can support digital signatures, digital certificates, biometric hardware (fingerprint readers), specialized software (such as voice analysis programs), and locking user accounts to registered IP or MAC addresses.
- *Authorization* – The authorization component defines the correct granularity for access controls and oversees the relationships between identities, access control rights, and IT resource availability.
- *User Management* – Related to the administrative capabilities of the system, the user management program is responsible for creating new user identities and access groups, resetting passwords, defining password policies, and managing privileges.
- *Credential Management* – The credential management system establishes identities and access control rules for defined user accounts, which mitigates the threat of insufficient authorization.

Although its objectives are similar to those of the PKI mechanism, the IAM mechanism's scope of implementation is distinct because its structure encompasses access controls and policies in addition to assigning specific levels of user privileges.

The IAM mechanism is primarily used to counter the insufficient authorization, denial of service, and overlapping trust boundaries threats.

CASE STUDY EXAMPLE

DTGOV requires that its clients use digital signatures to access its Web-based management environment. These are to be generated from public keys that have been certified by a recognized certificate authority (Figure 10.8).

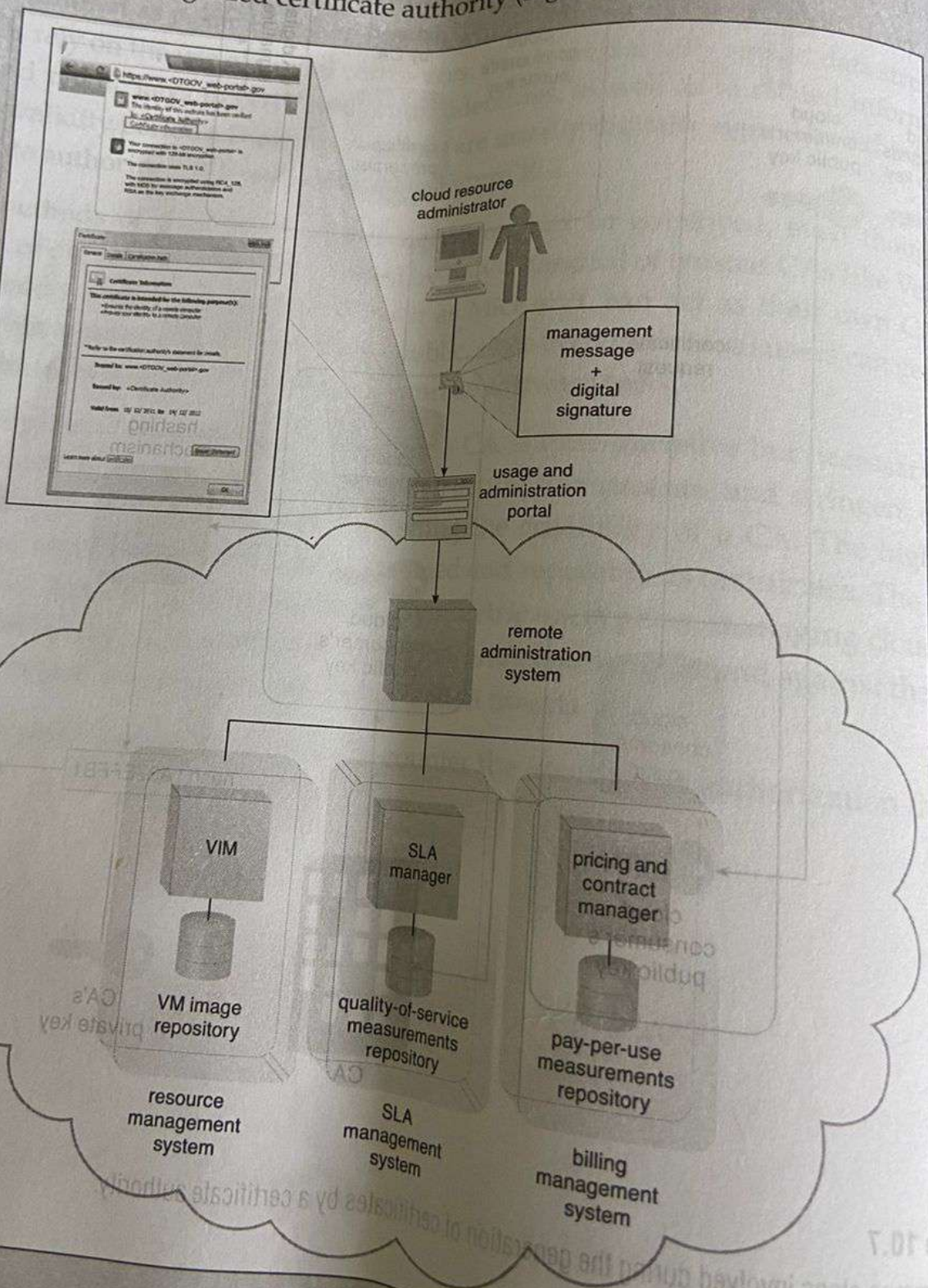

Figure 10.8

An external cloud resource administrator uses a digital [illegible] environment. DTGOV's [illegible]

10.7 Cloud-Based Security Groups

Similar to constructing dykes and levees that separate land from water, data protection is increased by placing barriers between IT resources. Cloud resource segmentation is a process by which separate physical and virtual IT environments are created for different users and groups. For example, an organization's WAN can be partitioned according to individual network security requirements. One network can be established with a resilient firewall for external Internet access, while a second is deployed without a firewall because its users are internal and unable to access the Internet.

Resource segmentation is used to enable virtualization by allocating a variety of physical IT resources to virtual machines. It needs to be optimized for public cloud environments, since organizational trust boundaries from different cloud consumers overlap when sharing the same underlying physical IT resources.

The cloud-based resource segmentation process creates *cloud-based security group* mechanisms that are determined through security policies. Networks are segmented into logical cloud-based security groups that form logical network perimeters. Each cloud-based IT resource is assigned to at least one logical cloud-based security group. Each logical cloud-based security group is assigned specific rules that govern the communication between the security groups.

Multiple virtual servers running on the same physical server can become members of different logical cloud-based security groups (Figure 10.11). Virtual servers can further be separated into public-private groups, development-production groups, or any other designation configured by the cloud resource administrator.

Cloud-based security groups delineate areas where different security measures can be applied. Properly implemented cloud-based security groups help limit unauthorized access to IT resources in the event of a security breach. This mechanism can be used to help counter the denial of service, insufficient authorization, and overlapping trust boundaries threats, and is closely related to the logical network perimeter mechanism.

CASE STUDY EXAMPLE

As a result of several past corporate acquisitions, ATN's legacy landscape has become complex and highly heterogeneous. Maintenance costs have increased due to redundant and similar applications and databases running concurrently. Legacy repositories of user credentials are just as assorted.

Now that ATN has ported several applications to a PaaS environment, new identities are created and configured in order to grant users access. The CloudEnhance consultants suggest that ATN capitalize on this opportunity by starting a pilot IAM system initiative, especially since a new group of cloud-based identities is needed.

ATN agrees, and a specialized IAM system is designed specifically to regulate the security boundaries within their new PaaS environment. With this system, the identities assigned to cloud-based IT resources differ from corresponding on-premise identities, which were originally defined according to ATN's internal security policies.

10.6 Single Sign-On (SSO)

Propagating the authentication and authorization information for a cloud service consumer across multiple cloud services can be a challenge, especially if numerous cloud services or cloud-based IT resources need to be invoked as part of the same overall runtime activity. The *single sign-on (SSO)* mechanism enables one cloud service consumer to be authenticated by a security broker, which establishes a security context that is persisted while the cloud service consumer accesses other cloud services or cloud-based IT resources. Otherwise, the cloud service consumer would need to re-authenticate itself with every subsequent request.

The SSO mechanism essentially enables mutually independent cloud services and IT resources to generate and circulate runtime authentication and authorization credentials. The credentials initially provided by the cloud service consumer remain valid for the duration of a session, while its security context information is shared (Figure 10.9). The SSO mechanism's security broker is especially useful when a cloud service consumer needs to access cloud services residing on different clouds (Figure 10.10).

This mechanism does not directly counter any of the [illegible] Chapter 6. It primarily enhances the usabili[illegible] and management of [illegible]

CASE STUDY EXAMPLE

Now that DTGOV has itself become a cloud provider, security concerns are raised pertaining to its hosting of public-sector client data. A team of cloud security specialists is brought in to define cloud-based security groups together with the digital signature and PKI mechanisms.

Security policies are classified into levels of resource segmentation before being integrated into DTGOV's Web portal management environment. Consistent with the security requirements guaranteed by its SLAs, DTGOV maps IT resource allocation to the appropriate logical cloud-based security group (Figure 10.12), which has its own security policy that clearly stipulates its IT resource isolation and control levels.

DTGOV informs its clients about the availability of these new security policies. Cloud consumers can optionally choose to utilize them and doing so results in increased fees.

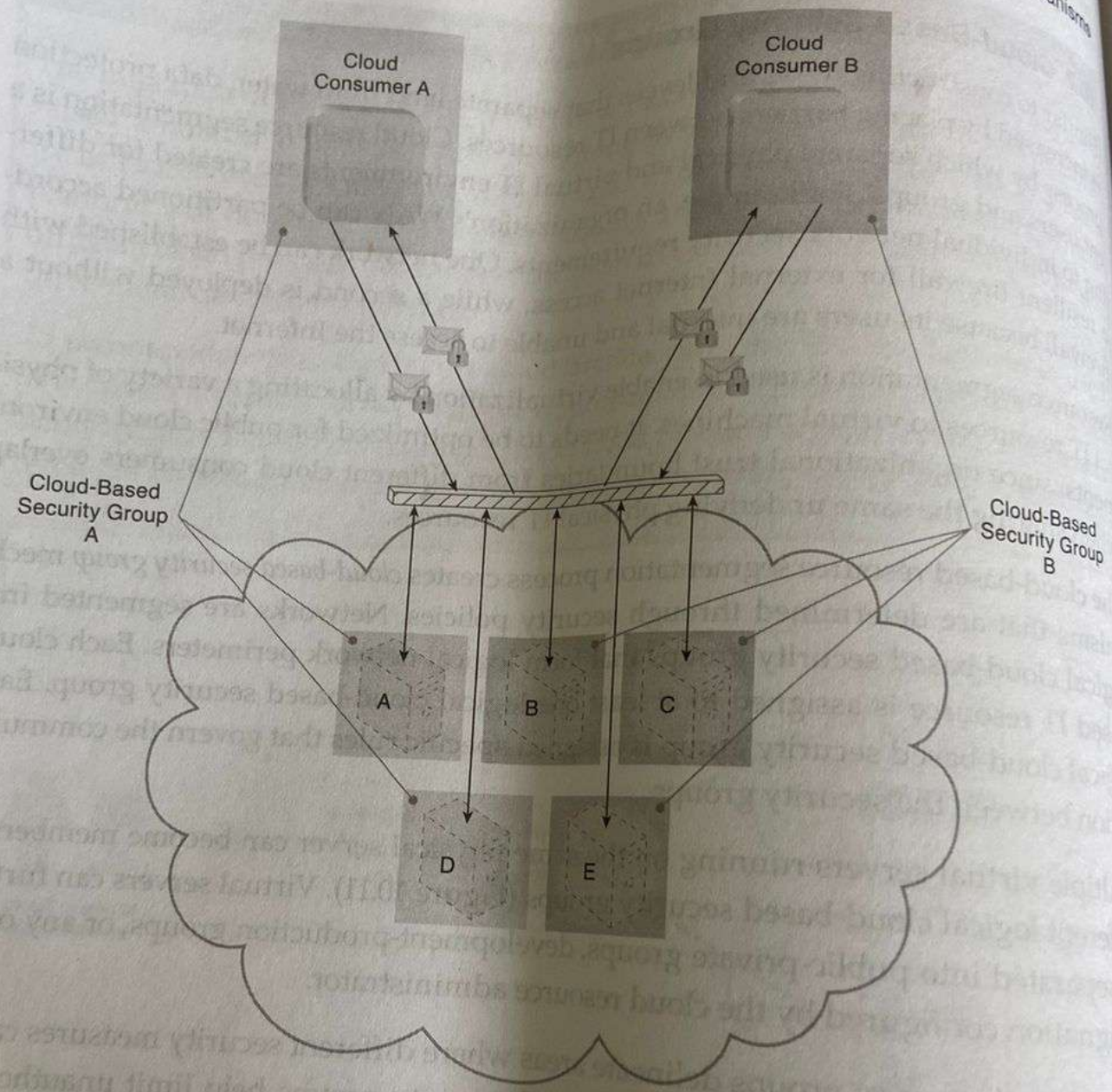

Figure 10.11

Cloud-Based Security Group A encompasses Virtual Servers A and D and is assigned to Cloud Consumer A. Cloud-Based Security Group B is comprised of Virtual Servers B, C, and E and is assigned to Cloud Consumer B. If Cloud Service Consumer A's credentials are compromised, the attacker would only be able to access and damage the virtual servers in Cloud-Based Security Group A, thereby protecting Virtual Servers B, C, and E.