Identity and Access Management

IAM is an integral part of most IT risk management efforts

Study FISCAM Table 3
p 270-2

College of Business

IT Auditing

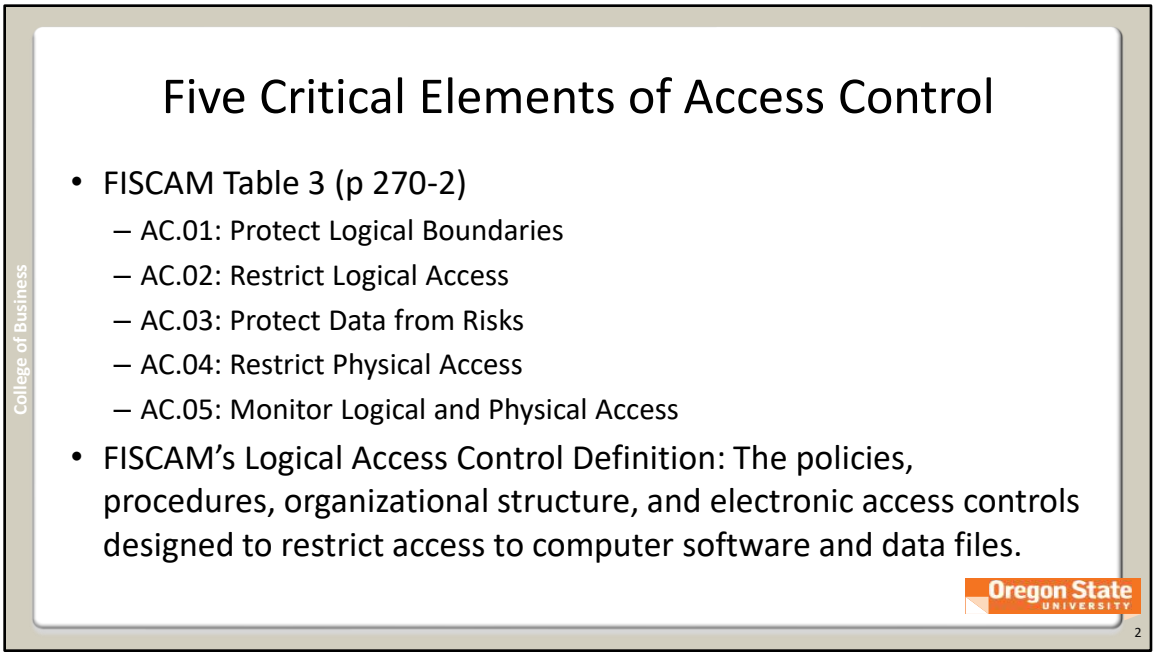
Oregon State University

Access control reviews are a major component of IT auditing and are fundamental to most IT risk management processes. You simply cannot secure IT systems without controlling who can access and change them.

Cyber security or IT risk management tools all have a lot to say about Identity and Access Management.

Fundamentally, if the systems which manage operational and financial data, and the accounts used to perform management operations are not appropriately safeguarded, any actions, decisions, or communications based on generated reports is unreliable.

And, perhaps more importantly, in today's digitized world, operations are heavily reliant on IAM. Systems aren't secure without solid IAM practices. And unprotected systems means an organization will be at high risk of not accomplishing their objectives.



FISCAM groups together critical elements of access control as listed in Table 3 on pp 270-2 and 270-3.

Information systems are made up of parts that form sub-groups. You could say that items that are logically similar or that interact are within the same "boundary." In some senses, boundaries are disappearing as things become increasingly digitally interconnected. What FISCAM is getting at is that organizations organize things into "bounded" groups to protect them. Any given component can be identified as either in or not in the group or boundary.

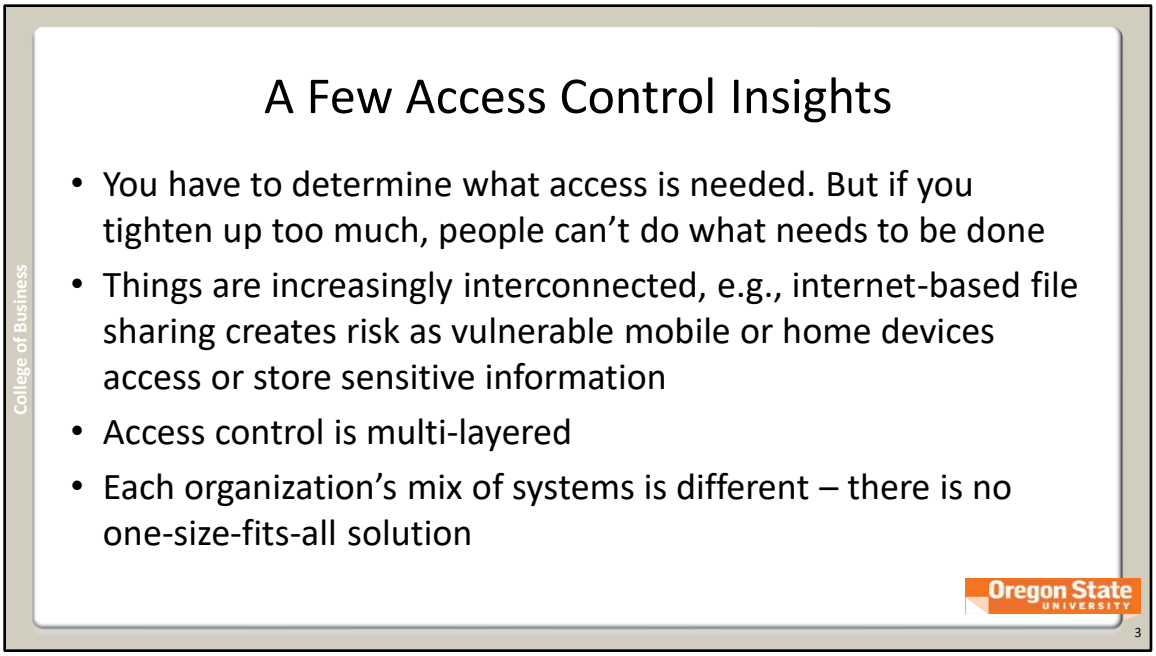
Note that both logical and physical organization matters. All the gear in a data center/server room or all the computers connected on the on-campus network are within a physical boundary. But often, not all the systems that access the ERP system are in the same city. We group people by their job roles and services by the functions or data they interact with forming "logical" boundaries.

So, don't take the boundary idea too seriously, its just a way of saying we need to identify groups of components that are, in some sense, similarly protectable.

For now, we will focus on two of the stated control objectives from FISCAM:
The two control objectives listed in FISCAM Table 3 (on pages 270-2 and 270-3) stand out.

- AC.02.02: Information system users, processes, and services are appropriately identified and authenticated before accessing information systems and information system resources.
- AC.02.04 Access privileges restrict access to information system resources to authorized individuals for authorized purposes.

You should remember from previous units that control objectives can be converted in audit objectives. Can you restate one of those control objectives into something an auditor could investigate? Of course that might be too detailed. It may be more likely that the critical elements would become audit objectives. Or, even more likely, all of FISCAM's Table 3 would be combined in a single audit objective. But the relationship holds: control objectives form the basis of audit objectives for an internal control audit. That makes FISAM really useful for audit planning.



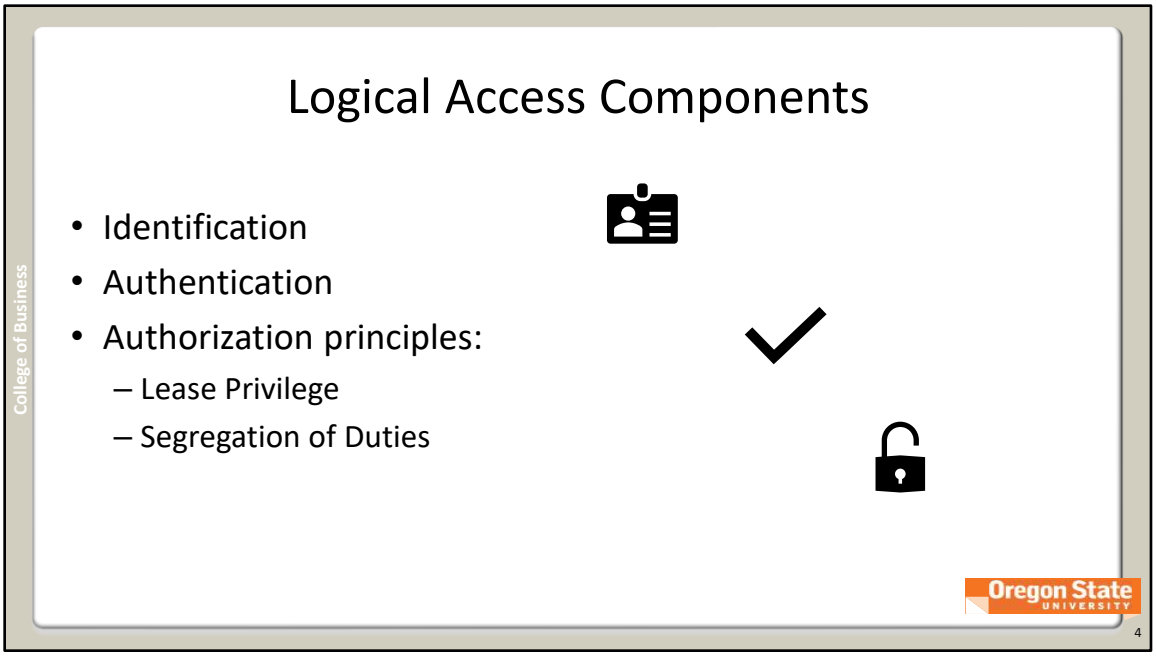
If you are a Harry Potter fan, you may wonder whether Alastor Moody or only Barty Crouch the younger was known to say that "constant vigilance" was important to protect against evil.
Or perhaps your understanding of US history resonates with the saying "Eternal vigilance is the price of liberty." That notion is considered to have been important in the thinking of Thomas Jefferson and other thought leaders in the revolutionary movements of his day.
Different governments around the world take different stances on how much data should be made available and by what means such access should be controlled.

The ethical questions that can arise while applying or not applying these principles are very real for the field of governance risk and compliance. There are good arguments to be made here. How much monitoring is too much monitoring? What rights does an organization (or society, or individual) have in protecting the privacy of "their" data from others? And, perhaps more practically, there are the more mundane risk analyses that considers the cost/benefit balance of various possible protections.

In any case, it is clear that information technology cannot possibly be secure without careful control of:
- Identity – a validated and nuanced understanding of who is using an information resource
- Authentication – a verification that a user is who they say they are, and
- Authorization – plans and tools that limit access to protection-worthy information

These concepts constitute the core of the conceptual substance of Identity and Access Management (IAM) processes.

Be sure you read over the points on the slide. These ideas may seem obvious, but they are not so easy to implement.



To make sense of information in authoritative sources, you need to have a few key ideas clear up front.

FISCAM's glossary is a good place to start:
Identification is: *The process of verifying the identity of a user, process, or device, usually as a prerequisite for granting access to resources in an information system.*
Identification processes assign unique identifiers to known entities. We can't secure things without knowing identifying actors.

Authentication is *verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.*
Once devices have names, they need to be authenticated so that the system knows the person, process, or device is authentically who they say they are.

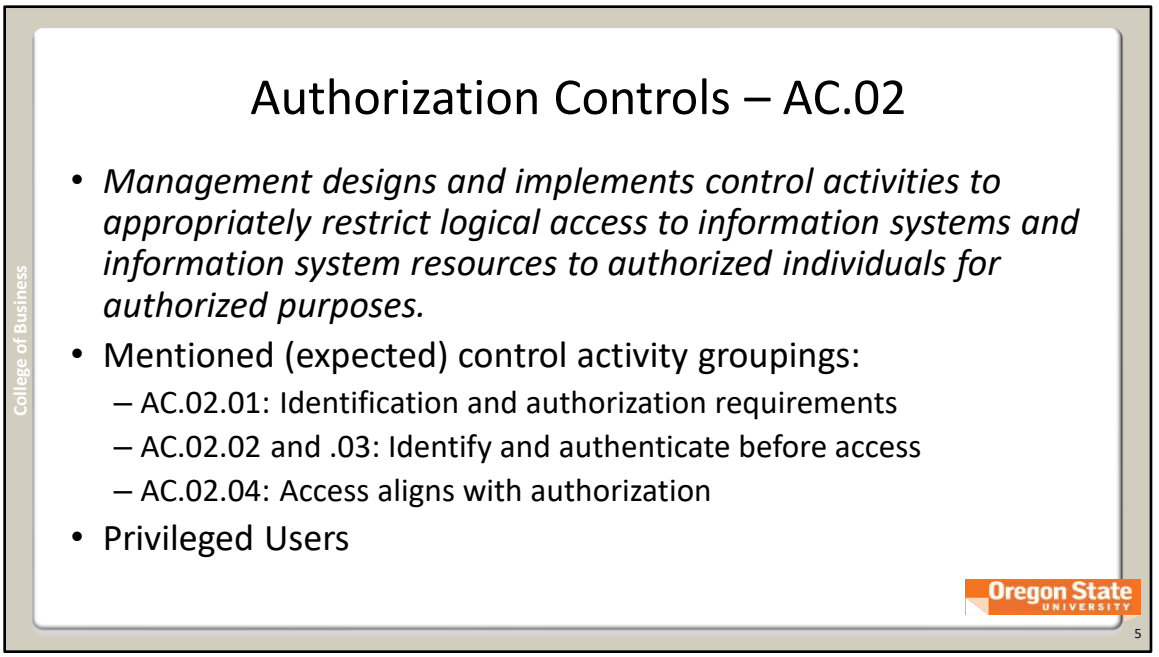
The definition of authorization in the FISCAM glossary is about the legal basis to operate as system. Such legalities have their place in IT audit. But in an IAM context, authorization is a combination of logical/abstract "permission to access" and some enforcement mechanism that ensures only "authorized" access is allowed in a system.

Logical authorization rests on two principles:
- Least Privilege: *The principle that a security architecture should be designed so that each entity is granted the minimum system resources and authorizations that the entity needs to perform its function.*
- *Segregation (SoD) of duties involves segregating* [dividing] *work responsibilities so that one individual does not control all critical stages of a process.*

Take note that in the IT audit context, segregation of duties is largely about system administrators and security managers. Separating invoice approval from payment authorization is one kind of SoD, but it is an application control. IT auditors often focus on general IT risks across multiple applications. For example, it one person can both change programs and them into production, there is risk that carelessness or malfeasance can result in error or disruption. If a system's SoD parameters prevent one person doing both things, some risks are reduced.

Generally, systems store managed lists of authorizations, that connect to identified users, devices, or services to specific activities. System functions check the list before allowing electronic access.

Together, these ideas form the basis for logical access controls.

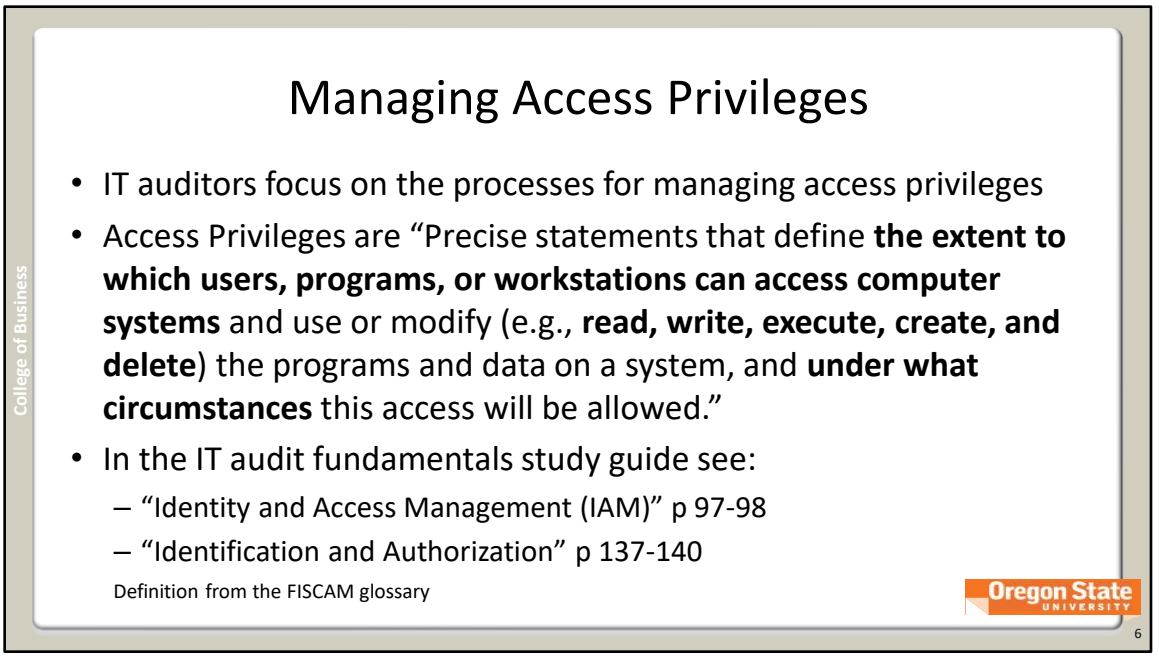


FISCAMs appendix 500B has several very large tables listing elements of the "FISCAM Framework". The pdf is searchable to support navigation. Table 10 (starting on page 500B-132) specifies the FISCAM framework for access controls.

One of the most important elements in Access control relates to restricting activities to "authorized individuals for authorized purposes." Exemplary control activities, audit procedures, and criteria sources for FISCAM "critical element" AC.02 are included in Table 10 and run from page 500B-141 – 500B-181. Whew! Lots to think about, but also lots of solid content an auditor can use in planning, carrying out, and reporting on an audit.

FISCAM's glossary defines a privileged users as "*A user who is authorized (and therefore trusted) to perform security-relevant functions that ordinary users are not authorized to perform.*"

Privileged access, that is access to systems by IT administrators and other super-users charged with managing systems, presents special problems. Admins need to be allowed to do most anything to the system or else they can't keep the system working. But if they can "do anything" they can probably make errant or nefarious changes and may also be able to hide their activity. Special precautions are needed for such accounts. Compromising such accounts is a key step in many cyber attack scenarios and when such accounts are compromised, damage can be swift and far reaching.



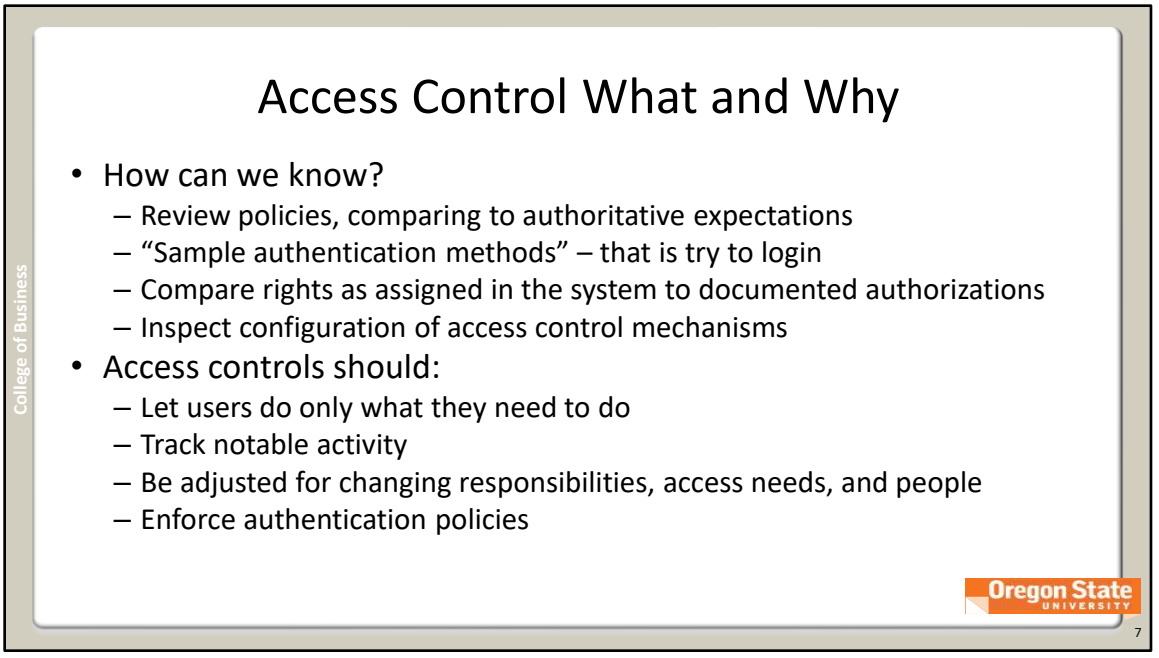
IT auditors don't have to be able to perform the tasks they audit. And they don't have to be able to be certain that the controls they audit are foolproof. Audit objectives aim to provide reasonable assurance that stakeholders can be reasonable assured that control objectives will be met.

Those four categories: read, write, execute, and delete closely map to how operating systems and application programs enforce authorization intentions. But take note: in many ways, the ability to "write" can amount to the ability to create or delete. To illustrate: what if an administrator can access a user account, change the email address, change the password, and change other settings? They have effectively deleted one user and created another. We could ask a similar question about altering firewall rules, programs, or system files.

It is not enough for an organization to currently have only appropriate authorizations in place. The auditor is looking for evidence that the processes used to managed access privileges is well designed and properly operated.

Chapter 3 of the ISACA IT Audit Fundamentals study guide (p 97-98):
- Talks about IAM generally
- Identifies common vulnerabilities
- Notes a few things an auditor should do when auditing IAM

Chapter 4 (p 137-140) includes more details:
- Common security gaps
- Suggested good control practices
- High level auditing steps



IT Auditors will encounter a wide variety of access control processes and mechanisms. Organizations are increasingly using integrated identity governance and administration (IGA) tools to help coordinate authorizations, but some tools will, and should, operate outside the bounds of such a system. For example, an organization may mostly use a cloud based active directory to manage most user rights, but assign right to manage cloud infrastructure or network gear separately. Auditors start by understanding the identity architecture.

The basics are clear. The auditor needs to investigate:
- How access is supposed to be authorized
- How authorizations (and authorization changes) are recorded
- How authentications and authorizations are enforced in real time, and
- How authorization and authentications are secured

As they investigate, they look for well known areas of control weakness in the design of the controls.

The least privilege principle applies here. Don't give the user more rights than they need. Of course, that can be really hard to do without either spending way too much money on detailed rights assignments and monitoring, or else seriously impairing operations by not allowing people to do things.

Space and time do not allow for organizations to track all activity. But creating access logs is always needed. Who tried to log in when? Did they succeed? If this is a privileged account, what was accessed? What changes in access rights were made by whom and when? Appropriate levels of tracking needs to be determined for each organization because organizations are not all the same. Costs and perceived benefits are weighed against each other. Because real cyber threats are relatively rare compared to regular activity, it is easy to ignore events that indicate risk.

When people change roles, their access rights probably need to change. But organizations often want to leave old rights in place for a while to ease transitions or support coverage of important functions. Unfortunately, its easy to forget to go back and correct things later.

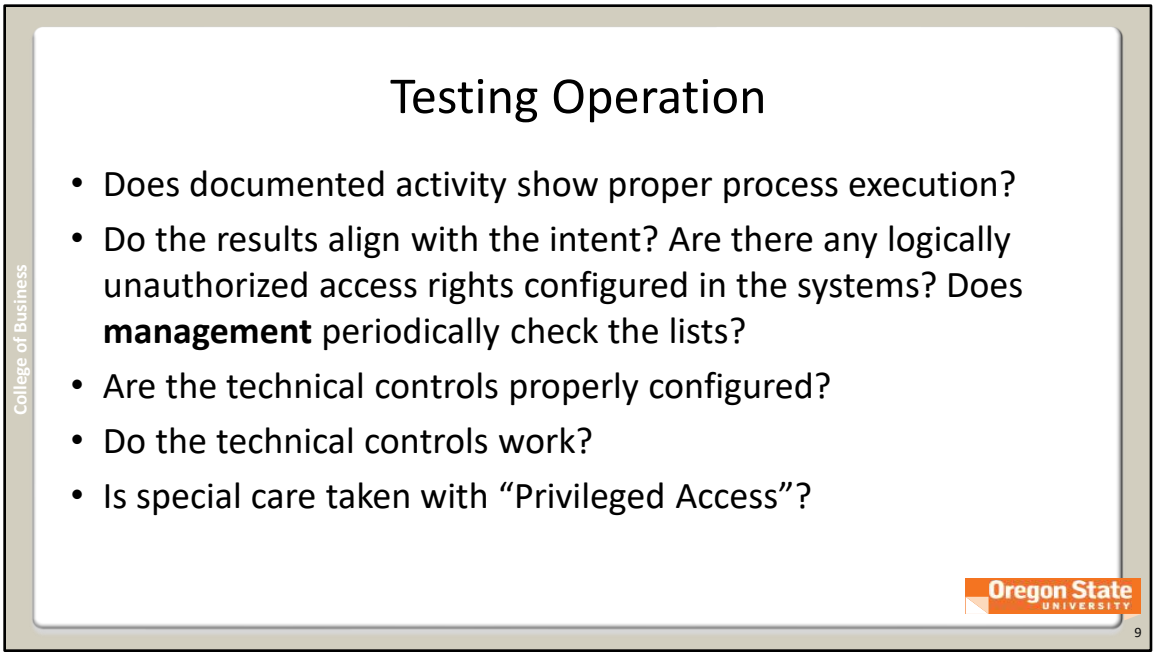
Identification and authentication are not the same thing. Simplistically, an entity on the system identifies itself by "saying" its name. But we only recognize "known" entities and we don't allow anyone to do things based on a username alone. Passwords and other "proofs" are required for authentication.

Identity is about making sure a person, process, or device is authentic even before access is requested. Managers verify new people, devices, and services before accounts are created or added to inventory as part of a provisioning process. People show their driver's license or passport to prove they are who they say they are before accounts are enabled. Secret cryptographic key are often employed to uniquely identify known/authorized devices and services. Systems block access requests from unknown users, devices, or processes. Guest accounts are discouraged, even if we want to allow guest access, we want to know who is doing what to track possible attacks.

Identified entities are authenticated when they request to access resources. The reality is that passwords – by themselves – are not sufficient to protect important computing resources. Multifactor authentication requires that you include elements from more than one of three authentication "factors": What you know, What you have, or What you are. Two passwords is not two-factor. A text message system (based on having a device in hand) and a security key (another device) is not two factor. In general, employing the same factor twice adds little to security. If they stole your phone, they could well have stolen your security key too. But with two factor, they ALSO need to know your password. Passwords are stored in your head - or maybe stored in a password manager.

Requiring multiple factors makes it harder for an attacker. For example, some bad actors hang around in bars and trick people into unlocking their phone using their pin – which is a password. Once they have the pin (password), they steal the phone and empty the target's bank account. This is one example of how two-factor is not entirely secure. But it is way more secure than one factor. If the attacker needed only the phone or only the password, it would be a lot simpler to carry out an effective attack.
When we talk about various multifactor authentication setups think about which two factors are required: something you know, something you have, or something that you are. If two categories are not involved, it is not multifactor.

Industry is moving towards pass keys that do not require a user to provide a password. We could debate how secure that really is across various situations. In any case, FISCAM demonstrates a good understanding in its glossary when it defines a multifactor authenticator as: *An authenticator that provides more than one distinct authentication factor, such as a cryptographic authentication device with an integrated biometric sensor*. Do you see how this kind "authenticator" – which can be implemented on a phone, uses "what you have" – the phone – and "what you are" – your fingerprint. Two factors! Its probably harder for the bad person in the bar to get you to put your finger on the phone on the way out the door!



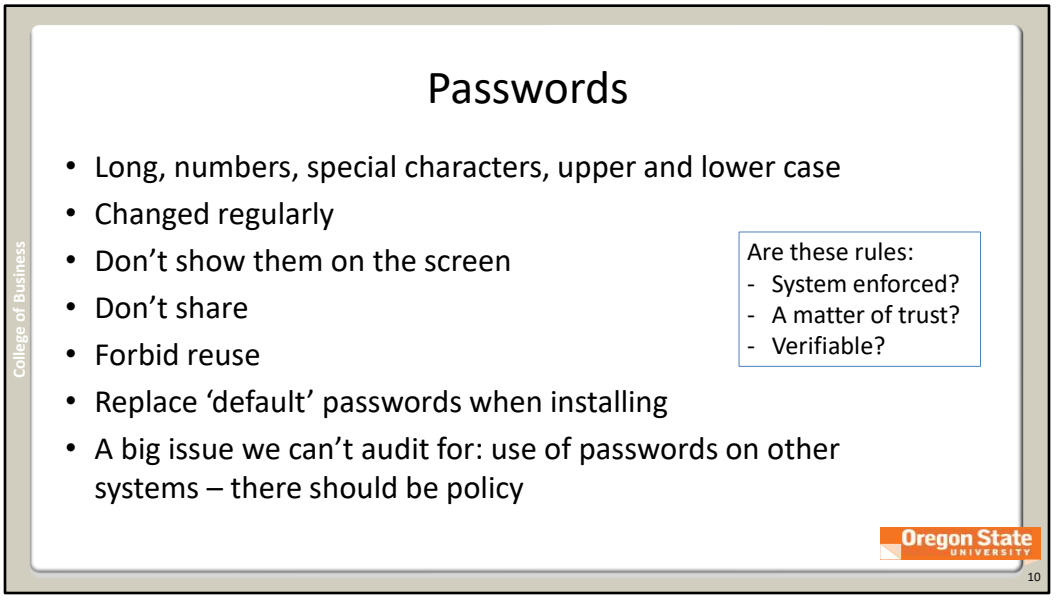
Previous slides talk about how things should work – controls need to be designed for success. But control operation and effectiveness are also audited. Auditors should pay special attention to privileged accounts because of the extra risk they pose.

Does documented activity show proper process execution?
Are authorizations removed per policy? Are rights added without proper and documented approval? Frameworks call for processes that ensure only authorized rights are added. But if an organization is to be assured the safeguards are working, authorization processes need to specify how authorizations are requested, approved, and documented. How do we know the manager approved? Perhaps from a memo or help desk ticket. Do the records reflect that the intended process is in operation?

Do the results align with the intent? Are there any logically unauthorized access rights configured in the systems?
Over time, various issues (errors, fraud, changing conditions, personnel changes) result in lingering authorizations. By policy, a person should no longer have access, but in practice, the computerized access control lists may not have been updated correctly. Comparing and fixing any variances is primarily a control activity, not an audit procedure. Management should regularly review access lists to see if they still align with intended policy. The auditor may 'reperform' the comparison to verify its effectiveness, but the auditor will mostly be looking for evidence that management regularly reviews the list and fixes identified mis-allocations.

Are the technical controls properly configured?
Real-time authentication tools verify authentication factors and log logins. But neglect, error, attack, or malfeasance can result in mis-configurations. Is multifactor authentication enabled per policy and are allowed exceptions documented and reviewed? Are password settings correct in operating systems, directory services, and applications? Are complete logs generated and protected. Attackers should be prevented from removing log entries. Weak configuration can mean logs are unavailable to investigate security events. Auditors check to see if systems are configured according to policy or good practice recommendations to reduce risk.

Do the technical controls work?
Auditors spend their best efforts on process and control implementation because those are what ensure protection over time. But direct tests that check to see if things are working are also important. Auditors sometimes test by attempting to access resources using unauthorized accounts or credentials. This can take several forms. The auditor might ask the client to demonstrate that logins from expired or disable accounts is blocked. They might perform sophisticated penetration tests with the knowledge and approval of the client. And access logs can be reviewed to see if any old accounts accessed resources after they should have been disabled.



Credentials can be hardened against attack, but details can be controversial. Systems can only automatically enforce some of these policies because passwords are encrypted before they are stored. Enforcement is a matter of trust for some protections.

Strong passwords that are long, and include numbers, upper- and lower- case letters, and special characters are harder to guess. If a system's password store (encrypted list of usernames and passwords) is obtained or can be repeatedly challenged, passwords can be "cracked." It is better if trillions of guesses are needed rather than mere millions. And many authoritative resources call for regular password changes. Because strong passwords or changed passwords can be difficult to remember, people:

- Write them down where someone where others might be able to see them,
- Forget them, requiring password reset mechanisms which can be misused, and
- Reuse them on multiple sites. As a result, compromises on one system can lead to compromises on another.

Good policies forbid the use of names, certain words, or dates within a password. Some hacking tools employ dictionary attacks. The hacker tries passwords with common words first. That can make it much easier to break into a system. Simplistically, the password CrazyPassword is much easier to guess than Crzy!Password because Crazy might be in a dictionary while Crzy! is not. A brute force attack would try a huge number of somewhat random variations before they would get around to Crzy!Password.

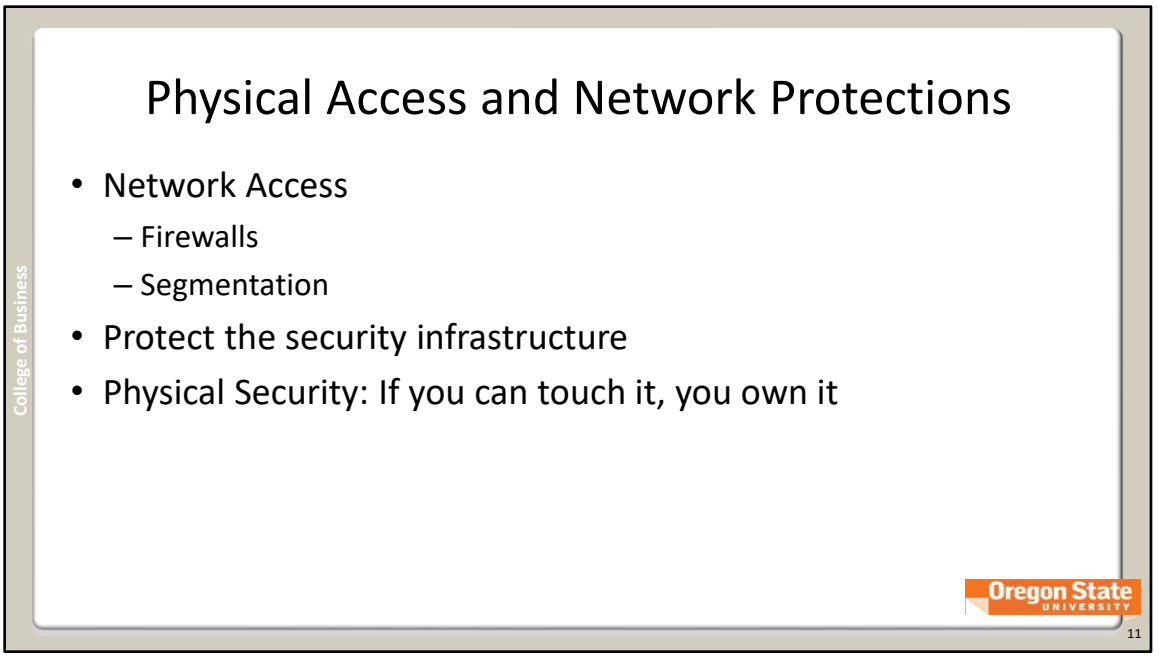
The value of changing passwords is arguable. People do risky things like add a number at the end. If an attacker comes to know that a user's current password is MyCred4 its not a stretch to guess that MyCred5 will be next after a forced password change.

Perhaps, in practice, changing passwords is marginally effective or even counter productive. But if users will act in good faith, making substantially different passwords and handling them in recommended ways, password changes do increase security. Attacker access can "dwell" in a system for months. Changing a password can eventually block use. And if a user reuses their organizational password across systems, say on some website on the internet, changing can help in the case that that website's password list is stolen and cracked. These things really happen and have, sometimes, been really helpful!

Identities should be associated with specific people and should not be shared. This reduces risks as people come and go and enables improved investigation in the case of a security event. But people do sometimes share their password to allow someone to fill in for them when they are gone. That is a signal that authentication and authorization procedures need to be improved.

Failing to change detail passwords creates is a big risk. Purchased programs or devices often start up with published passwords to enable setup. But if that default password is not removed or changed, and attacker can look it up on the internet and run amok.

Auditors should be aware of all these considerations in evaluating the design and operational effectiveness of password controls. System rules for length, strength, and change frequency can be configured in many operating systems and applications. Auditors can ask for screenshots of settings, use scripts to list configurations, and inspect logs that would show rule changes.



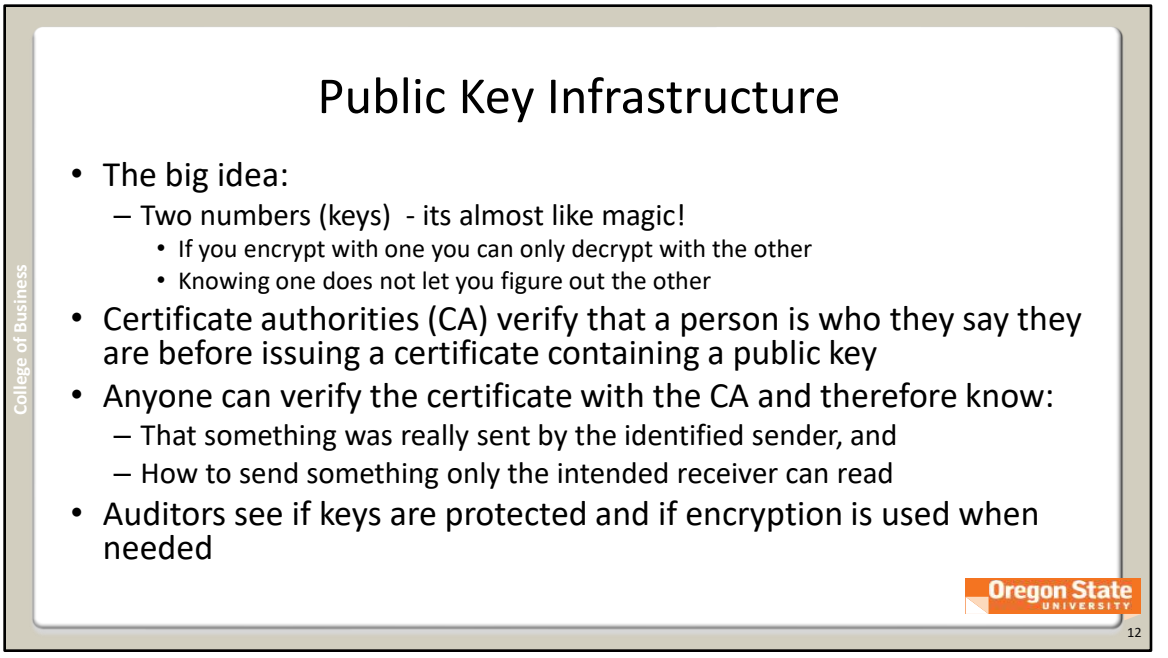
Let's consider a few more elements of access control.

Networks and systems have boundaries. The idea that we can "keep the baddies out" is important. If we don't allow, for example, a rogue computer to connect to the same network as the database server – enforcing the network boundary – it becomes much harder to compromise that valuable resource. This is a good consideration. But you should also realize that boundaries have become hard to enforce as wireless technology and ubiquitous internet access have become the norm.

The infrastructure that supports IAM tools need very strong protections. If encryption keys or other authentication or authorization system elements are compromised, security safeguards are nullified. Access to the configuration consoles for infrastructure is an issue. It is important to prevent attackers from reconfiguring routers, firewalls, directory resources, log servers, and other security-related components. Just as user authorizations should allow only needed access, networks should be configured to restrict traffic to and from infrastructure configuration consoles. Zero-day (previously unknown) flaws in infrastructure tools have resulted in billions of dollars of damage and disruption in recent years.

Network defenders use firewalls and network segmentation to block some unauthorized messages before they are even delivered to the computers or services used to manage networks or security elements. But great care has to be taken because message blocking can cause disruptions. For example, in the case of a natural disaster, remote access to servers and network gear may be needed. If controls are too tight, recovery efforts may be thwarted. Auditors will likely verify that clients have solid processes for managing these risks, but because the issues are highly technical, many audits will only consider the processes that manage these protections rather than the sufficiency of the implemented controls themselves. Still, specialized (and expensive) reviews of these protections are often in order.

Physical access generally allows a sophisticated actor to override IAM controls. If someone can plug a device into a computer, it is nearly impossible to keep them from obtaining information or changing configuration elements. Physical security measures such as having locks, id card scanners, video surveillance, or even armed guards for data centers or back-office areas in a building still matter.



See FISCAM AC.03.02: Cryptographic controls are appropriately selected and employed based on risk. (p 500B-185 and following) It lists a variety of control activities with audit procedures and relevant criteria sources.

Auditing controls intended to protect encryption keys is required in many situations.
If an organization uses standard levels of security, e.g., https on a website, they acquire and manage public and private keys. Encryption details are complex. Let's focus on a few key ideas:

- Although it is not intuitive to think so, it is possible to generate two numbers that can be used for encryption system such that:
    - If you encrypt with one you can only decrypt with the other
    - Knowing one does not let you figure out the other
- We need to share the public half of that key pair so others can secure their communications with us.
- But the private half of the key must be kept secret. If the private key becomes known the encryption fails completely.

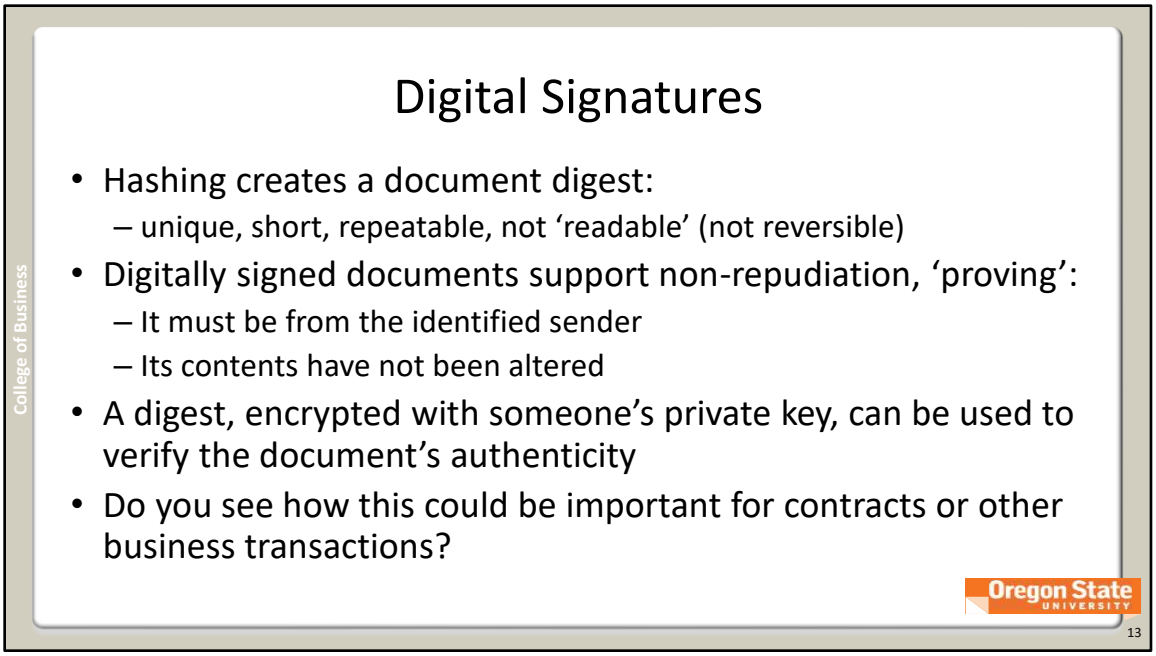
Here is a practical example.
A company sets up TLS security on its web server. Users open web pages using https instead of just http. This has become standard and if a website does not employ TLS, web browsers now warn that a site using only http is unsafe. To support TLS, organizations deploy digital certificates. Those certificates include a public key. People who connect to the site use the readily available public key to encrypt communications with the site. Modern browsers validate certificates based on the reputation of the certificate authority that generated the certificate.

This approach is of no use unless:

- Users can be confident that the public key they are using actually belongs to the right web site.
- The corresponding private key has been kept safe. Only the website (and perhaps a very few people who work for the company that hosts the web site) have access to the private key.

Certificate authorities maintain lists of known public keys and cross reference those key values to specific known parties. For example, a bank purchases a digital certificate from a certificate authority. The certificate authority has gathered evidence that the actual bank was the one who purchased the certificate and that the bank has the corresponding private key.

The underlying technologies for sharing and validating public keys and for protecting private keys are beyond our scope here. It is enough to know that IT practitioners deploy keys and controls to protect the keys. IT auditors assess those controls.



See FISCAM AC.02.02.06 *Hint: quiz and exam questions may well ask about the details here!*

We are not going to go deeply into how encryption works – auditors and IT practitioners need to know: (a) what encryption tools do, (2) how to safely deploy encryption tools, and (3) where risks arise. But they don't need to understanding the mathematics. Here are some non-technical details:

In addition to use of a public/private key pair, digital signatures and other encryption-powered pipelines use hashing algorithms: A hashing algorithm makes a digest of a document:

- Digests are short, perhaps a couple thousand ones and zeros – equivalent to less than a page of simple text. This is true even if the source document is a whole book – the digest is still short.
- Digests are unique. Changing even a single character in the source document will change the resulting digest. And, changes are not incremental. Change of a single bit in the source document will result in changes throughout a digest.
- Hashing is repeatable. Parties agree on the algorithm so both will get the same digest from the same source document.
- You cannot recreate the original document from the digest. Hashing is, essentially, a one-way process.

When you combine the digest with public key encryption, a receiver can be confident that they have received exactly what the sender sent – no one altered it in transmission.
And the sender can not credibly deny having sent the exact contents of the message. We call that non-repudiation. In order for the digital signature to be invalid, the sender must have – perhaps inadvertently – allowed some to use their private key. A very unlikely occurrence.

In short, the sender encrypts the digest using their public key and sends both the encrypted digest and the source document.

Steps to validate a digitally signed message:

1. Obtain the sender's public key from the third-party certificate authority.
2. Decrypt the digest using that public key.
3. Compute a new digest based on the provided source document.
4. Compare the provided digest with the one you computed.

If the digests match, the document must be authentic because only the sender could make a file that would properly decrypt using their public key. And the file must be unchanged because the digest matches.