

# Chapter 10. Security and Privacy

Now that you've learned about the data engineering lifecycle, we'd like to reiterate the importance of security and share some straightforward practices you can incorporate in your day-to-day workflow. Security is vital to the practice of data engineering. This should be blindingly obvious, but we're constantly amazed at how often data engineers view security as an afterthought. We believe that security is the first thing a data engineer needs to think about in every aspect of their job and every stage of the data engineering lifecycle. You deal with sensitive data, information, and access daily. Your organization, customers, and business partners expect these valuable assets to be handled with the utmost care and concern. One security breach or a data leak can leave your business dead in the water; your career and reputation are ruined if it's your fault.

Security is a key ingredient for privacy. Privacy has long been critical to trust in the corporate information technology space; engineers directly or indirectly handle data related to people's private lives. This includes financial information, data on private communications (emails, texts, phone calls), medical history, educational records, and job history. A company that leaked this information or misused it could find itself a pariah when the breach came to light.

Increasingly, privacy is a matter of significant legal importance. For example, the Family Educational Rights and Privacy Act (FERPA) went into effect in the US in the 1970s; the Health Insurance Portability and Accountability Act (HIPAA) followed in the 1990s; GDPR was passed in Europe in the mid-2010s. Several US-based privacy bills have passed or will soon. This is just a tiny sampling of privacy-related statutes (and we believe just the beginning). Still, the penalties for violation of any of these laws can be significant, even devastating, to a business. And because data systems are woven into the fabric of education, health care, and business, data engineers handle sensitive data related to each of these laws.

A data engineer's exact security and privacy responsibilities will vary significantly between organizations. At a small startup, a data engineer may do double duty as a data security engineer. A large tech company will have armies of security engineers and security researchers. Even in this situation, data engineers will often be able to identify security practices and technology vulnerabilities within their own teams and systems that they can report and mitigate in collaboration with dedicated security personnel.

Because security and privacy are critical to data engineering (security being an undercurrent), we want to spend some more time covering security and privacy. In this chapter, we lay out some things data engineers should consider around security, particularly in people, processes, and technology (in that order). This isn't a complete list, but it lays out the major things we wish would improve based on our experience.

## People

The weakest link in security and privacy is *you*. Security is often compromised at the human level, so conduct yourself as if you're always a target. A bot or human actor is trying to infiltrate your sensitive credentials and information at any given time. This is our reality, and it's not going away. Take a defensive posture with everything you do online and offline. Exercise the power of negative thinking and always be paranoid.

## The Power of Negative Thinking

In a world obsessed with positive thinking, negative thinking is distasteful. However, American surgeon Atul Gawande wrote a 2007 op-ed in the [New York Times](#) on precisely this subject. His central thesis is that positive thinking can blind us to the possibility of terrorist attacks or medical

emergencies and deter preparation. Negative thinking allows us to consider disastrous scenarios and act to prevent them.

Data engineers should actively think through the scenarios for data utilization and collect sensitive data only if there is an actual need downstream. The best way to protect private and sensitive data is to avoid ingesting this data in the first place.

Data engineers should think about the attack and leak scenarios with any data pipeline or storage system they utilize. When deciding on security strategies, ensure that your approach delivers proper security and not just the illusion of safety.

## Always Be Paranoid

Always exercise caution when someone asks you for your credentials. When in doubt—and you should always be in extreme doubt when asked for credentials—hold off and get second opinions from your coworkers and friends. Confirm with other people that the request is indeed legitimate. A quick chat or phone call is cheaper than a ransomware attack triggered through an email click. Trust nobody at face value when asked for credentials, sensitive data, or confidential information, including from your coworkers.

You are also the first line of defense in respecting privacy and ethics. Are you uncomfortable with sensitive data you've been tasked to collect? Do you have ethical questions about the way data is being handled in a project? Raise your concerns with colleagues and leadership. Ensure that your work is both legally compliant and ethical.

## Processes

When people follow regular security processes, security becomes part of the job. Make security a habit, regularly practice real security, exercise the principle of least privilege, and understand the shared responsibility model in the cloud.

## Security Theater Versus Security Habit

With our corporate clients, we see a pervasive focus on compliance (with internal rules, laws, recommendations from standards bodies), but not enough attention to potentially bad scenarios. Unfortunately, this creates an illusion of security but often leaves gaping holes that would be evident with a few minutes of reflection.

Security needs to be simple and effective enough to become habitual throughout an organization. We're amazed at the number of companies with security policies in the hundreds of pages that nobody reads, the annual security policy review that people immediately forget, all in checking a box for a security audit. This is security theater, where security is done in the letter of compliance (SOC-2, ISO 27001, and related) without real *commitment*.

Instead, pursue the spirit of genuine and habitual security; bake a security mindset into your culture. Security doesn't need to be complicated. For example, at our company, we run security training and policy review at least once a month to ingrain this into our team's DNA and update each other on security practices we can improve. Security must not be an afterthought for your data team. Everyone is responsible and has a role to play. It must be the priority for you and everyone else you work with.

## Active Security

Returning to the idea of negative thinking, *active security* entails thinking about and researching security threats in a dynamic and changing world. Rather than simply deploying scheduled simulated phishing attacks, you can take an active security posture by researching successful phishing attacks

and thinking through your organizational security vulnerabilities. Rather than simply adopting a standard compliance checklist, you can think about internal vulnerabilities specific to your organization and incentives employees might have to leak or misuse private information.

We have more to say about active security in [“Technology”](#).

## The Principle of Least Privilege

The *principle of least privilege* means that a person or system should be given only the privileges and data they need to complete the task at hand and nothing more. Often, we see an antipattern in the cloud: a regular user is given administrative access to everything, when that person may need just a handful of IAM roles to do their work. Giving someone carte blanche administrative access is a huge mistake and should never happen under the principle of least privilege.

Instead, provide the user (or group they belong to) the IAM roles they need when they need them. When these roles are no longer needed, take them away. The same rule applies to service accounts. Treat humans and machines the same way: give them only the privileges and data they need to do their jobs, and only for the timespan when needed.

Of course, the principle of least privilege is also critical to privacy. Your users and customers expect that people will look at their sensitive data only when necessary. Make sure that this is the case. Implement column, row, and cell-level access controls around sensitive data; consider masking PII and other sensitive data and create views that contain only the information the viewer needs to access. Some data must be retained but should be accessed only in an emergency. Put this data behind a *broken glass process*: users can access it only after going through an emergency approval process to fix a problem, query critical historical information, etc. Access is revoked immediately once the work is done.

## Shared Responsibility in the Cloud

Security is a shared responsibility in the cloud. The cloud vendor is responsible for ensuring the physical security of its data center and hardware. At the same time, you are responsible for the security of the applications and systems you build and maintain in the cloud. Most cloud security breaches continue to be caused by end users, not the cloud. Breaches occur because of unintended misconfigurations, mistakes, oversights, and sloppiness.

## Always Back Up Your Data

Data disappears. Sometimes it's a dead hard drive or server; in other cases, someone might accidentally delete a database or an object storage bucket. A bad actor can also lock away data. Ransomware attacks are widespread these days. Some insurance companies are reducing payouts in the event of an attack, leaving you on the hook both to recover your data and pay the bad actor who's holding it hostage. You need to back up your data regularly, both for disaster recovery and continuity of business operations, if a version of your data is compromised in a ransomware attack. Additionally, test the restoration of your data backups on a regular basis.

Data backup doesn't strictly fit under security and privacy practices; it goes under the larger heading of *disaster prevention*, but it's adjacent to security, especially in the era of ransomware attacks.

## An Example Security Policy

This section presents a sample security policy regarding credentials, devices, and sensitive information. Notice that we don't overcomplicate things; instead, we give people a short list of practical actions they can take immediately.

## Example Security Policy

### Protect Your Credentials

Protect your credentials at all costs. Here are some ground rules for credentials:

- Use a single-sign-on (SSO) for everything. Avoid passwords whenever possible, and use SSO as the default.
- Use multifactor authentication with SSO.
- Don't share passwords or credentials. This includes client passwords and credentials. If in doubt, see the person you report to. If that person is in doubt, keep digging until you find an answer.
- Beware of phishing and scam calls. Don't ever give your passwords out. (Again, prioritize SSO.)
- Disable or delete old credentials. Preferably the latter.
- Don't put your credentials in code. Handle secrets as configuration and never commit them to version control. Use a secrets manager where possible.
- Always exercise the principle of least privilege. Never give more access than is required to do the job. This applies to all credentials and privileges in the cloud and on premises.

### Protect Your Devices

- Use device management for all devices used by employees. If an employee leaves the company or your device gets lost, the device can be remotely wiped.
- Use multifactor authentication for all devices.
- Sign in to your device using your company email credentials.
- All policies covering credentials and behavior apply to your device(s).
- Treat your device as an extension of yourself. Don't let your assigned device(s) out of your sight.
- When screen sharing, be aware of exactly what you're sharing to protect sensitive information and communications. Share only single documents, browser tabs, or windows, and avoid sharing your full desktop. Share only what's required to convey your point.
- Use "do not disturb" mode when on video calls; this prevents messages from appearing during calls or recordings.

### Software Update Policy

- Restart your web browser when you see an update alert.
- Run minor OS updates on company and personal devices.
- The company will identify critical major OS updates and provide guidance.
- Don't use the beta version of an OS.
- Wait a week or two for new major OS version releases.

These are some basic examples of how security can be simple and effective. Based on your company's security profile, you may need to add more requirements for people to follow. And again, always remember that people are your weakest link in security.

## Technology

After you've addressed security with people and processes, it's time to look at how you leverage technology to secure your systems and data assets. The following are some significant areas you should prioritize.

### Patch and Update Systems

Software gets stale, and security vulnerabilities are constantly discovered. To avoid exposing a security flaw in an older version of the tools you're using, always patch and update operating systems and software as new updates become available. Thankfully, many SaaS and cloud-managed services automatically perform upgrades and other maintenance without your intervention. To update your own code and dependencies, either automate builds or set alerts on releases and vulnerabilities so you can be prompted to perform the updates manually.

### Encryption

Encryption is not a magic bullet. It will do little to protect you in the event of a *human* security breach that grants access to credentials. Encryption is a baseline requirement for any organization that respects security and privacy. It will protect you from basic attacks, such as network traffic interception.

Let's look separately at encryption at rest and in transit.

#### Encryption at rest

Be sure your data is encrypted when it is at rest (on a storage device). Your company laptops should have full-disk encryption enabled to protect data if a device is stolen. Implement server-side encryption for all data stored in servers, filesystems, databases, and object storage in the cloud. All data backups for archival purposes should also be encrypted. Finally, incorporate application-level encryption where applicable.

#### Encryption over the wire

Encryption over the wire is now the default for current protocols. For instance, HTTPS is generally required for modern cloud APIs. Data engineers should always be aware of how keys are handled; bad key handling is a significant source of data leaks. In addition, HTTPS does nothing to protect data if bucket permissions are left open to the public, another cause of several data scandals over the last decade.

Engineers should also be aware of the security limitations of older protocols. For example, FTP is simply not secure on a public network. While this may not appear to be a problem when data is already public, FTP is vulnerable to man-in-the-middle attacks, whereby an attacker intercepts downloaded data and changes it before it arrives at the client. It is best to simply avoid FTP.

Make sure everything is encrypted over the wire, even with legacy protocols. When in doubt, use robust technology with encryption baked in.

# Logging, Monitoring, and Alerting

Hackers and bad actors typically don't announce that they're infiltrating your systems. Most companies don't find out about security incidents until well after the fact. Part of DataOps is to observe, detect, and alert on incidents. As a data engineer, you should set up automated monitoring, logging, and alerting to be aware of peculiar events when they happen in your systems. If possible, set up automatic anomaly detection.

Here are some areas you should monitor:

## Access

Who's accessing what, when, and from where? What new accesses were granted? Are there strange patterns with your current users that might indicate their account is compromised, such as trying to access systems they don't usually access or shouldn't have access to? Do you see new unrecognized users accessing your system? Be sure to regularly comb through access logs, users, and their roles to ensure that everything looks OK.

## Resources

Monitor your disk, CPU, memory, and I/O for patterns that seem out of the ordinary. Did your resources suddenly change? If so, this might indicate a security breach.

## Billing

Especially with SaaS and cloud-managed services, you need to oversee costs. Set up budget alerts to make sure your spending is within expectations. If an unexpected spike occurs in your billing, this might indicate someone or something is utilizing your resources for malicious purposes.

## Excess permissions

Increasingly, vendors are providing tools that monitor for permissions that are *not utilized* by a user or service account over some time. These tools can often be configured to automatically alert an administrator or remove permissions after a specified elapsed time.

For example, suppose that a particular analyst hasn't accessed Redshift for six months. These permissions can be removed, closing a potential security hole. If the analyst needs to access Redshift in the future, they can put in a ticket to restore permissions.

It's best to combine these areas in your monitoring to get a cross-sectional view of your resource, access, and billing profile. We suggest setting up a dashboard for everyone on the data team to view monitoring and receive alerts when something seems out of the ordinary. Couple this with an effective incident response plan to manage security breaches when they occur, and run through the plan on a regular basis so you are prepared.

# Network Access

We often see data engineers doing pretty wild things regarding network access. In several instances, we've seen publicly available Amazon S3 buckets housing lots of sensitive data. We've also witnessed Amazon EC2 instances with inbound SSH access open to the whole world for 0.0.0.0/0 (all IPs) or databases with open access to all inbound requests over the public internet. These are just a few examples of terrible network security practices.

In principle, network security should be left to security experts at your company. (In practice, you may need to assume significant responsibility for network security in a small company.) As a data engineer, you will encounter databases, object storage, and servers so often that you should at least be aware of simple measures you can take to make sure you're in line with good network access practices.

Understand what IPs and ports are open, to whom, and why. Allow the incoming IP addresses of the systems and users that will access these ports (a.k.a. whitelisting IPs) and avoid broadly opening connections for any reason. When accessing the cloud or a SaaS tool, use an encrypted connection. For example, don't use an unencrypted website from a coffee shop.

Also, while this book has focused almost entirely on running workloads in the cloud, we add a brief note here about hosting on-premises servers. Recall that in [Chapter 3](#), we discussed the difference between a hardened perimeter and zero-trust security. The cloud is generally closer to zero-trust security—every action requires authentication. We believe that the cloud is a more secure option for most organizations because it imposes zero-trust practices and allows companies to leverage the army of security engineers employed by the public clouds.

However, sometimes hardened perimeter security still makes sense; we find some solace in the knowledge that nuclear missile silos are air gapped (not connected to any networks). Air-gapped servers are the ultimate example of a hardened security perimeter. Just keep in mind that even on premises, air-gapped servers are vulnerable to human security failings.

## Security for Low-Level Data Engineering

For engineers who work in the guts of data storage and processing systems, it is critical to consider the security implications of every element. Any software library, storage system, or compute node is a potential security vulnerability. A flaw in an obscure logging library might allow attackers to bypass access controls or encryption. Even CPU architectures and microcode represent potential vulnerabilities; sensitive data can be [vulnerable](#) when it's at rest in memory or a CPU cache. No link in the chain can be taken for granted.

Of course, this book is principally about high-level data engineering—stitching together tools to handle the entire lifecycle. Thus, we'll leave it to you to dig into the gory technical details.

### Internal security research

We discussed the idea of *active security* in [“Processes”](#). We also highly recommend adopting an *active security* approach to technology. Specifically, this means that every technology employee should think about security problems.

Why is this important? Every technology contributor develops a domain of technical expertise. Even if your company employs an army of security researchers, data engineers will become intimately familiar with specific data systems and cloud services in their purview. Experts in a particular technology are well positioned to identify security holes in this technology.

Encourage every data engineer to be actively involved in security. When they identify potential security risks in their systems, they should think through mitigations and take an active role in deploying these.

## Conclusion

Security needs to be a habit of mind and action; treat data like your wallet or smartphone. Although you won't likely be in charge of security for your company, knowing basic security practices and keeping security top of mind will help reduce the risk of data security breaches at your organization.

## Additional Resources

- [Building Secure and Reliable Systems](#) by Heather Adkins et al. (O'Reilly)
- [Open Web Application Security Project \(OWASP\) publications](#)

- [\*Practical Cloud Security\*](#) by Chris Dotson (O'Reilly)