When an auditor asks who can approve a wire transfer, a manager wants to check access to a loan system, or an incident team needs to know which accounts can change member records, a static quarterly report may not be enough. You need a practical way to generate **ad-hoc access reports** that answer a specific question about access—without starting a new spreadsheet from scratch every time.

A useful report shows more than a list of names. It connects identities to accounts, systems, roles, and entitlements, then makes clear who reviewed the information and where the evidence lives. That takes reliable data, a repeatable workflow, and a way to distinguish employees from service accounts, shared mailboxes, and privileged or duplicate accounts.

This guide covers how to build that process, whether you are working from spreadsheets or considering an identity governance platform.



## Define the question before building the report

“Show access by role” can mean several different things. A report might list everyone assigned to a business role, show every entitlement held by people in a department, or identify accounts with a particular permission in one system. Start by defining the question the report must answer.

- **Who has access?** List people and non-person accounts that match the criteria.
- **Where is the access?** Identify the application, directory, database, or other system.
- **What access do they have?** Show business roles, technical roles, groups, and individual entitlements.
- **As of when?** Include the report’s data timestamp and the date the report was generated.
- **Who needs to act?** Identify the reviewer or owner responsible for validating the access.

A clear question also helps prevent a common reporting mistake: treating a role name as if it fully explains a person’s access. A role might bundle several permissions, while a user may also have direct entitlements outside that role. Include both the role assignment and the underlying permissions when they are available.



## Build a reportable access model

To **search user permissions** consistently across systems, establish a shared set of fields. Source systems often use different names for similar concepts, and the same display name can refer to different roles in different applications.

Field Why it matters Person or account identifier Provides a stable key for matching records; do not rely on display name alone. Account type Distinguishes employee, contractor, service account, shared account, and other account types. System or application Shows where the role or entitlement is assigned. Role and entitlement Separates business-friendly role labels from the actual permissions or groups. Account status Helps reviewers spot disabled, locked, or otherwise inactive accounts. Manager, owner, or reviewer Supports routing and makes accountability explicit. Source and last-updated timestamp Helps establish data lineage and whether the report is current enough to use.

Set rules for naming and mapping roles before aggregating data. For example, one system may call a permission “Loan Maintenance,” while a directory group uses an abbreviation and a spreadsheet uses a business-friendly label. Preserve the original source value, and add a normalized value for searching and reporting. That way, the report is easier to read without losing the detail needed to trace an entry back to its source.

## Prepare reliable source data

Many organizations start with exports from Active Directory, application administration screens, core systems, and spreadsheets maintained by system owners. A spreadsheet can be a useful input, but it should not silently become the authoritative record simply because it is convenient.

Before combining files, document who supplied each file, which system it represents, when it was extracted, and who owns the data. Then check for issues that can distort results:

- Duplicate accounts or people represented under different identifiers.
- Admin clones that are not clearly linked to the person who owns them.
- Shared mailboxes or shared accounts that appear to be individual users.

- Service accounts that lack a named business or technical owner.
- Inconsistent role names, missing values, and stale or disabled accounts.
- Different date formats or status values across imports.

These details are not cleanup trivia. They affect the report's meaning. If an administrator's elevated account appears separately from their everyday account, reviewers need a way to understand the relationship. If a service account is included in a person-only review, a manager may be unable to make a meaningful decision about it.

## From spreadsheet exports to repeatable reporting

Spreadsheet-based reviews are often the first step toward automation. The goal is not to ban spreadsheets overnight; it is to replace manual copying, email follow-ups, and disconnected decision records with a repeatable workflow.

1. **Standardize the input.** Define required columns, naming conventions, and accepted account types. Use a controlled template when systems cannot provide a direct feed.
2. **Validate the import.** Check required fields, duplicates, unexpected values, and file dates before treating the data as ready for review.
3. **Map records to identities and systems.** Match accounts using stable identifiers where possible, and retain the original source value for traceability.
4. **Generate the report or review population.** Filter by system, role, entitlement, department, manager, status, or account type.
5. **Assign decisions to accountable reviewers.** Route each item based on system ownership, management structure, or another documented rule.
6. **Record decisions and follow-up.** Keep approvals, revocations, exceptions, comments, and completion status together in the workflow—not in an email thread.
7. **Retain evidence.** Preserve the source data, mappings, report criteria, timestamps, reviewer decisions, and remediation status.

This approach makes spreadsheet imports more controlled while creating a path to platform-based access review automation. If a platform can ingest data from many systems, it may reduce repeat manual preparation—but only if the imports, mappings, and ownership rules are maintained. Automation does not make inaccurate source data accurate.

## Make reviewer assignments and decisions usable

A report is only useful if the right person can understand and act on it. Sending every access item to one broad group creates confusion; routing items based only on a manager field may also fail when access decisions belong to an application owner or data owner.

Define assignment rules that match the decision being requested. A manager may be the right person to confirm whether an employee still needs access. A system owner may be better placed to assess whether a technical role is appropriate. For a service account, the review may need to go to its named business owner and technical custodian.

Give reviewers enough context to make a decision without requiring them to decode raw exports. A good review view includes the account, system, role or entitlement, status, relevant owner, and a concise

description of the access. It should also offer clear actions such as retain, revoke, or request clarification, with a place to explain exceptions. If a reviewer cannot tell what an entitlement permits, they are being asked to approve a label rather than assess access.

Keep assignment, reminders, decisions, and remediation status in a trackable workflow. Email can notify a reviewer that action is needed, but an email reply should not be the system of record. The record should show who decided, what they decided, when they decided, and whether required changes were completed.

## Choose identity governance tools around the workflow

When evaluating identity governance options, begin with your reporting and evidence requirements—not a feature list. Ask vendors or internal teams to demonstrate the full workflow using a realistic access question, such as finding all active accounts with a specific role, assigning the results to the right owners, and documenting the final decisions.

- **Data coverage:** Can the solution accept data from the systems you need, including systems without modern connectors?
- **Import flexibility:** Can it handle structured files, recurring imports, field mapping, and validation without relying on one-off manual manipulation?
- **Search and filtering:** Can analysts search user permissions by person, role, entitlement, system, account type, and status?
- **Workflow assignment:** Can review items be routed by manager, application owner, or another defined responsibility?
- **Reviewer experience:** Is the access understandable, and can reviewers make and explain decisions efficiently?
- **Evidence and audit trail:** Where does the evidence live? Can you retrieve the source, criteria, decisions, timestamps, and remediation history?
- **Operational fit:** Can the organization maintain mappings, data owners, and review rules after implementation?

For credit unions and their technology partners, the environment may span directories, business applications, and core-related systems. Organizations working with providers such as Member Driven Technologies (MDT) should include the relevant systems and ownership boundaries in their reporting design rather than assuming every account is managed in the same place. Security Compliance Corp (SCC) and Access Auditor can also be included in a market or process evaluation, but assess each option against your own data sources, review responsibilities, and evidence requirements. A name on a shortlist is not a substitute for validating the workflow.

## Keep the report defensible

An ad-hoc report can support an investigation or a one-time management question, but it should still be reproducible. Record the filters used, source systems included, extraction dates, and any role mappings applied. Keep a copy of the output or a system-generated record, depending on your retention policy, along with the evidence supporting the underlying data.

Where logs or system records exist, use them to support the review rather than relying on screenshots as the primary evidence. Screenshots can be helpful for context, but they may not show when data was

captured, whether it changed, or how a decision was recorded. A report should make it possible to trace an entry back to the source and understand what happened after the review.

Finally, treat reporting as a lifecycle. Revisit mappings when roles change, **thesiliconreview.com** confirm that system and account owners remain current, and investigate exceptions rather than carrying them forward indefinitely. This helps keep future reports useful instead of turning each new request into another round of manual reconciliation.

## A practical starting point

To begin, choose one high-value reporting question and a manageable set of systems. Define the fields, validate the source data, and test whether the intended reviewers can understand and act on the results. Then document the evidence trail and the time required to complete the process.

Once the workflow is clear, compare spreadsheet-driven review with platform automation using the same scenario. The right approach is the one that can reliably **search user permissions**, present **roles and entitlements** in context, assign decisions to the right people, and preserve evidence that can be retrieved later. That is what turns **ad-hoc access reports** from one-off exports into a dependable access governance capability.