**Threat Modeling Protocol for Horizontal Cooperatives**

**Introduction and Key Principles**

Horizontal cooperatives face unique security challenges. Without a central authority, they are vulnerable to attacks that exploit their open, democratic nature. For example, an attacker might create fake member identities to sway decisions (a **Sybil attack**) or abuse the consensus process to cause chaos. At the same time, horizontality can be a **security strength**: distributing decisions means there’s no single point of failure. This protocol treats horizontality as an asset, making security a **collective endeavor** rather than a top-down mandate.

**Key Design Principles:** This threat modeling approach is built on values that mirror cooperative principles:

* **Transparency:** Security activities (like decisions, configurations, and incidents) should be visible to members. Open logs and auditable records ensure nothing happens “behind closed doors,” building trust and accountability among the group.
* **Decentralization:** No single person should have unchecked power over systems or data. Access and control are distributed. This prevents a “single admin” from being a weak link and avoids creating a **digital vanguard** (where a tech-savvy few hold all the keys).
* **Democratic Participation:** All members can participate in identifying and addressing threats. Security decisions are made through inclusive discussions or votes, so measures have collective buy-in. This keeps the process aligned with the coop’s democratic governance.
* **Traceability:** Every important action (granting access, making a change, etc.) leaves an immutable trail. For example, changes can be logged on tamper-proof ledgers and digitally signed by those who approved them. This way, if something goes wrong, the coop can trace what happened and who was involved, without relying on memory or hearsay.
* **Resilience:** The protocol aims to strengthen the coop’s ability to withstand and recover from threats. By eliminating single failure points and planning for crises (with backup plans and rapid response mechanisms), the organization stays resilient even under attack. If one safeguard fails, others are in place to limit damage and bounce back quickly.

These principles ensure that improving security will **not undermine the organization nature** of the group. Instead, security measures will reinforce collaboration, shared responsibility, and trust. In practice, this means building security into everyday cooperative workflows and governance. What follows is a step-by-step threat modeling process designed with these values in mind. It’s written in accessible language so that **any member – not just IT experts – can take part.** Each step includes guidance and checklists for participatory activities, and the protocol can be scaled or adapted for organizations of different sizes and structures.

*(Note: While this protocol is inspired by established frameworks like PASTA (Process for Attack Simulation and Threat Analysis), it does* ***not*** *use the formal seven-stage PASTA terminology. Instead, it presents an equivalent logic in a more accessible, coop-friendly format.)*

**Threat Modeling Process Overview**

The threat modeling process is broken into **eight collaborative steps**. In a small cooperative, most steps can be taken in all-hands meetings or workshops with everyone. In larger groups, you might delegate initial work to committees or working groups, but every member should have a chance to review and contribute at each stage. The process is iterative and **modular – you can adjust the depth or format of each step** based on your organization’s size and needs. For each step below, we outline the purpose, activities, and participatory approach, along with tips to adapt to different situations.

**Step 1: Establish Context and Goals (What Are We Protecting?)**

**Purpose:** Set the stage by agreeing on what assets and operations you need to protect, and what your security objectives are. This ensures everyone is on the same page about why you’re doing threat modeling and what “success” looks like. In traditional terms, this is like defining the business/mission objectives and scope for security.

**Activities/Checklist:**

* **Identify Critical Assets:** As a group, list out what is most valuable to your cooperative. This can include **digital assets** (member data, documents, the website, chat platforms), **physical assets** (office space, devices, servers), and **intangible assets** (the coop’s reputation, member trust, know-how). Ask yourselves: “What would really hurt if it were stolen, destroyed, or made public?” Write these down for all to see.
* **Outline Key Operations/Workflows:** Describe in simple terms what the coop does day-to-day. For example, “We coordinate orders through an online platform,” or “We have weekly meetings to make decisions,” or “We run a community space with an entry badge system.” Understanding these workflows helps identify where disruptions would be most damaging.
* **State Security Objectives and Requirements:** Discuss what security means for your coop. Do you need to keep member data private? Ensure your service is always available? Meet any legal/privacy regulations (like GDPR)? Also consider coop bylaws or policies about confidentiality and data handling. For instance, if your bylaws say all financial info must be accessible to members, that influences how you balance transparency with confidentiality. Jot down these objectives and any compliance requirements.
* **Define the Scope and Boundaries:** Decide what **will** and **won’t** be covered in this threat modeling exercise. Maybe you want to focus on a particular system (e.g. your member database and communication tools) and not on unrelated areas. Or include only digital systems but not physical office security – or vice versa. Clearly defining scope prevents the discussion from going off-track. It’s okay to start with a narrow scope (like “our shared Google Drive and Slack workspace”) and expand later if needed.
* **Agree on Terminology:** Ensure everyone understands basic terms you’ll use. For example, define what you mean by “asset,” “threat,” “vulnerability,” etc., in plain language. This avoids confusion later (a quick glossary on a whiteboard or shared doc can help).

**Participation Tips:** In a small coop, you can do this step in a single meeting where everyone contributes to the list of assets and goals. In a larger coop, consider sending a short survey or having breakout groups to gather input, then consolidating the results in a plenary session. Make sure the final list of assets and scope is reviewed or approved by the group (e.g. via a show of hands or online poll) so you have **collective agreement** on “what we care about protecting.” This collaborative start sets a democratic tone for the whole process.

**Step 2: Map Systems and Trust Boundaries (How Do We Work?)**

**Purpose:** Create a shared understanding of how information and processes flow in your coop, and where important trust boundaries lie. Essentially, **draw a map of your organization’s sociotechnical system** including people, tech, and their interactions. In threat modeling, this is similar to diagramming your system architecture and identifying entry points. For a cooperative, it also means noting social trust assumptions (who/what we trust and in what ways).

**Activities/Checklist:**

* **List Components and Assets:** Building Step 1’s asset list, break things down further. Inventory all major components:
  + **Hardware/Infrastructure:** e.g. member laptops, a server or NAS, routers, smartphones used for work, IoT devices in the office.
  + **Software/Tools:** e.g. the platforms you use – Google Drive, Slack/Matrix, Nextcloud, Loomio for decisions, etc., as well as any custom software or website your coop has.
  + **Data Stores:** e.g. databases, cloud storage, email archives – what data is stored and where.
  + **People/Roles:** e.g. general members, IT volunteers, finance coordinators, any outsiders like an accountant or platform provider.
  + **Processes:** e.g. how a decision is made (proposal -> discussion -> vote), how a new member is onboarded, how finances are managed.

Write these out, possibly in categories. Essentially, you’re enumerating what pieces make up your coop’s “system.”

* **Diagram the Workflow (Data Flow Diagram):** On a large paper or using a simple online diagram, sketch how these components connect. Draw **who interacts with what**: e.g. members (people) log into the chat platform (software) to discuss, or the website communicates with a payment processor. Draw arrows for data flow or interaction: emails sent, files shared, money transferred, etc. Keep it understandable – you can use simple icons or just labeled boxes and arrows. Mark any external services (like a third-party payment gateway or cloud provider) clearly, maybe with a different color or a cloud icon, since those are partly outside your control.
* **Identify Trust Boundaries:** A **trust boundary** is a point in the system where the level of trust changes. For instance:
  + Between an external user and your internal system (e.g. a public website vs. your internal database).
  + Between a regular member and an admin interface (if any).
  + Between your coop’s network and the open internet.
  + Social trust boundaries: e.g., you trust members not to leak info from private discussions, or you trust a core team with certain credentials.

On your diagram, draw a dotted line or a firewall symbol where these boundaries are. Essentially ask: “At what points do we assume things are safe on one side and potentially risky on the other?” (For example, data inside our Nextcloud is trusted to be seen by members, but anything coming from outside (like file uploads from a new user) might be untrusted until checked.)

* **Document Who Has Access to What:** Alongside the map, list which roles or people have access to which assets. E.g., “Only tech team members can access the server,” or “All members can post in the forum,” or “Treasurer has the bank account login.” This helps spotlight any **concentrations of access** (if one person holds many keys, that’s noted) and areas where trust is placed in individuals.
* **Note External Dependencies:** Write down services or partners you rely on and mark them on the diagram (for example, your website host, email service, or any software provider). These are outside your coop but critical; threats can come through them (a concept known as supply chain or third-party risk).

**Participation Tips:** Building the system map can be a fun group exercise. In a small coop, do it together on a whiteboard or shared screen, asking everyone to call out components and connections (“Don’t forget we also use Pad for meeting notes!”). In larger coops, you might have a smaller group draft the diagram (say, an IT working group or a mix of tech and non-tech members) and then present it to others for additions and corrections. Make it interactive: people can stick Post-its on a draft poster or add comments on an online diagram with “Did you include the volunteer’s laptop that runs X?” Keep refining until ,members feel the map is an accurate picture of “how things work.” The final diagram should be saved in a place where all members can see it – this is now a shared reference for your threat discussions.

**Step 3: Identify Threats Collaboratively (What Could Go Wrong?)**

**Purpose:** Brainstorm all the potential threats and bad things that could happen to the assets and processes you identified. The goal is a **comprehensive list of threat scenarios**, covering both technical attacks and social/governance risks. At this stage, quantity is more important than quality – we want to surface as many ideas as possible, without judging them yet. This step harnesses the diverse perspectives in your coop: tech-savvy members might think of hacking scenarios, whereas others might point out process failures or insider issues that a pure tech focus could miss.

**Activities/Checklist:**

* **Brainstorm in a Safe Environment:** Gather a group of members (ideally representing different roles or viewpoints in the coop) for a threat brainstorming session. Set some ground rules: no idea is too small or too “out there,” and everyone’s input is valued. It’s important people feel comfortable mentioning even unpleasant hypotheticals (“What if one of us turned rogue?”) – assure everyone this is about hypothetical situations, not personal distrust.
* **Use Prompts and Creative Tools:** To get the ideas flowing, use structured prompts:
  + Walk through the **system map** from Step 2 and pause at each component or boundary: ask “What could go wrong here?” For instance, at the database: “Could someone steal or delete this data? Who might and how?” At a boundary like the internet connection: “What if an attacker intercepts data here or floods us with traffic?”
  + Introduce the **STRIDE categories** (a classic security mnemonic) in simple terms as a checklist:
    - **Spoofing** (pretending to be someone else – e.g. fake member login, impersonating an admin),
    - **Tampering** (messing with data or systems – e.g. altering records, defacing the website),
    - **Repudiation** (denying an action – e.g. a member does something and later claims they didn’t, which is an issue if there’s no proof),
    - **Information Disclosure** (leaks – e.g. private member info exposed),
    - **Denial of Service** (disrupting service – e.g. someone takes down your communication channel or overloads your server),
    - **Elevation of Privilege** (gaining higher access – e.g. a regular member somehow gets admin rights). Use these as thought-starters: “Do we have a risk of someone spoofing identity? Tampering with records? etc.”
  + Use **Security Cards** (if available) or hypothetical scenario prompts: Security Cards are a deck with categories like *Attackers’ Motivations, Methods, Impacts*. You can simulate this by asking questions in those dimensions:
    - Motivations: “Who might want to attack us and why? (Ex: for money, for political reasons, disgruntled ex-member, random mischief)”
    - Methods/Resources: “What skills or tools could they use? (Ex: Phishing emails, malware, physical break-in, legal threats, bribes, social engineering phone calls)”
    - Impacts: “What would be the impact if they succeeded? (Ex: website down for days, loss of member trust, financial loss, legal trouble)”

These prompts help the group consider not just obvious IT threats, but also things like internal misuse, mistakes, or external events.

* **Distinguish Different Threat Sources:** As ideas emerge, note whether each threat scenario is **external** (coming from outside, e.g. a hacker, a virus, a competitor, a random troll) or **internal** (coming from within the coop, e.g. a member error, an insider attack, conflict/miscoordination). Both are important. For example:
  + External threat example: “A hacker defaces our website or steals our member list.”
  + Internal threat example: “A coop member accidentally shares a private document publicly,” or “Two factions in the coop conflict and someone locks others out of an account.”
  + Hybrid threat: “An external attacker tricks a member (social engineering) to gain access” or “An ex-member colludes with an outsider.”

By labeling these, you ensure internal governance vulnerabilities (like abuse of trust or poor processes) get attention alongside technical attacks.

* **Write Down Concrete Scenarios:** For each idea, capture it as a short scenario description. For instance:
  + *“Sybil Attack on decision-making:”* An attacker (or someone from the community) creates multiple fake member accounts to gain extra votes in an online poll, influencing a cooperative decision illicitly.
  + *“Insider data leak:”* A discontented member with access to sensitive data decides to leak member emails and addresses to the public.
  + *“Ransomware on shared drive:”* Malware infects a member’s computer and encrypts the shared cloud drive files, making them inaccessible until a ransom is paid.
  + *“Lost credentials:”* A member who manages the Twitter account leaves suddenly, and no one else has the password – the coop loses control of its own social media for a time.
  + *“Miscoordination outage:”* In a crisis, no one is designated to respond (because everyone thinks someone else will) and a small issue (like a certificate expiry) escalates, taking the website offline for days.
  + *“Service provider failure:”* The third-party platform (e.g. web host or payment processor) goes down or is compromised, affecting the coop’s operations.

Aim for a broad list, covering cyber-attacks, human mistakes, physical events (the office gets robbed or a server gets wet), and governance failures. Don’t worry at this stage if some scenarios seem very unlikely – list them if someone is concerned about it.

* **Ensure Social/Process Threats are Included:** Cooperatives might face threats like **quorum manipulation** (exploiting the rules of consensus/voting), **abuse of emergency powers** (someone invoking a crisis to grab authority), or **“digital vanguard” accumulation** (one person quietly gaining control of many digital assets because no one else steps up). Include these in your brainstorming. For example, *“Member X holds all the keys and if they quit or go rogue, we’re locked out”* is a valid threat scenario to record (it’s an internal risk).

**Participation Tips:** Use sticky notes or a shared online document during the brainstorm so everyone can contribute simultaneously. One approach is to give participants 5–10 minutes to silently write as many “What if…?” threats as they can (one per sticky note), then post them and discuss. This helps include those who might be quieter in a big group. After the brainstorm, **group similar threats** together (cluster duplicates) but **do not dismiss any threat yet** just because it seems minor or absurd. The point is to capture the collective worries and creative ideas. If you have a very long list (which is good), you can categorize them into groups like “Tech Infrastructure,” “Member Behavior,” “Outsider Attacks,” “Natural/External,” etc., for clarity.

Finally, thank everyone for their ideas – emphasize that all contributions (even wild ones) help build a complete picture. Save this threat list somewhere accessible (it becomes an input to later steps). In a coop, making this list visible (on the wall or online) also signals transparency: everyone sees what threats are on the radar.

**Step 4: Profile Adversaries (Who Might Attack Us?)**

**Purpose:** Humanize the threats by creating **adversary personas** – fictional characters that represent types of attackers or sources of threats. This helps the group think from an attacker’s perspective (“what would X do?”) and ensures you consider the motivations and capabilities behind the threats. In a cooperative, adversaries aren’t only outside hackers; they could be internal (like a frustrated member) or systemic (like software bugs or accidents – though we focus on personas for intentional actors here). Developing these profiles makes later analysis more concrete and relatable.

**Activities/Checklist:**

* **Identify Key Threat Actors:** Look at the threat scenarios from Step 3 and ask, “Who would carry out these actions?” You’ll likely find a few recurring archetypes. For example:
  + A **malicious outsider** (generic hacker or vandal) with no stake in the coop, just attacking for personal gain or fun.
  + A **state or corporate actor** who opposes your coop’s mission (if relevant politically or competitively).
  + A **disgruntled member or ex-member** who knows the internal system and might seek revenge or change outcomes.
  + A **negligent insider** (not malicious, but someone who might make mistakes – although this is more a cause of accidents than an “attacker,” it’s still a persona to consider for unintentional threats).
  + **Technical vs non-technical adversaries:** e.g., a script-kiddie hacker with moderate skills, vs. an internal activist who might misuse rules rather than code.
* **Create Persona Descriptions:** For each actor type, write a short profile including:
  + **Name & Role:** Give them a nickname that captures their role, like “Mallory the Malicious Member” or “Ingrid the Inattentive Intern” or “Oscar the Outside Hacker.” This makes it easy to refer to them.
  + **Motivations/Goals:** What do they want? Money, disruption of the coop’s activities, ideological reasons, personal vendetta, thrill? For instance, *Mallory (disgruntled ex-member)* might be motivated by revenge for being voted out of a project, whereas *Oscar (outsider)* might just want to steal data to sell.
  + **Capabilities/Resources:** What skills or resources do they have? *Oscar* might have hacking tools or know how to exploit software. *Mallory* has insider knowledge of how your coop operates and where the weaknesses in process are, but maybe not advanced technical skills. *Ingrid* (the negligent intern) isn’t trying to attack but might unknowingly introduce risk by ignoring policies or falling for scams.
  + **Possible Methods:** Given their motivation and skills, how might this persona attack or cause an incident? E.g., *Oscar* might use phishing emails or find a bug in your website; *Mallory* might exploit their still-active login or sow misinformation in meetings; *Ingrid* might click a malicious link or use a weak password that gets guessed.
  + **Scenario Tie-In:** Connect each persona to one or more of the scenarios from your threat list. For example, note that *“Sybil attack on voting”* would be done by someone like *Mallory* (if internal) or an outsider *Sam the Sockpuppet* if external. *“Ransomware on shared drive”* could be set off by *Ingrid’s mistake* or by *Oscar* deliberately.
* **Document in an Accessible Format:** Make one page or slide for each persona. You can even include a representative image (e.g., an icon or cartoon; avoid using real people’s photos to prevent bias or confusion). The idea is to make these threat actors memorable. Write in plain language – this is an internal tool for understanding, so it doesn’t need to be formal. For example:
  + **Persona: Mallory (Disgruntled Ex-Member)**
    - *Background:* Former member who left after a conflict. Still has some credentials left over.
    - *Motivation:* Revenge, proving a point about weak security.
    - *Skills/Resources:* Knows our systems well, not a hacking expert but savvy enough to abuse any oversight. Possibly still in contact with current members (could socially engineer info).
    - *Likely Actions:* Try to log in to systems with old account, misuse a still-active access, or spread false messages pretending to be someone else (impersonation). Might collude with an outsider for technical help.
    - *Targeted Assets:* Member directory (to expose personal info), internal chat (to disrupt communications), decision platform (to sway outcomes or just cause chaos).

Do this for each major adversary type.

* **Include at Least One Insider Persona:** It may be uncomfortable but include a scenario of a malicious or careless insider. Cooperatives thrive on trust, yet history shows sometimes insiders can cause harm (intentionally or not). By creating, say, *“Insider Irene”* who is well-meaning but prone to bypassing rules, or *“Rogue Ray”* who turns against the group, you can discuss those threats without pointing fingers at a real person. Make it clear this is hypothetical to improve security for everyone.
* **Use Personas in Discussion:** Once you have personas, you can use them in future steps. For example, when thinking about mitigations, you might ask “Would this stop Mallory?” or “How would we detect Oscar’s actions?” Personas help ground these discussions.

**Participation Tips:** Developing personas can be done in small breakout groups, with each group taking one persona to draft. It’s a creative exercise – encourage storytelling, but keep it grounded in plausibility. After drafting, share with the whole team for feedback. Non-technical members often contribute richly here (“Actually, a really angry member might do X, not Y”), and technical members can refine how an external hacker persona might behave. Keep the tone collaborative and even a bit playful (people can get quite engaged coming up with hacker nicknames!), but always tie it back to real threats you identified. In the end, post the personas alongside the threat list. They become part of your “threat model” documentation that everyone can reference.

**Step 5: Analyze Attack Scenarios (How Could Attacks Happen?)**

**Purpose:** Now take your threat list and personas, and dive deeper into **how those threats could play out step-by-step**. This scenario analysis helps you understand the sequence of events in an attack and where your weak points are. In practice, this means building **attack narratives or attack trees** and possibly simulating some scenarios in a tabletop exercise. This step turns abstract threats into concrete stories, revealing exactly what vulnerabilities enable an attack and how severe the consequences would be. It’s a bridge from brainstorming to action: by visualizing attacks, you prepare to figure out defenses.

**Activities/Checklist:**

* **Construct Attack Trees or Flowcharts:** Pick a high-priority threat scenario (you will prioritize formally in the next step but start with one that seems obviously serious or emblematic). For example, take *“steal member data”* or *“Sybil attack on voting”*. Write that as the **attacker’s goal** at the top (root of the tree). Then brainstorm the possible **paths to reach that goal**. For each path, break it into steps:
  + Example: Goal = *“Unauthorized person accesses member list.”* Path 1 might be: External hacker exploits a software vulnerability in the database -> gains admin privileges -> extracts the data. Path 2: Malicious insider uses their legitimate access -> downloads the data -> shares it out of malice. Path 3: Social engineer tricks a member into giving up their password -> logs in as them -> navigates to data and copies it.
  + Draw this as a tree: the goal at top, then branches for each different method, and sub-branches for steps/prerequisites. (It can be as simple as bullet indentations if not drawing: e.g. a numbered list of steps for each scenario.)
  + Mark points on the tree where a **defense exists or fails**. E.g., “Does our system have a vulnerability? Possibly if software not up-to-date.” Or “Insider route: relies on that member having access – do all members have access to full member list? Maybe that’s a policy question.” This highlights vulnerabilities at each step.
  + Keep the tree at a detail level that’s useful – not every keystroke, but significant steps and decision points. Two or three levels deep is usually fine (primary ways -> specific steps).
* **Conduct Tabletop Simulations:** For some scenarios, especially ones involving multiple people or processes, do a role-play **tabletop exercise**. Assemble a small group and narratively walk through the scenario:
  + Assign someone to play the adversary (using one of your personas) and others to play defenders or just observers.
  + Example: Simulate *“Sybil attack during an online vote.”* Narrator says: *“It’s the day of a big proposal vote. Unknown to the group, Mallory has created three fake member identities over the last month.”* Then step by step: *“Vote opens. Mallory votes as herself and as Alice, Bob, and Charlie (her fake profiles). The system tallies four votes from what appears to be four people.”* Discuss: Would anything at that moment flag an issue? How would the coop notice? Perhaps another member finds it odd that there were three new members who never spoke but voted. Or maybe nobody notices until later. Continue: *“The vote passes with those extra votes. Later, someone questions the outcome…”* This kind of storytelling helps highlight if your current processes have detection or not. Participants can chime in with “At that point, I would check the member list…” or ask “Do we verify new online voters somehow?” Write down these insights.
  + Another example: *“Insider incident response.”* Simulate what happens if an insider is caught doing something suspicious – do you have an agreed response or does chaos ensue?
  + Or *“Server ransomware attack at 2 AM.”* Who gets the call? Is there a backup? Walk through who does what.

The idea is to **practice an attack in theory** to see where your response or system breaks. It’s much cheaper to find gaps this way than during a real incident.

* **Identify Vulnerabilities at Each Step:** As you chart out these scenarios, explicitly list the vulnerabilities or weak points that make the attack possible. These could be technical (e.g. “Outdated plugin allows injection”, “No backup exists for database”), or organizational (e.g. “New members are not verified, allowing fakes”, “Only one person knows how to reset the server”). Also note any existing controls and whether they work:
  + For each step in the scenario, ask “What should stop this? Do we have something to stop it? Does it actually stop it or can it be bypassed?”
  + E.g., in the Sybil scenario: Vulnerability = lack of strict member verification in the voting system. Existing control = new accounts require admin approval (does that happen? maybe someone auto-approved without checks).
  + In the hacker scenario: Vulnerability = software not patched; Control = we have a firewall, but if the attack comes through a web port, firewall doesn’t stop it; or Control = we rely on strong passwords, which might not help if exploit exists.
  + Write these vulnerabilities next to the steps or in a separate list mapped to the scenario. This will directly feed into deciding mitigations.
* **Assess Impact and Likelihood for Scenarios:** As part of analysis, discuss for each scenario, **how bad would it be if this happened?** and **how likely is it to happen?** Use qualitative terms (you will formalize in next step, but start conversation):
  + Impact: High (e.g. coop might dissolve or face legal action), Medium (painful but survivable), Low (minor inconvenience or embarrassment).
  + Likelihood: High (we’ve seen attempts or it’s easy to do), Medium, Low (requires many unlikely failures or very targeted effort).
  + Example: Ransomware encrypting files – Impact High (work comes to halt, data loss) but Likelihood maybe Medium if members are generally careful. Sybil attack – Impact High on governance legitimacy, Likelihood Low/Medium depending on how easy it is to create fake accounts in your system.
  + Capture these impressions because they will help when you formally prioritize in Step 6.
* **Leverage Past Incidents:** If your coop or similar groups have experienced incidents, incorporate those into scenarios. “This happened before – could it happen again in a worse way?” Learning from near-misses or history makes scenarios very concrete. For example, *“Last year someone guessed our Twitter password – what if they had tweeted offensive stuff? Let’s play out that scenario.”*

**Participation Tips:** For attack trees, it can help to have a tech-savvy member sketch the first draft with input, then review with the group to fill gaps. For tabletop simulations, try to involve a mix of roles – perhaps an IT person, a governance person, and a regular member. Keep the tone educational, not fearful or accusatory. Encourage people to *“think like an attacker”* for a moment – it’s often eye-opening for non-security folks and can be even a bit fun in a serious way. Time-box the simulation (e.g. 20-30 minutes each) to keep it focused. After each scenario, have a brief “debrief” discussion: *What did we learn?* *What worked or failed in our current setup?* This will set you up well for the next step, because you’ll have identified where the biggest weaknesses are.

**Step 6: Prioritize Risks Together (Which Problems Matter Most?)**

**Purpose:** Not all threats are equal. In this step, the cooperative evaluates all the identified threat scenarios and **decides which ones to address first**. This is essentially a **risk analysis and ranking**. Risk is usually judged by two factors: how **severe** the impact would be and how **likely** the threat is to occur. By scoring or discussing these, the group can focus on the most critical issues. Importantly, this is done *participatorily* – everyone’s perspective on what is important is considered, keeping the process democratic. The output will be a clear list of top-priority risks that the coop will invest effort in mitigating.

**Activities/Checklist:**

* **Set Risk Criteria (Impact & Likelihood):** As a group, establish a simple shared understanding of risk levels:
  + Define what **High, Medium, Low Impact** mean for your coop. For example:
    - High Impact: Threatens the existence of the coop, causes major financial loss or legal violation, or fundamentally damages trust (e.g. member personal info exposure, loss of critical systems for long time, major public scandal).
    - Medium Impact: Disruptive and costly, but manageable (e.g. service outage for a day, loss of some data that has backups, temporary hit to reputation).
    - Low Impact: Annoying but minor consequences (e.g. defacement of a webpage, a single incorrect transaction that can be fixed).
  + Define **Likelihood** similarly:
    - High Likelihood: We have evidence or reason to believe this could happen frequently or easily (e.g. known attempts have occurred, or known vulnerabilities exist and are unpatched).
    - Medium: It could happen under certain conditions or if an attacker dedicates effort, but not trivial or not seen yet.
    - Low: Very rare or would require a perfect storm of failures or an extremely capable adversary.
  + If you prefer numbers, you can rate both on a 1-5 scale or 1-10. But keeping it simple with words or a 3-point scale is often fine for coops, as long as everyone talks through the meanings.
* **Evaluate Each Threat Scenario:** Go through the list of scenarios (from Step 3/5) one by one and discuss:
  + “If this happened, how bad would it be? (Impact)” and “How likely is it to happen given what we know? (Likelihood)”.
  + Encourage input: maybe the IT person knows that a certain attack is actually quite hard, so likelihood is low; but a governance person might point out that even if low, that attack’s impact on member trust would be catastrophic. Both points are valid.
  + You can do this discussion informally and just mark each scenario High/Med/Low, or use a more structured method like **voting** or rating:
    - For instance, create a grid on a whiteboard with Impact on one axis and Likelihood on the other, and place each threat (on a sticky note) in the grid where the group thinks it belongs. This visual “risk map” often sparks debate (“Should this be higher impact than that?” etc.).
    - Alternatively, have each member or a small group give a quick rating for each scenario anonymously (e.g. via a Google Form or pieces of paper), then average the scores. This can help if the group is large to get a baseline, then discuss outliers.
  + Aim for consensus or at least a general agreement on where each major threat sits (exact precision not needed – you mainly want to clearly identify the High-High items).
* **Rank the Risks:** Based on the evaluations, identify the **top tier of risks** that demand action. These are typically the scenarios with High impact (even if low likelihood) and those with High likelihood (even if medium impact), especially anything that’s High in both. Often you’ll find a handful that stand out as “we absolutely must address these.”
  + Also pay attention to “low-hanging fruit” – a scenario that might be Medium risk but can be fixed easily. Sometimes the group might say, “This isn’t our worst problem, but it’s so simple to prevent that we should just do it.” (Example: “Data loss due to no backup” might be medium likelihood and medium impact, but the fix – implement backups – is straightforward, so you might prioritize it early.)
  + You might end up with categories like: **Critical (must fix ASAP)**, **Moderate (plan to address)**, **Acceptable or Low (acknowledge but no immediate action)**.
* **Document Rationale:** For transparency, write down a brief note next to each top risk about why it’s ranked high and what the group’s judgment was. E.g., “Sybil attack on voting – **High Impact** (could undermine our governance legitimacy), **Low Likelihood** (member sign-up currently requires approval, and we’ve seen no attempts) – **Still a Top Concern** because of impact on trust.” This record helps if later someone asks “Why are we focusing on this threat over that one?” Everyone can see the reasoning agreed upon.
* **Verify Against Goals:** Revisit Step 1’s assets and objectives. Ensure that the top risks you’ve chosen indeed relate to what you care about most. If an important asset has no high risks listed, double-check: did we miss a threat for it, or is it truly low-risk? Conversely, if a scenario came up high risk but involves something not central to your mission, consider if it’s truly a priority. This sanity check keeps the process aligned with cooperative values and priorities.
* **Obtain Group Endorsement:** Formally or informally, get the group’s nod that “Yes, these are the threats we’re most concerned about.” In a smaller coop, that might be as simple as everyone saying “Sounds right.” In a larger one, you might do a quick vote to approve the risk ranking or have the board/steering committee ratify it. This is important for accountability – it shows the decision on what to do next was made collectively.

**Participation Tips:** Risk prioritization can sometimes lead to **debate or disagreement**, because people perceive risks differently. That’s okay – the discussion is part of the participatory process. Ensure that louder voices don’t completely drown out quieter ones; actively ask for input from different members (“We’ve heard a lot about technical likelihood – how about the user experience perspective, what do you think the impact on members would be?”). If consensus is hard, you can fall back to a vote or averaging scores, as long as the group accepts that outcome. Remember the principle of **democratic decision-making followed by unity in execution**: once the group decides that, say, “Insider data leak” is a top risk and “DDoS attack on our website” is lower priority, everyone should respect that and focus on the top risks first. This avoids second-guessing later and ensures a coherent security effort.

By the end of Step 6, you should have a short list of the most important threats to tackle, with group support behind that list. Now it’s time to figure out how to **mitigate** those threats.

**Step 7: Plan Mitigations and Governance Decisions (How Do We Fix or Prevent Issues?)**

**Purpose:** For each of the top-priority threats, decide on **countermeasures** and integrate those decisions into the coop’s action plan. In other words, figure out **what security measures to implement** and **how to approve and enforce them democratically**. This step is where you turn analysis into concrete changes: technical fixes, new or improved policies, training, etc. It’s also where you make sure that implementing these fixes doesn’t accidentally centralize power or violate cooperative principles. We design the mitigations to both reduce risk **and** fit into the coop’s governance structure.

**Activities/Checklist:**

* **Brainstorm Mitigation Options:** For each high-priority threat scenario, list possible ways to mitigate it. Mitigations can fall into different categories:
  + **Technical solutions:** e.g. apply a software patch or upgrade, enable two-factor authentication for logins, encrypt data at rest, set up an intrusion detection system, require multi-signature for financial transactions.
  + **Process or workflow changes:** e.g. institute a checklist for onboarding/offboarding members (so fake identities are caught and departing members lose access promptly), establish a routine backup procedure, require a second pair of eyes (peer review) before pushing changes to the website, add a step to verify any important request (like fund transfers) through a secondary channel.
  + **Education and training:** e.g. do a phishing awareness session so members can spot suspicious emails, create a quick guide on how to choose strong passwords or use the password manager the coop adopts, run an orientation on security policies for all members annually.
  + **Governance/policy measures:** e.g. create a policy that “All admin passwords are stored in our shared password manager vault accessible by at least 3 people,” or “For emergency decisions, at least 2 out of 3 designated responders must agree,” or “Every proposal in the decision platform must be made by a verified member identity.” Essentially, rules that formalize the security practices.

Use the adversary personas as a lens: ask “If we do this, would it stop or deter [Persona] from succeeding?” For instance, to counter *Mallory (disgruntled ex-member)* misusing credentials, one mitigation might be “immediately deactivate accounts when someone leaves (offboarding policy).” To counter *Oscar (outside hacker)* exploiting a bug, mitigation is “keep software updated, and maybe run security scans.” To counter the *Sybil scenario*, mitigation could be “implement stricter member verification for online voting (like each account must link to a real person’s membership record).”

List multiple options if they exist, then discuss feasibility:

* + How hard or expensive is it?
  + Does it require outside help or new tools?
  + Does it slow down any workflow (introduce friction)?
  + Does it align with our values (e.g. a mitigation “ban all new members for safety” would harm the coop’s openness, so probably unacceptable)?
* **Collective Deliberation on Mitigations:** For each threat and its potential solutions, discuss with the group (or relevant sub-group) to choose the best course of action. This might happen in a dedicated security meeting or as part of regular meetings. Ensure to involve those who will be responsible for implementing or affected by the change. For example, if the mitigation is “use a new encrypted chat tool,” the IT team and regular members should both weigh in (IT on how to implement, members on usability).

Some mitigations might be contentious (maybe someone feels requiring 2FA is too burdensome, or rotating admin duties is too chaotic). Strive for consensus by addressing concerns: perhaps provide alternatives or phased approaches. If consensus can’t be reached, use your coop’s normal decision process (majority vote, etc.) to finalize the decision on that mitigation.

**Important:** Document the decision for each major mitigation: the group agrees “Yes, we will do X to address threat Y.” This can be recorded in meeting minutes or an action plan.

* **Assign Responsibility and Resources:** For each chosen mitigation, decide **who will carry it out and by when**. Since this is a horizontal organization, avoid dumping everything on one person. Instead:
  + Spread technical tasks among tech-skilled members or a tech working group, but maybe pair them with a non-tech member for transparency.
  + Assign policy drafting to a small team that includes people from governance and operations.
  + If training is needed, maybe identify a member who’s good at teaching, or an external expert if budget allows.
  + Ensure each mitigation has an “owner” or small team responsible for seeing it through. This distributes responsibility and builds shared security knowledge.
  + If certain mitigations require budget (e.g. buying a security certificate, or paying for a backup service), note that and ensure it goes through the normal budgeting process of the coop.

Essentially, integrate these tasks into your coop’s project management – treat them like important initiatives with accountability. This also helps avoid the situation where plans are made but no one follows up.

* **Implement with Democratic Oversight:** As mitigations are implemented, keep the group informed. For example:
  + If the tech team is deploying a new secure tool, they should report progress at meetings or in the group chat.
  + If a policy is drafted (e.g. an “Acceptable Use Policy” for members), circulate it for comments and formally adopt it through the coop’s decision mechanism.
  + **Unity in execution:** Once a security measure is agreed on, all members should cooperate in making it work, even if it’s a bit inconvenient. This might mean everyone *actually enables 2FA* on their accounts, or *actually uses the new password manager*, not just a few. Culturally, frame it as “we all agreed to this because it protects all of us.” When needed, gently remind folks that this was a collective decision (the democratic centralism idea: we discussed openly, now we implement uniformly).
* **Plan for Emergency Situations (Temporary Delegation):** One big governance aspect to settle is: *How will we handle a security emergency or critical incident?* In a crisis (like an active cyber-attack, or a major breach in progress), coops might need to act faster than usual consensus allows. The protocol encourages having a pre-agreed **emergency response team or procedure**:
  + Decide if you will appoint a small **Security Incident Response Team (SIRT)** or similar. This could be, say, 3 trusted members with the technical know-how (or quick learners) who are empowered to make snap decisions when an incident is detected.
  + Define the **scope and limits** of their power: e.g., “They can temporarily shut down a server, or revoke someone’s access, or spend up to $X on emergency help, without full group approval *in that moment*.” Also define the timeframe: their actions are meant to contain the issue, and must be reported ASAP to the membership and reviewed, say, within 24-48 hours.
  + This delegation is **temporary and accountable**: as soon as the immediate threat is handled, they must explain their actions to the whole coop (transparently via an incident report), and if any decision needs to be permanent (like firing an IT provider or changing a policy), it goes to the group for normal decision-making.
  + If your coop is small, your “team” might just be the few people who know what to do, but still explicitly acknowledge it (“If something blows up, Alice and Bob will take charge for the moment, and inform everyone as they do”).

Having this plan prevents paralysis in a crisis and avoids someone unilaterally taking over just because nothing was defined. Everyone knows *in advance* who will act and trusts that it’s been agreed upon.

* **Integrate Security into Governance Documents:** Once decisions are made (normal mitigations or emergency roles), codify them. Update your coop’s handbook, wiki, or policy docs to include the new security measures. For example, add an “Access Control Policy” section that outlines how accounts are managed collectively, or a note in the decision-making section that describes the emergency procedure. This ensures new members will learn these practices during onboarding, and it institutionalizes the security improvements.
* **Set Review Milestones:** Before closing this step, decide when you will **check back on these mitigations**. It could be:
  + A quick check-in at the next general meeting (“Did we do what we said for threat X? Is it working?”).
  + A formal review after 3 or 6 months to see progress on longer tasks.
  + Incorporating it into an **annual coop review** or similar. By scheduling a follow-up, you create accountability and also acknowledge that mitigations may need tweaking.

**Participation Tips:** Mitigation planning might involve different sub-groups for different items, but make sure to bring it back to the full coop for transparency. For instance, let’s say a tech group comes up with a plan for backups and multi-sig wallets – they should present it to others in plain language (“We propose to require two people to sign off on any expense over $1000 – here’s how that works... do we all agree?”). Use the coop’s normal proposal-and-decision process to adopt significant changes, so they have legitimacy. Keep records of all these decisions (perhaps in a Security section of your meeting notes).

Also, acknowledge limitations: the group should be aware of any risks that you decide **not** to fully mitigate due to resource constraints. For example, “We know our website could be DDoS’ed (taken down by an overload attack), but at this time we can’t afford a mitigation for that; we accept that risk and will focus on higher priorities.” Being transparent about accepted risks is part of the security culture too.

By the end of Step 7, you have an action plan: who will do what to improve security, and the coop has agreed on these moves. Now the key is to implement and maintain these practices.

**Step 8: Ongoing Improvement and Monitoring (Keep Security Alive in the Coop)**

**Purpose:** Threat modeling isn’t a one-time project – especially in a cooperative that evolves. Step 8 is about **making security a continuous, integrated part of your coop’s governance and operations**. This means regularly monitoring for new threats, reviewing the effectiveness of your defenses, and adjusting to changes, all without losing the cooperative spirit. In short: build a living security program that grows and adapts with your organization.

**Activities/Checklist:**

* **Include Security in Routine Governance:** Treat security as a standing topic in meetings. For example:
  + In your monthly general meeting or weekly check-in, have a brief agenda item like “Security and Privacy Check.” This could be a quick report: “Any incidents or near-misses this month? Any new vulnerabilities found or new tools added that we should consider in our model?”
  + Quarterly or semi-annually, do a slightly deeper review. This could coincide with other reviews (financial, operational). Go over the threat list: have any new threats emerged? (Perhaps the coop started using a new platform – that introduces new risks to consider.) Are there threats that can be downgraded because of improvements? Basically, keep the threat model updated.
  + If you have a rotating security committee, they can take charge of preparing these updates. But even if not, someone can volunteer each time to gather relevant info.
* **Maintain Logs and Audit Trails:** Earlier we emphasized logging and traceability. Make sure those logs (system logs, decision logs, access records) are actually being looked at. For example:
  + If you set up an immutable log of administrative actions (like who changed what settings), maybe once a month a couple of members review it to see if all actions were expected and legit.
  + If you require multi-signature on an account, periodically check that the rule is being followed (no one found a shortcut).
  + Conduct an **internal audit** once a year: pick a couple of key policies and verify they are enforced. E.g., “Check 5 random user accounts to ensure they all have 2FA on as required,” or “Verify that all ex-members in the past year have indeed been removed from our systems within the agreed 24 hours of departure.” Such audits can be done by peers (any member can do it following a script) and results shared openly: “We checked and found 1 of 5 accounts didn’t have 2FA; we helped that member set it up.”

This might sound formal, but it can be lightweight and it greatly helps catch when practices slip. Because coops have no boss checking up, **the members collectively check up in a transparent way**.

* **Periodic Retraining and Onboarding:** Over time, members may forget certain security practices or new people join who weren’t part of the initial threat modeling. Address this by:
  + Creating a short “Security Handbook” or section in your member handbook that summarizes key policies (“We use a shared password manager; here’s how to get access,” “Never share your login links with others,” “Report any lost device immediately,” etc., and also the why – referencing threats like “to prevent incidents like [scenario]” which makes it relatable).
  + Holding a yearly refresher workshop or sending a summary of “Our coop’s top 5 threats and how we mitigate them – reminders for all members.”
  + Simulating a phishing email test or an impromptu drill (with consent) to keep awareness sharp, then discussing the results in a blameless way (“Hey, 3 of us clicked that fake link – let’s go over how to spot those signs!”).
  + Updating onboarding processes: when new members join, include a quick orientation on security norms (for instance, how accounts are created, who to ask for tech help, basic do’s and don’ts). If they know from day one that “security is everyone’s responsibility here,” it sets a good culture.
* **Adapt to Changes in the Coop:** If your cooperative grows from 5 members to 50, or starts a new project, revisit the threat model more formally. New scale or activities can mean new threats:
  + With more members, the risk of insider issues might increase simply by volume, and communication overhead grows – maybe you need more structured processes or automation.
  + With new projects (say you start handling money for members, or you open a second location), add those assets and processes to the model and run through the threat steps again in a lighter form.
  + Schedule a **full threat modeling redo or refresh** perhaps annually or bi-annually. It doesn’t have to start from scratch each time; you can use the previous notes and just update. But having a cycle ensures that, for example, in two years you’re not using the exact same assumptions while the world changed around you.
* **Engage with the Wider Community:** Horizontal coops can learn a lot from each other. Consider sharing non-sensitive parts of your security approach with other cooperatives or participating in forums/discussion groups about security in decentralized organizations. You might discover common threats others have seen, or tools specifically helpful for coops (for example, an open source tool for consensus that has anti-Sybil features, etc.). By contributing your experiences (without giving away your secrets), you also help the movement as a whole become more secure. This external input can be brought back to your coop’s next threat modeling session (“We heard another coop had an issue with X, should we consider that scenario for us?”).
* **Monitor and Respond to Evolving Threats:** Stay informed about general cybersecurity news that might affect you (e.g., a new vulnerability in a software you use – ensure someone updates it; a wave of phishing targeting small nonprofits – maybe alert members to be extra careful). Since no one person is “the security officer,” create channels where such information can be shared. Maybe a #security Slack/Matrix channel or an email list for anyone who finds something relevant. Collective eyes can watch more ground.
* **Governance Feedback Loop:** Evaluate how well the **balance of democracy and efficiency** in your security process is working:
  + After any incident or even after a year, ask: Did our emergency response plan work? Did anyone feel left out of the loop unnecessarily? If so, adjust the protocol.
  + Are members generally comfortable with the security measures or is there grumbling about complexity? If many find it too cumbersome (say the multi-sig process is taking too long for routine tasks), maybe tweak the thresholds or invest in a smoother tool. Always weigh these changes against the risk: maybe it’s fine to ease up if the threat is low, or maybe the group agrees the inconvenience is worth the protection.
  + Essentially, treat the security policies themselves as living documents subject to the coop’s governance. Amend them when needed, just like you would bylaws or other procedures, through the collective decision process.

**Participation Tips:** Keeping momentum is often the hardest part – enthusiasm is high during initial modeling, but can fade. To counter this, make security part of the coop’s **culture**. Celebrate successes: “We successfully thwarted a phishing attempt because Maria remembered the training – kudos!” or “Our audit showed 100% compliance on backups – great job team!” Positive reinforcement shows that these efforts matter and work. Also, distribute the load: rotate who leads the security check-in, who runs the next training game, etc., so it’s not always one person driving it (avoid creating a de facto security czar). The more members have a role over time, the more ingrained it becomes.

By following these eight steps and continually cycling through improvement, a horizontal cooperative can maintain a strong security posture **without losing its character**. Security becomes another aspect of the cooperative’s collective self-management – aligning with the principle that *“we secure everyone by involving everyone.”*

**Mitigation Strategies by Category**

Throughout the threat modeling process, numerous **mitigation strategies** will be identified. These countermeasures can be grouped into three broad categories: **technical controls, process-based measures, and governance/policy measures**. Below is a consolidated list of relevant strategies from the protocol (and supporting documents), organized by type. Use this as a reference or checklist when deciding how to address different threats:

* **Technical Controls:** These involve tools or configurations that directly strengthen the technology.
  + *Distributed Cryptography:* Use multi-signature authentication for critical actions (e.g., require 2 or 3 authorized members to sign off on large fund transfers or important configuration changes) and threshold encryption for sensitive information (so no single person holds a complete decryption key). This ensures no single compromised account can wreck the system.
  + *Strong Authentication:* Enforce two-factor authentication (2FA) for member logins to coop systems. This adds a one-time code or second device check to passwords, making it much harder for attackers to use stolen passwords.
  + *End-to-End Encryption:* Utilize encrypted communication tools (for chat, email, file sharing) so that even if external networks are tapped, the content remains private. For example, require that all confidential discussions happen on a platform with strong encryption (like Signal or Matrix with encryption enabled).
  + *Secure Configuration & Patching:* Keep all software up-to-date with security patches. Remove or disable any default accounts or unnecessary services in the systems you use to reduce attack surface. In practice, designate someone or a team to regularly apply updates and announce when critical patches are done.
  + *Firewalls and Network Security:* Use basic firewalls or access control lists to limit access to servers or admin panels to only what is necessary. If your coop has its own server, ensure it’s configured to only allow trusted traffic (e.g., maybe only VPN or certain IP ranges if applicable).
  + *Intrusion Detection/Monitoring Systems:* If feasible, set up alerts for unusual activity. For instance, get notifications for multiple failed login attempts (which might indicate a brute force attack), or when a new device connects to the network. There are lightweight open-source tools that can be configured for this, or even scripts that email the tech group on suspicious log entries.
  + *Immutable Logs:* Implement tamper-evident logging (possibly using append-only ledgers or blockchain-based logs) for critical actions. For example, each time a high-privilege action is taken (server config change, adding a new admin user, etc.), a record is automatically written to an audit log that cannot be easily altered or deleted. This could be as simple as a Git-based log or a dedicated auditing system. The key is that if something goes wrong, you have a reliable history of events.
  + *Backups and Redundancy:* Regularly back up important data and systems, and store backups in a secure, access-controlled manner. Test the backups periodically. This mitigates threats like data loss (whether from cyber attacks like ransomware or accidents). Ideally, have an off-site or cloud backup that is only writable (not erasable by an attacker who gets into the main system).
  + *Secure Development Practices:* If your coop develops any software or website, adopt secure coding guidelines, code review (at least one other person must approve code changes), and possibly run static analysis or vulnerability scanning tools. This helps catch technical vulnerabilities early.
  + *Password Management:* Use a shared, secure password manager for any credentials that multiple members need (so you’re not sharing passwords over chat or email). This also allows controlled sharing and revocation of access. Educate members to use the password manager for their individual coop-related accounts too, promoting strong unique passwords for all services.
  + *Device Security:* Encourage or require basic device security for member laptops/phones that access coop data – e.g., use device encryption, keep OS updated, and have screen lock/password on devices. One compromised laptop could lead to coop account breaches, so personal device hygiene is part of the technical defense.
  + *Isolation of Privileges:* On shared systems, use separate accounts with least privilege. For example, don’t have everyone use a general “admin” login; instead, have individualized accounts with only the needed rights (and use sudo or privilege elevation only when necessary). Also consider segmented access: someone handling finances only needs access to finance systems, not everything.
* **Process-Based Measures:** These are changes in how people operate or tasks are carried out, to improve security through better procedures and oversight.
  + *Onboarding & Verification Process:* Strengthen the process of adding new members or users to systems. For example, require that new user accounts on critical platforms are approved by a real person and tied to verified member identities (no self-signup without verification). Perhaps have a membership committee verify identities or require new folks to go through an intro session (deterring random fake signups). This mitigates Sybil attacks and ensures everyone with access is accountable.
  + *Offboarding Checklist:* Create a formal checklist for when members leave (or roles change). It should include steps like: remove or disable their accounts on all systems, reallocate any keys or tokens they held, transfer ownership of documents, and collect any devices or cards. Do this quickly (e.g., within 24 hours of departure) to minimize window of vulnerability. Share this checklist so it’s clear and can be followed every time.
  + *Routine Audits and Peer Review:* Schedule regular check-ups of critical settings and logs. For instance, every month have a pair of members (could be a rotating pair) review user access lists: “Are there any accounts that shouldn’t be here anymore? Any permission that looks too broad?” Another example: quarterly financial cross-checks to detect any irregular transactions early. By making it routine, it becomes normal rather than personal (“we always do this” rather than “we don’t trust Bob so we’re checking on him”).
  + *Rotating Duties:* Rotate certain sensitive duties among members to avoid burnout and concentration of knowledge. For example, if one person is usually handling server maintenance, have another member shadow them and take over periodically. Rotation not only prevents a single point of failure (or power accumulation) but also spreads knowledge (making the coop more resilient if someone is unavailable).
  + *Incident Response Drills:* Practice what to do in case of specific incidents. Maybe twice a year, run a short drill: “Today we simulate that our main server is hacked, what do we do?” or “Simulate that a malicious proposal is being pushed through governance, how do we catch it?” These drills help refine your incident response plan (who contacts whom, what steps to take first, etc.) and make sure everyone knows there is a plan.
  + *Communication Protocols for Crises:* Establish clear communication channels for emergencies. For instance, “If something urgent happens, we will send a Signal message to the board and IT group immediately,” or “We have a phone tree if the internet is down.” Write this down and maybe even have a quick reference sheet. This way, if normal channels are compromised (or the threat is that your communications are intercepted), you have a backup method that everyone is aware of.
  + *Member Training and Workshops:* Conduct periodic workshops on topics like phishing, social engineering, and privacy. These can be interactive (show examples of phishing emails and have members spot the red flags, etc.). Encourage members to share any suspicious occurrences (scam emails they got, etc.) so others learn. Create a culture where asking “Hey, is this email legit?” is absolutely okay and encouraged.
  + *Documentation and Knowledge Sharing:* Keep an updated doc or wiki on how to handle common security-related tasks (e.g., “How to update the website safely,” “How to grant a new member access to tools,” “Steps to recover a lost account”). This reduces mistakes and ensures continuity if someone new has to do the task. Well-documented procedures are a security asset because they leave less room for ad-hoc errors.
  + *Backup Communications:* For critical decisions or approvals, consider using a second channel to confirm authenticity. For example, if a financial transaction above a threshold is requested via email or chat, require a quick voice or video confirmation from the requester or a second person’s confirmation. This can catch impersonation attempts (since an attacker might compromise one channel but not a live face-to-face or voice check).
  + *Monitor Community Health:* Keep an eye on social factors – unresolved conflicts, overworked members, or lack of transparency can themselves become security issues (disgruntlement leads to insider threats, etc.). Implement conflict resolution and ensure transparency to maintain trust. In a sense, **good governance is good security**: if people feel heard and issues are addressed, there’s less likelihood of sabotage or neglect.
  + *Continuous Threat Awareness:* Encourage members to report when they see something odd, even if it’s small (like their computer behaving strangely, or they heard a rumor of someone trying to breach another org). Have a non-judgmental reporting mechanism. Better to investigate a false alarm than miss a real one. Make sure people know whom to inform if they suspect a security issue (perhaps a small “security team” alias or just instruct to tell the board confidentially).
* **Governance/Policy Measures:** These strategies embed security into your rules and decision-making structures, ensuring the coop’s governance supports and enforces security.
  + *Collective Access Control Policy:* Adopt a **Collective-Based Access Control (COLBAC)** approach. Instead of one admin granting permissions, make it a group decision or use defined criteria. For example, a policy could state: “To gain access to the financial system, a member must be elected by the coop meeting or at least X members must approve.” This way, no single person can secretly give themselves or a friend elevated access. It formalizes the idea that **power over resources comes from the collective**.
  + *Dynamic Quorum Rules:* If your coop makes decisions by voting, consider rules that defend against low-participation decisions. For instance, require a higher percentage of yes votes if fewer people vote, to avoid a small clique passing something unnoticed. E.g., “At least 50% of members must participate for a policy change vote to be valid,” or “If <30% participate, the vote is extended or escalated.” This helps mitigate a scenario where an attacker or subgroup exploits apathy to slip a bad decision through.
  + *Emergency Powers with Oversight:* As decided in Step 7, have a clear mandate for emergency actions. For example, include in your bylaws or official policies: “In case of a critical security incident, the incident response team is authorized to take immediate protective actions. All such actions must be logged and presented to the membership within 48 hours for review.” This clause gives legitimacy to quick actions and reassurance that they’re not bypassing democracy, just temporally suspending it for safety and then fully reporting.
  + *Transparency Mandates:* Build in requirements that certain information must always be accessible to members. For example, a policy that “All configuration changes on shared infrastructure must be posted in the IT updates forum,” or “All decisions (including security decisions) are documented and available to members by default (except sensitive incident details which can be anonymized).” By mandating transparency, you make secrecy itself against the rules, which deters anyone from attempting to cover up issues.
  + *Role Rotation / Term Limits:* For any positions of trust (e.g., a security committee member, a financial key holder), consider term limits or rotations embedded in policy. E.g., “Tech Steward role rotates every 6 months to a different volunteer.” This ensures knowledge transfer and avoids entrenchment of power.
  + *Enforce Separation of Concerns:* If your coop has sub-committees or circles (e.g., finance circle, IT circle), establish checks and balances between them for security matters. For instance, the IT circle might implement a control but the board or another circle must ratify it for it to stick long-term. Or if an emergency team acts, another group (maybe an oversight committee) must audit their actions post-incident. This prevents even a small emergency group from becoming a permanent power center.
  + *Legal/Compliance Alignment:* Ensure your policies align with any legal obligations (data protection laws, etc.). For instance, if law requires notifying affected parties of a breach, make that a policy too. Or if handling personal data, have a privacy policy that dovetails with your security measures (only collect what you can protect, etc.).
  + *Community Norms Against Risky Behavior:* Sometimes, the coop may want to formally discourage certain behaviors. For example, a policy: “Members should not use public Wi-Fi to access sensitive coop systems unless they use a VPN,” or “No single member should host official data on a personal account.” While these can’t be easily enforced, writing them in a policy sets expectations and can be referred to in peer pressure style (“Hey, our policy says not to do that, for everyone’s safety”).
  + *External Sharing and Transparency:* Decide and document what security information can be shared outside the coop. For instance, you might have a policy that you will publicly disclose any breaches that affect stakeholder data (to be accountable), or that you share your security policy openly (to build trust with members and partners). On the flip side, you might also have a rule that certain sensitive details (like server passwords or personal information) are never to be shared externally. Clarity on this prevents ad-hoc decisions in stressful times and preserves trust.

Each cooperative can tailor these strategies to its context. Some mitigations will be more relevant than others depending on the threats identified. **When implementing, consider combinations**: technical measures often work best with an accompanying process and policy. For example, introducing multi-signature (technical) will require a policy on how signers are chosen and a process for how they coordinate signing. Likewise, a policy of rotating duties (governance) needs a process to hand over credentials and maybe a technical solution to facilitate it.

This categorized list is meant as a **toolkit**. If a new threat comes up, refer here to see what kind of solutions might apply. And remember, every mitigation should uphold or at least not undermine the coop’s democratic and open culture – there’s usually a creative way to solve a security problem *and* strengthen the cooperative ethos at the same time.