What works today, and how to demonstrate it

This is the Capacity rehearsal, not a CPC tenant installation. Octopus runs the application services; Hamster supplies HTTPS through SWAG. Microsoft operates Teams/Entra and file storage. Obot.ai runs the approved demo Microsoft connectors. The Obot.ai Microsoft intermediary must be replaced before CPC production.

A personal Teams request generated a German invitation containing KASTANIE from the brief and an approved synthetic MCP guide. The model received no identity JWT or Microsoft OAuth credentials. Model failure stops drafting without a template substitute.

Workflow	Verified result	Remaining boundary
Teams draft	Real model-generated personal reply	No publishing or file creation
Web approval	Exact reviewed text to synthetic sink	Requester approves own job; externalSend=false
Direct Obot Excel	Worksheet created; six rows written/read	Existing workbook was created through OneDrive UI
Teams workbook link	New source candidate only	Configured owner-only URL; live acceptance pending
CPC rollout	No CPC grants claimed	Tenant, owners, hosting and approvals to agree

- Teams demo: open Casimir Capacity Lab in a personal Capacity chat. Ask: Draft a German office-pilot invitation with codeword KASTANIE.
- Web demo: sign in at the live lab, draft, prepare publication, review exact text, then check confirmation. The submit button stays disabled while unchecked.
- Microsoft demo: connect personally in Obot, use the synthetic existing workbook and inspect the worksheet read-back. Direct Obot writes do not inherit broker approval.

Evidence: Teams real-LLM screenshot, smoke-real-llm.json, microsoft-runtime-read-check.json and microsoft-excel-write-check.json. Listing null was expected for the initially empty account; later worksheet writes have separate evidence.

Expectations for future CPC AI integration

Expected outcomes compared with the verified Capacity rehearsal. These are CPC targets, not completed CPC integrations.

Area	Expected outcome	Lab today / gap
Employee interface	Teams requests; fewer logins and transfers	Teams drafts live. File tools still use Obot separately.
Central administration	IT owns connectors, credentials and access	Local Obot running. CPC tenancy and SSO are not set up.
Access control	Broker enforces user, tool and action rights	Role/resource checks tested. Direct M365 path is separate.
Private context	One Casimir identity; separate user context	Private broker jobs tested. Two-user M365 isolation pending.
Useful workflows	Marketing, tasks, reports, docs; later finance/HR	Draft and Excel-write demos. Tasks, finance and HR absent.
Human control	Kenzie reviews drafts and approves writes	Exact dry-run approval tested. Kenzie flow not integrated.
Reliable automation	Company-run reports reach Teams or email	No recurring delivery yet. Octopus must remain available.
Operational support	Recovery, updates, logs and named owners	SQLite copy checked. Full service / Obot recovery pending.
Adoption	Small pilot, examples, training, then expand	Demo and guide ready. Pilot users and training to agree.

Canonical source: docs/expectations-assessment.json. Runtime evidence and the detailed acceptance plan below determine when a gap can be closed.

CPC replacement map and startup gate

Mike/CPC IT first confirms the tenant, andreas.petersson@cpc.com, MFA, Teams availability, permanent app owners, runtime and pilot membership. Copy the prototype into a CPC-owned environment. Never copy Capacity credentials.

On a fresh checkout only, npm run bootstrap creates random lab keys and .env. It also seeds the Capacity client and URL: replace those values after bootstrap, before startup. Existing .env is protected against overwrite.

Location	CPC value / decision	Owner / verification
.env	ENTRA_CLIENT_ID, PUBLIC_URL; fresh workload/bootstrap keys	CPC IT; inspect non-secret /api/config
config/policy.json	CPC tenant, pilot object IDs, roles/resources, new version	IT + business data owner
Teams manifest/runtime file	CPC bot ID, URLs, personal scope and bot secret	IT; install and receive personal reply
Compose + SWAG	LAN binds, hostnames, HTTPS upstreams, volumes and isolated backend	Runtime owner; public health/routing checks
Inference + connectors	Approved endpoint/model, credentials and resource IDs	IT/data owner; real draft and allowed/denied tool checks
Document-link configuration	CPC resource, owner, HTTPS file URL and host validation	Data owner; candidate code needs CPC adaptation

- Use package.json Node >=24.14 and <25; run npm ci, npm run build and npm test in the CPC checkout.
- Provision the external Obot volume deliberately. Its lab name is not the CPC storage plan. Create the required secret files on the runtime host.
- Before pilot access, /api/config must show the CPC tenant, app, API scope and HTTPS redirect. Test CPC sign-in, wrong-tenant denial, a real draft and personal Teams delivery.

The lab runtime apps have empty requested-Graph-permission and owner lists in the recorded Graph inventory. Add permanent named CPC owners; app ownership allows credential changes. Native Obot Entra login remains a separate directory-permission/licensing decision.

Provider, secret files and runtime prerequisites

config/inference.json selects provider, baseUrl, model, output limit and timeout for the OpenAI-compatible chat endpoint. Earlier local MLX generation used an existing Octopus process outside Compose; fresh Compose does not provision it. The DeepSeek template is prepared but is not active or authenticated.

For a credentialed provider, create .secrets/llm-runtime.env with LLM_API_KEY. The broker optional env_file wiring is now validated. A key in .env alone is insufficient unless passed into the broker. Endpoint/model edits hot-apply because config is read per request; credential changes require docker compose up -d --no-deps --force-recreate broker.

Required file	Contents / setup	Custodian
.env	Fresh workload/demo/bootstrap keys and CPC app/URL	CPC runtime owner
.secrets/teams-runtime.env	TEAMS_CLIENT_ID and TEAMS_CLIENT_SECRET	CPC bot owner
.secrets/obot-encryption.yaml	Vetted Obot v0.26 EncryptionConfiguration with new AES-GCM key	IT key custodian; separate recovery copy
.secrets/llm-runtime.env	LLM_API_KEY only when approved provider needs it	Model-service owner

- Restrict secret files to the runtime owner (mode 0600); do not print expanded docker compose config, tokens or secret contents in a screen share.
- On a fresh installation only, run node scripts/init-obot-encryption.mjs. It exclusively creates the vetted Obot v0.26 AES-GCM configuration with a new key, mode 0600, and refuses an existing file. Back up that key separately; do not use this helper for rotation. Bootstrap does not create the Teams or model-provider files.
- When approved provider credentials are present, run node scripts/check-inference.mjs before activation. It prints only synthetic-probe status, provider, model and latency. Require a successful broker draft and Teams reply afterward; unavailable provider must produce no substituted draft.
- If DeepSeek is chosen and its approved API key is present, run node scripts/check-inference.mjs config/inference.deepseek.example.json. This tests the template without changing live config. Only after success, copy the chosen template to config/inference.json; recreate the broker when its credential file changes, then verify broker and Teams results.

Latest local MLX probe timed out. The prepared DeepSeek API template uses https://api.deepseek.com, deepseek-flash, disabled thinking and a 30-second timeout; it has no validated key and is not active. Record the chosen provider, destination/data decision, model and fresh successful probe/broker/Teams results before claiming readiness.

Sources: [10], [19], [20], [21]

Human control: the Kenzie workflow

The current web lab binds approval to the exact input hash, requester, policy version and ten-minute expiry. It writes only to a synthetic sink. A requester approves their own private job; Kenzie cannot currently approve another user's job merely by having a reviewer role.

CPC target: requester supplies brief and destination; Casimir prepares the draft/change; Kenzie reviews the exact version and returns changes or approves; the permitted adapter executes once; Teams returns a receipt and the actual resource link. Mike and the business owner must name Kenzie's Microsoft account, review scope and replacement reviewer.

Gate	CPC target	Acceptance
Review visibility	Kenzie sees only assigned review payloads	Separate requester/reviewer accounts; unrelated private jobs remain hidden
Revision	Changes create a new exact version	Previous approval cannot authorize edited text, destination or arguments
Approval	Explicit confirmation of exact write	Button disabled unchecked; denied after expiry or policy change
Execution	User/resource authority checked again	One write on replay; receipt names destination and result

- This separate-review workflow needs implementation; owner-only private-state checks must not be bypassed to make it work.
- Keep payroll changes, employee creation, arbitrary shell, credentials and mail sending denied until a separately scoped workflow exists.
- For WordPress/Shopify changes, review the proposed diff and backup/rollback point before approval. Drafting is not permission to publish.

Microsoft sign-in, tenant/app ownership, Teams availability, upstream file rights, broker roles and human review are separate gates. The model cannot choose an identity, resource or authority.

Microsoft files: the exact remaining integration

Personal Excel/OneDrive OAuth is complete in Capacity. The consent showed Acorn Labs Integrations, an unverified publisher, with organization-wide delegated offline_access, User.Read, Files.ReadWrite and Files.ReadWrite.All. Actions use each signed-in user's existing rights; this is not CPC consent or app-only access to every tenant file.

Managed vMCP vmcp1rmdls exposes six Excel list/read/worksheet/row/column tools and four read-only OneDrive tools. Delete and sharing tools are excluded. A blank workbook was created through authenticated OneDrive UI; Excel MCP created a worksheet, wrote six synthetic rows and read them back.

The chosen vendor MCPs do not expose new-workbook create/upload. Teams-to-file completion therefore needs a create/upload adapter, user-bound upstream credentials, typed folder/workbook/action rules, reviewed writes and a returned file link. Direct Obot writes remain a separate path.

- Open Obot, select the vMCP and Connect personally. Administrative discovery OAuth and each user's runtime connection are separate steps.
- Test real users A and B: A's unshared workbook must be denied through B's connection. Verify job/approval isolation and then revoke B's connector/consent.
- The new Teams link source is a configured owner-only lookup, not file search, file creation, anonymous sharing or broker-delegated Graph access. Keep it marked pending until live reply and denial evidence exists.
- For CPC production, replace the Obot.ai Microsoft intermediary with the agreed CPC-controlled path; repeat final scopes, grants, resource, isolation and revocation checks. A different vendor remains its own explicit CPC decision.

Record Microsoft write evidence separately from Teams evidence. Teams drafting plus Obot Excel editing does not prove an end-to-end Teams file workflow.

Sources: [11], [12], [13], [14], [15]

Register an MCP, then authorize a broker workflow

An Obot connection and a broker capability are two registration steps. Adding an MCP URL, package or CLI definition does not automatically authorize a Teams action. Begin with one synthetic resource and a read operation.

- Obot/vendor step: identify the actual MCP endpoint or pinned hosted stdio package; record operator, destination, credential owner and user/service identity model. Review requested upstream rights.
- Select Managed tools, remove delete/sharing or unneeded writes, and scope all matching access profiles to the pilot. Profiles are additive. Create a stable vMCP endpoint and complete each required personal connection.
- Verify an allowed read and a denied resource using the intended user. Save non-secret evidence of tool names, resource, outcome and consent/revocation decisions.
- Broker step: register the reviewed connection in config/connectors.json; implement a typed adapter and strict argument schema; add explicit tool/role/resource rules in config/policy.json and increment its version.
- Bind credentials and resource selection to trusted actor context. Add exact review for writes, replay/reconciliation handling, and deterministic Teams routing. Record an actual Teams-to-adapter result before claiming integration.
- CLI alternative: package the required executable in an isolated runtime; use fixed executable/argv, reviewed environment and working resources, output/time limits and broker audit. Never let the prompt supply arbitrary shell or a full user home directory.

The current lab CLI is a fixed read-only inspection. WordPress, Shopify, Amazon and other CLI integrations below are planned examples; no connection or write is claimed.

Sources: [11], [14]

Planned resources, credentials and owners

These are proposed registration examples, not connected CPC systems. For every connector, name the upstream account/resource, credential custodian, allowed tools, business reviewer and rollback owner before the first write.

Service	Resource and credential boundary	Owner / first reviewed workflow
WordPress	Named site and allowed page/post IDs; dedicated user capabilities and approved application credential	Website owner; read content, draft exact diff, take backup/revision point, approve then apply and verify
Shopify	Named store and resource IDs; installed app with only the approved Admin API scopes	Store owner; read-first, review product/content diff and export/backup point before any write
Amazon selling	Seller/Vendor account, marketplace and approved SP-API roles; selling-partner authorization	Marketplace owner; start with read/reporting. Seller Central/SP-API access is separate from AWS account access
Call reports	Named call/report source and approved records; separate connector and access decision	Report owner; agree collection, allowed recipients and retention. No automatic call ingestion or delivery is implemented

- A storefront or portal login is not evidence that its API credentials/scopes are approved. Verify the installation's actual rights and a denied resource.
- Weekly reviews/report delivery currently depend on separate VM processes. A CPC company runtime needs a scheduler owner, recipient allowlist, duplicate handling and a test while the user's VM is off.
- Keep recurring reports read-only first. Publishing, sending, finance and HR are separate workflows with their own authority and review decisions.

Primary platform documentation is linked in References. The connector choice, specific scopes and production destinations remain CPC decisions.

Sources: [16], [17], [18]

Acceptance with outcomes and named owners

Mike assigns an owner and evidence record for each gate below. Existing 14 live smoke checks use loopback synthetic personas; they are not two real CPC/Microsoft users. Do not mark a planned workflow passed from a unit test or successful OAuth alone.

Check	Expected result	Owner / evidence
Identity	CPC pilot succeeds; wrong tenant/non-pilot denied	IT; real sign-in and personal Teams checks
Two users	B cannot read A's private job, approval or unshared workbook	IT + data owner; two real connections
Human approval	Unchecked/changed/expired/cross-user/replayed request cannot cause unintended write	Kenzie/business owner; reviewed version + receipt
Provider failure	No template, substituted draft or publication	Model/runtime owner; live failure and recovery
File workflow	Approved resource written once; link opens for intended owner; read-back matches	Data owner; actual Teams-to-file evidence
Revocation	Removing pilot/connector/consent removes intended access	IT; before/after access result
Restart/recovery	Jobs persist; uncertain writes reconciled; clean-host restore usable	Runtime/key owners; recovery log
Scheduled delivery	Approved recipient receives one report with personal VM off	Report owner; schedule/run/delivery receipt

- Record runtime/image/config versions, timestamp, actor/resource class, result and evidence location without tokens or private file contents.
- After passing the chosen small pilot workflow, remove temporary setup roles/consent and agree a maintenance review date. Expand only after its next workflow passes these gates.

Immediate handoff decisions: CPC tenant/account/owners; Kenzie's review identity; pilot users and test resources; model/hosting; connector replacements; backup/recovery custodians. These remain unfilled decisions, not assumed grants.

Operations, backup and clean-host recovery

The broker stores durable SQLite WAL jobs and audit metadata. A tested consistent snapshot was opened read-only in a network-isolated container with integrity OK, 13 jobs and 4 synthetic publications. Restart preserved state. This is database-copy evidence, not a full clean-host application or Obot recovery.

An executing job becomes unknown on restart. Pause real writers and reconcile against upstream receipts before replaying it. The current prototype is not a production queue worker and does not automatically resume every queued job.

- Daily check: docker compose ps, harness /healthz, model reachability, authenticated web/Teams request and the intended personal Obot tool. Name the on-call owner and escalation contact.
- Run node scripts/backup-broker.mjs for a consistent snapshot and isolated database check. Keep dated configuration/image versions with backups; copy encrypted backups off-host under the assigned owner.
- Clean-host broker exercise: stop/quiesce writers, provision isolated storage, restore the selected snapshot with correct ownership, deploy the recorded image/config, start services, verify jobs/audit and execute one safe read. Keep real writes disabled until reconciliation finishes.
- Obot exercise is separate: back up its supported database/volume and recover the matching encryption configuration/key from separate custody. On a clean isolated host, verify login, restored connections and a permitted tool before declaring recovery passed.
- Assign bot, connector and model credential rotation dates. Selected-field Obot encryption does not cover every record, log, backup or export.
- Company-run automation needs a hosted scheduler, job owner, allowed recipients, failure notification and recovery test; no recurring delivery is currently claimed.

Verified direct Obot Excel worksheet creation, six-row write and read-back are recorded in microsoft-excel-write-check.json. Teams file creation, Kenzie review, two-user isolation and full Obot/key recovery remain open.

Sources: [10]

Mike: what the lab actually needed

Most engineering work can be delegated through ownership of Casimir's specific app registrations. Mike can retain CPC tenant consent, Teams rollout and data approvals. The lab was built in capacity.at as andreas@capacity.at; it is not a permission grant in CPC.

The single-tenant web app uses delegated access_as_user for its own API and has no runtime Microsoft Graph permissions; default User.Read was removed. The Teams bot is personal-chat only, with no manifest resource-specific permissions or file support. A German Teams invitation containing KASTANIE was verified with the approved synthetic MCP guide and real local MLX generation. Teams SSO is absent.

Setup tooling is a separate authority: Teams CLI 3.0.3 requests AppDefinitions.ReadWrite from Teams Developer Portal and delegated Graph Application.ReadWrite.All for secret creation and app inspection. Its doctor report demonstrates that a Graph token was available. This is not a Graph permission on the Casimir runtime app. When and by whom the CLI grant was consented has not been verified.

Obot currently uses a local Owner account. Native Entra login and its directory reads have not been granted. Docker administration on Octopus and SSH root access for hamster SWAG were used; Obot's Docker socket can control the host daemon. Earlier Teams generation used Octopus MLX Serve outside Compose (Qwen3.8-Flash-Next-Abliterated-MLX-Serve-mixed-4-8bit). The latest probe timed out; DeepSeek awaits a validated key. Only the brief and synthetic guide reach inference, not identity JWTs or Microsoft OAuth credentials; the template generator and fallback were removed.

The broker matrix is explicit code/configuration, currently binding the verified Capacity user to pilot, marketing, reviewer and operator roles. Sensitive payroll, employee-creation, shell, credential and mail tools are denied. Publication is a synthetic dry run. CPC group integration is proposed, not configured.

- Actual account directory role and CPC licensing remain unverified. Graph inventory confirms both lab apps request no Graph permissions and have empty owner lists; add permanent named owners before rollout. Portal access does not establish Global Administrator status.
- Lab web/API client: b1785104-df20-4bec-81ba-52e97d886be1. Bot/client: 71a69aff-c8fe-4ba5-a0a0-29270f9c9b0e.
- Fourteen current source checks pass, including four link-candidate checks. Earlier fourteen live smoke checks passed. A consistent SQLite copy preserved thirteen jobs and four synthetic publications with integrity OK. Full service and Obot/key recovery remain pending.
- Safe basis: Compose, policy, Teams manifest/doctor, installed CLI source and verified UI evidence. No credential values inspected.

Sources: [5], [6]

A practical delegation to Andreas

Use andreas.petersson@cpc.com. Mike creates the agreed registrations, keeps a permanent CPC owner and makes Andreas owner of only those registrations. App ownership includes credential changes, so it is sensitive. Add ownership of the corresponding enterprise apps separately only if Andreas needs to manage pilot assignments.

If ordinary users cannot create registrations, a temporary Application Developer role enables creation as owner. It is a privileged role and also enables certain self-consent. Cloud Application Administrator and Application Administrator manage applications broadly across the tenant; they are not the default requirement for this pilot. Application Administrator also covers Application Proxy.

Mike or another CPC Cloud Application Administrator/Application Administrator can approve tenant-wide delegated consent. Graph application permissions require Privileged Role Administrator or a suitably authorized custom role. Registration ownership alone does not confer tenant-wide consent authority.

Andreas can develop and upload the personal Teams app when his custom-upload setup policy allows it. Mike/Teams Administrator retains app availability and organization rollout. Manage pilot availability through the tenant's app-centric management or existing app-permission policies. A normal pilot security-group owner can manage membership, but cannot approve OAuth scopes.

Enterprise-app/service-principal ownership can allow assignment administration. Group-based enterprise-app assignment needs Entra P1/P2; verify CPC licensing or use individual assignments. Data owners share only the dedicated test workbook/folder. File sharing and consent are separate gates.

Mike-led alternative: Mike performs portal setup and creates/seeds bot credentials in CPC's secret store; Andreas provides exact names, redirects, endpoints, scopes, manifest and user list, then handles code and deployment. This avoids giving Andreas a broad administrator role or approving a new broad CLI grant.

- Time-limit any creation or administrator role; use eligible PIM activation if licensed, otherwise record and remove assignments manually.
- Keep a separate CPC Owner in Obot. Pilot users do not need Obot administration or Docker/root access.
- After acceptance, remove unnecessary setup-tool consent and temporary roles; keep only agreed maintenance ownership with a review date.

Sources: [1], [2], [3], [4], [7], [8]

Microsoft tools: approvals and remaining work

For the Capacity demo, Andreas approved Obot.ai as the Excel/OneDrive intermediary, with replacement required later. Microsoft displayed Acorn Labs Integrations, an unverified publisher, and Consent on behalf of your organization was checked. The rights map to offline_access, Files.ReadWrite, Files.ReadWrite.All and User.Read. Admin/discovery and personal runtime OAuth completed for both Excel and OneDrive. OneDrive drive listing succeeded and Excel workbook listing succeeded with a null result. Excel MCP created a worksheet, wrote six synthetic rows and read them back. Two-user isolation remains untested.

This is organization-wide delegated consent, not app-only access to every tenant file. Actions still use an authenticated user's rights, including shared files. It is broader than an Andreas-only grant and is not CPC approval. Managed vMCP vmcp1rmdls exposes six Excel list/read/worksheet/row/column tools and four OneDrive read tools; delete and sharing tools are disabled. The final Entra grant and app ID still need inventory.

The direct Obot Microsoft playground and typed broker workflows are separate paths today. Direct Excel writes do not inherit the broker's marketing approval or resource rules. CPC implementation must integrate per-user credentials, tool and workbook/folder authorization, review of writes and revocation. Keep deletion and file-sharing changes disabled until reviewed.

Native Obot Entra login is another decision. Current v0.26 documentation lists delegated User.Read and ProfilePhoto.Read.All plus application GroupMember.Read.All and User.Read.All for directory/group lookup. None of those additional Obot directory grants exists in this local-owner lab. Mike should review them separately; Community registration or Enterprise licensing is also required for Entra login.

The handoff should record each app/tool's owner, permissions by type, consent actor, publisher, destination and expiry. CPC owns identities, pilot membership, upstream file access, data-processing approvals, secrets and rotation. Replacing the vendor endpoint requires repeating the OAuth and isolation tests.

- Test two Microsoft users, denied workbook/folder access, private context and cross-user token/job/approval isolation.
- Verify wrong-tenant/unassigned users are blocked and removing membership/assignment removes access.
- Verify changed, unchecked, expired, repeated and cross-user approvals cannot execute an unintended write.
- Test consent/connector revocation, credential rotation and restoration of state plus encryption keys.

Sources: [4], [9], [10], [11], [12], [13]

Primary references and evidence

Official sources support permission, provider and planned connector decisions. Rehearsal evidence is separate from CPC acceptance; successful OAuth or a source change alone does not prove a workflow.

[1] Microsoft Entra: delegate app administration

[2] Microsoft Entra: Application Developer role

[3] Microsoft Entra: assign enterprise-app users/groups

[4] Microsoft Entra: grant tenant-wide admin consent

[5] Microsoft Graph: Application.ReadWrite.All permission

[6] Microsoft Graph: application addPassword

[7] Microsoft Teams: custom app policies and settings

[8] Microsoft Teams: app-centric management

[9] Obot: Entra authentication provider

[10] Obot: encryption providers

[11] Obot: MCP access policies

[12] Obot catalog: Microsoft Excel connector

[13] Obot catalog: Microsoft OneDrive connector

[14] Obot: virtual MCPs

[15] Obot: Microsoft Office 365 MCP source

[16] WordPress: REST API authentication

[17] Shopify: API access scopes

[18] Amazon: selling-partner application authorization

[19] Obot: custom encryption provider configuration

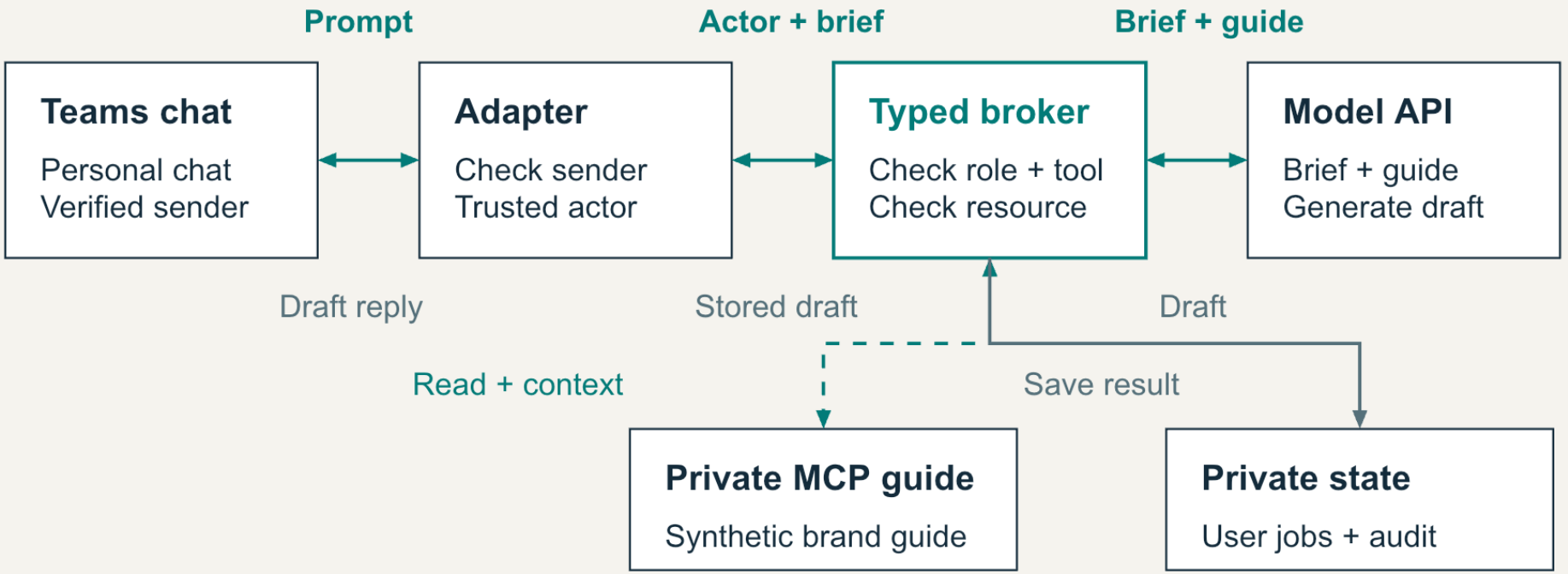
[20] DeepSeek: model/pricing reference

[21] DeepSeek: thinking mode

Safe evidence: real-inference smoke and Teams reply; personal Microsoft OAuth/listing; microsoft-excel-write-check.json; consistent broker snapshot check; read-only app/owner inventory. Record fresh runtime evidence after a provider, policy or connector change. No credentials are reproduced.

Today’s Teams prompt and draft flow

Adapter, broker and MCP context: Octopus. Inference: Local MLX Serve on Octopus.

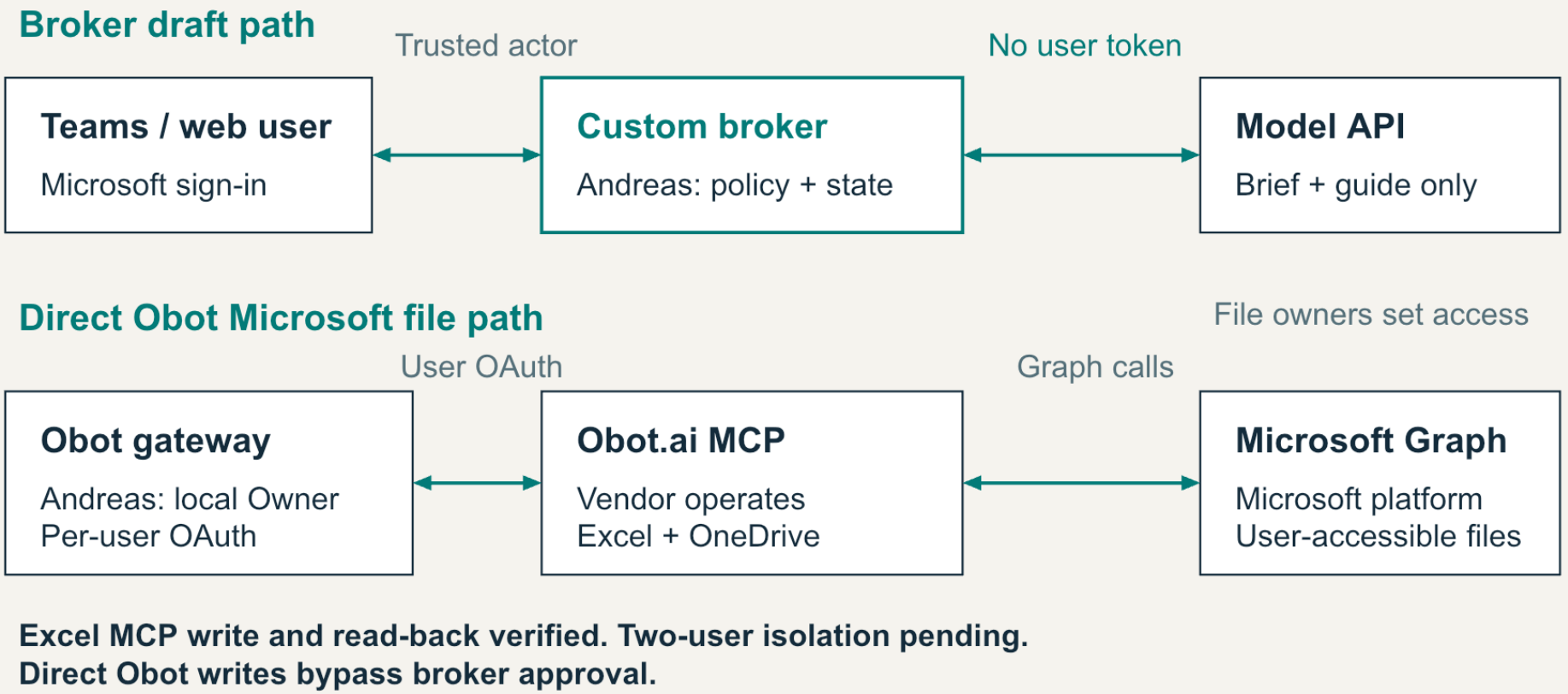


Andreas controls policy, compute and state. Microsoft operates Teams and Entra.

Capacity controls the adapter, broker policy, approved context and private state on Octopus; Microsoft operates Teams and Entra. Only approved prompt data reaches inference. The current draft path does not create Microsoft files.

Today’s Microsoft files and credential boundaries

HTTPS: Hamster / SWAG. App compute: Octopus. Inference follows configuration.



Excel MCP write and read-back verified. Two-user isolation pending.
Direct Obot writes bypass broker approval.

Direct Obot worksheet creation, six-row write and read-back succeeded on a UI-created blank workbook. User/file owners govern access. Direct writes bypass broker review; Teams file creation and two-real-user isolation remain pending.

CPC rollout ownership and resource access

Proposed setup. CPC has not granted these permissions.

Mike / CPC IT

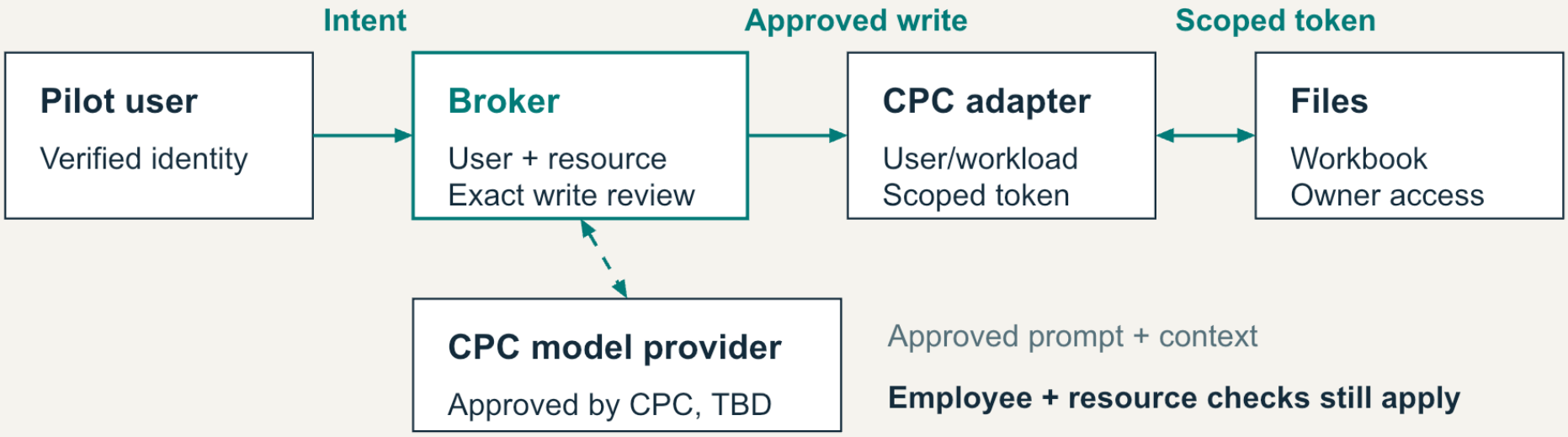
Consent + app setup
Keys, runtime, audit

Business / data owner

Pilot roles, resource allowlist
Review and write approvals

Andreas

Implementation and deployment
Temporary named-app ownership



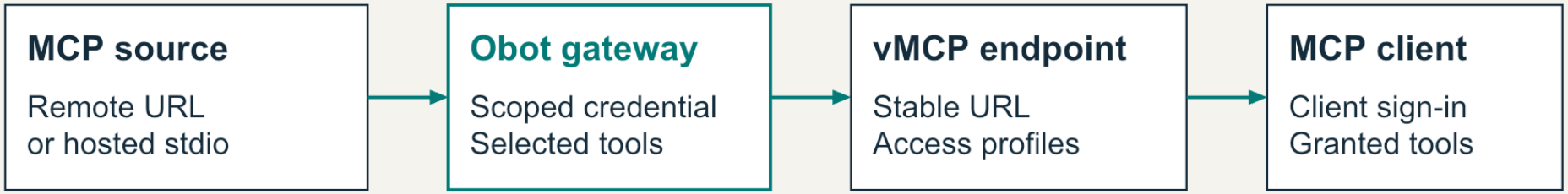
Obot.ai Microsoft connectors must be replaced before CPC production.

Proposed: Mike/CPC IT owns tenant consent, hosting, keys and model decisions; business owners define resources/review; Andreas implements through agreed named-app ownership. CPC has granted no access in this rehearsal.

WordPress, Shopify and Amazon connections

Approved MCP or CPC-owned API adapter. User / service credentials stay in the connector.

Obot registration and client access



Planned CPC resources

WordPress

Approved site + pages
MCP or API adapter

Shopify

Named store + Admin scope
MCP or API adapter

Amazon Seller Central

Seller account + marketplace
Selling Partner API adapter

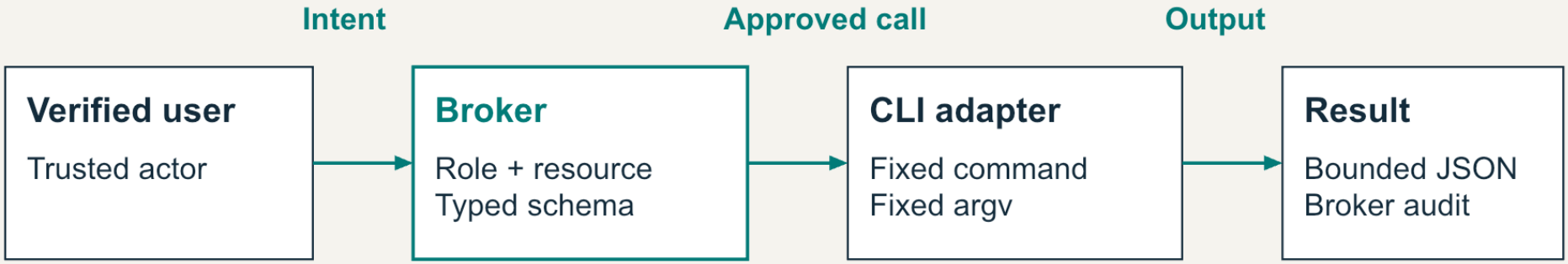
Broker route: connectors.json + typed schema / adapter + policy.json role and resource rules
Configuration alone adds no authority. Teams currently runs fixed workflows.

Verified today: private knowledge read and Microsoft Excel read / write

Planned registration path: reviewed MCP endpoint/package, upstream credential owner, selected tools and pilot access profile. A separate typed broker adapter/schema plus role/resource policy is required before Teams may invoke it.

A reviewed CLI through Casimir

Authenticated user. Typed authorization. Fixed executable and arguments.



Execution controls

shell:false
Minimal environment
5 s timeout and 32 KiB cap

Third-party CLI integration

Reviewed executable / API wrapper in container
User / service credentials + resource guard
No arbitrary shell or full host HOME mount

Today: cli.inspect uses a synthetic adapter. Vendor CLI connections remain planned.

The model chooses neither actor nor credentials. Tokens do not pass to the CLI automatically.

The verified lab inspection uses fixed executable/argv and bounded output. Future CLIs need packaged dependencies, reviewed user/service credentials and explicit resource guards. Arbitrary shell remains denied.