Remote Login App — Design Notes

Overview

This proof of concept demonstrates a secure, disposable remote desktop session that allows a user to log in to a third-party web service without exposing their credentials to the backend. After login, the system extracts, encrypts, and stores the authentication cookies, then destroys the session VM and its subdomain.

Key Architecture Components

Frontend

- Lets the user create a new session via a simple UI.
- Connects to an isolated VM desktop using noVNC over HTTPS.
- Lets the user extract past and current session cookies.

Backend API (FastAPI)

- `POST /session` : Creates an ephemeral VM, sets up a Cloudflare subdomain, and configures HTTPS routing.
- `DELETE /session/{session_id}` : Tears down the VM and its DNS record.
- `GET /extract_cookies` : Uses Chrome DevTools to extract cookies securely, encrypts them, and saves to MongoDB Atlas.
- `GET /cookies` : Lets the user fetch their encrypted cookies with a session ID and token.

VM Environment

- Oracle Cloud VM launched with a custom script.
- Runs Xvfb + LXDE, Google Chrome with remote debugging, noVNC, and Caddy as a reverse proxy.
- Python script runs a FastAPI server which extracts cookies and returns them to the backend when invoked by it.

Infrastructure Automation

- Cloudflare handles DNS, HTTPS, and proxying.
- DNS records and VMs are ephemeral and destroyed on session end.
- Background process enforce auto termination after 15 minutes.

Security Measures

- **No credential capture**: All credentials stay inside the VM; only cookies are extracted.
- **Encryption at rest**: Cookies are encrypted using Fernet keys, stored separately from DB config.
- **End-to-end TLS**: Cloudflare proxy ensures all traffic uses valid HTTPS.
- **Ephemeral sessions**: Nothing persists after VM teardown.
- **Access controls**: CORS and ingress rules (On Oracle cloud) limits who can hit the API.
- **Rate limiting**: Protects against abuse of VM creation.

Observability

- Logs for session lifecycle, DNS actions, VM events, and cookie extraction.
- Validation script logs key steps.
- Cloud dashboard confirms ephemeral resources.
- Could expand with metrics for active sessions and usage.

Design Trade-offs

Aspect	Decision	Trade-off
VM vs. Container	VM for simpler PoC	Containers are faster to start and stop, but add networking complexity (overlay, orchestration).
Cloud Provider	Oracle cloud infrastructure	Free or lower-cost credits for PoC. AWS or GCP might have more automation tools but higher cost.
Database	MongoDB Atlas	Easy JSON storage for unstructured cookie data; a SQL DB could add stricter schema and ACID guarantees but more overhead.

Aspect	Decision	Trade-off
Remote Access	noVNC over HTTPS	Easier than setting up WebRTC or SSH tunneling, slightly higher latency than native solutions.
Cookie Extraction	Chrome DevTools	Simple, reliable and easier than hard-coding open tabs in script. Headless scraping tools or proxies could offer more stealth but add complexity.
Dynamic DNS	Cloudflare subdomains	Rapid SSL + DNS management, custom certs with static IPs could remove propagation delays but are harder to automate.
Auto Termination	Handled by backend background processes	Could use instance metadata or serverless for reliability

Alternate Approaches

- **Containers for faster boot and teardown**
- **WebRTC instead of VNC** for better UX
- **Secrets Manager for keys** instead of local config

Summary

Disposable VMs, ephemeral DNS, and strong encryption ensure we can capture and store session cookies securely without ever handling user credentials.