Technical Architecture and Implementation Roadmap for the SwissWorkspaceGateway: A Native Termux-First Remote Environment

Executive Summary

The SwissWorkspaceGateway represents a sophisticated architectural response to the requirement for localized, isolated, and highly secure remote desktop environments hosted on high-performance Samsung Android hardware. In the current landscape of mobile computing, the intersection of Android's sandboxed environment and the versatility of the Termux terminal emulator provides a unique opportunity to deploy professional-grade workspaces without the overhead of cloud-based Virtual Private Servers (VPS). This project specifically targets the deployment of two operationally isolated workspaces (Workspace A and Workspace B) on a Swiss-based Samsung tablet, ensuring that all computing remains on-device while being accessible through standard web browsers on remote machines via noVNC.

The core technical challenge involves navigating the lack of traditional Linux primitives such as systemd and Docker while maintaining session persistence and isolation. Through an exhaustive evaluation of contemporary Termux graphical strategies, this report identifies the PRoot-distro approach as the superior methodology for a Minimum Viable Product (MVP), primarily due to its support for standard Linux Filesystem Hierarchy Standards (FHS) which are critical for running modern web browsers. The architecture utilizes Tailscale for zero-trust network access, ensuring that the tablet's attack surface is minimized by binding all graphical and session services to the local loopback interface. Automation is achieved through a suite of idempotent bash scripts and a task-orchestration layer managed by the Cloudy CLI, facilitating streamlined deployment and lifecycle management.

The following analysis details the architectural trade-offs, package requirements, security configurations, and automation logic necessary to realize a robust and reliable remote workspace on the Android platform. It further addresses critical performance hurdles, notably the Android Phantom Process Killer, providing actionable workarounds to ensure long-term stability in a production environment.

Comprehensive Evaluation of GUI and Session Strategies

The selection of a graphical delivery mechanism is the most pivotal decision in the

SwissWorkspaceGateway architecture. The unique constraints of the Android kernel—specifically the absence of native X11 support and the reliance on the Wayland-based SurfaceFlinger—require a translation layer to bridge the gap between Linux desktop applications and the Android display subsystem.

Architectural Strategy Comparison

Feature	Approach A: Termux:X11	Approach B: PRoot-distro	Approach C: Headless Browser
Primary Mechanism	Native X Server / Wayland Compositor [1]	User-space Syscall Translation (ptrace) [2]	Chromium DevTools Protocol (CDP) [3]
Rendering Target	Android Activity / VNC capture [4]	VNC Server (TigerVNC/Xvnc) [6]	Virtual Framebuffer / Bitmap [3]
System Compatibility	Termux-native (non-FHS) [7]	Standard Linux (FHS-compliant) [8]	Minimalist Node/Python [3]
Isolation Capability	Environment variable based (Shared) [11]	Strong (Separate rootfs or users) [8]	Strong (Process-level isolation) [13]
Browser Support	Limited (Otter, TUR Chromium) [14]	Full (Firefox, Chromium, Chrome) [9]	Native control via Puppeteer [10]
Hardware Acceleration	Native via VirGL / Turnip [9]	VirGL bridge required [9]	Limited for remote streaming
Automation Ease	Moderate (Complex X11 setup) [19]	High (Apt and standard shell) [9]	High (Native Node.js/Python) [3]

Analysis of Approach A: The Termux:X11 Paradigm

The Termux:X11 approach is widely regarded as the most performant method for running local GUI applications on Android. It operates as a Termux plugin app that implements a minimal Wayland compositor, allowing X11 applications to render with near-native efficiency.[1] For the local user, it offers the lowest latency and the best support for hardware-accelerated 3D rendering via Turnip and VirGL.[17]

However, for a remote workspace accessed via noVNC, Approach A introduces significant complexity. Termux:X11 is primarily designed to render to the tablet's physical screen.[5] To expose this to a remote browser, one must implement a complex chain where a VNC server (such as x11vnc) attaches to the existing X session and converts the frame buffer into a VNC stream.[4] Furthermore, because Termux does not follow the standard Linux Filesystem Hierarchy Standard (FHS), many desktop-grade browsers fail to run natively in this mode without extensive patching of hardcoded paths.[7] While the Termux User Repository (TUR) offers Chromium, its stability varies significantly across Android versions, and the lack of a system-wide dbus often leads to degraded browser functionality.[15]

Analysis of Approach B: The PRoot-distro Methodology

The PRoot-distro approach serves as the traditional backbone for running full Linux environments on non-rooted Android devices. PRoot utilizes the ptrace system call to intercept and manipulate system calls, effectively tricking binaries into believing they are operating within a standard Linux filesystem structure.[2] This compatibility is the critical differentiator for the SwissWorkspaceGateway, as it enables the installation of standard apt packages for browsers like Chromium and Firefox without the need for manual patching.[9]

Isolation in Approach B is naturally achieved through the creation of distinct PRoot instances or separate user accounts within a single instance. This allows for Workspace A and Workspace B to have entirely separate configuration directories, browser profiles, and application states.[8] By running two instances of Xvnc (part of the tigervnc suite) on different display numbers (e.g., :1 and :2), the architecture provides discrete graphical sessions for each workspace.[6] While ptrace introduces a measurable performance overhead, the stability and package ecosystem of a Debian-based PRoot environment make it the most reliable choice for a professional MVP.[8]

Analysis of Approach C: Headless Browser and Streaming

Approach C represents a modern, minimalist alternative where no full Desktop Environment (DE) is installed. Instead, a headless instance of Chromium or Firefox is launched, and its window is streamed via a virtual frame buffer or directly controlled through the Chrome DevTools Protocol (CDP).[3] Tools like Puppeteer or Playwright can be used to manage these browser instances programmatically.[10]

While Approach C is highly efficient in terms of CPU and RAM usage, it often lacks the "workspace" feel requested by the user. Most operational tasks require file management, terminal access, and the ability to interact with multiple windows—features inherent to a DE but absent in a pure headless browser stream. On Android, managing the bridge between a headless browser and a browser-based VNC client (noVNC) requires a virtual X server like Xvfb or Xvnc regardless, effectively merging this approach with the complexities of Approach B without the benefit of a full user interface.[10]

Detailed MVP Plan: The PRoot Debian Architecture

The selected MVP architecture for the SwissWorkspaceGateway is based on **Approach B (PRoot-distro with Debian)**. This path offers the highest degree of reliability for browser-based tasks and provides the most straightforward implementation of multi-workspace isolation.

Architectural Component Diagram

+---+

| Android Tablet (Samsung S-Series) | |
+---+ | | | Termux Environment (Host) | | | | - Tailscale Node (Userspace Networking) | | | | - Cloudy CLI (Automation Runner) | | | | - Websockify Proxy A (Port 6080 -> :1) | | | | - Websockify Proxy B (Port 6081 -> :2) | | | | - PulseAudio Server (Audio over TCP) | | |
+---+ | | | PRoot Environment (Debian) | | | |
+--------------------------+ +-----------------------+ | | | | | Workspace A (:1) | | Workspace B (:2) | | | | | | - Xvnc (TigerVNC) | | - Xvnc (TigerVNC) | | | | | | - XFCE4 Session | | - XFCE4 Session | | | | | | - Chromium (Profile A) | | - Chromium (Profile B)| | | | | | - Log/PID (Dir A) | | - Log/PID (Dir B) | | | | | +--------------------------+ +-----------------------+ | | |
+---+ |
+---+ [6]

Exact Termux Package Requirements

Implementation requires the installation of packages both on the Termux host and within the PRoot Debian guest.

Termux Host Packages:

1. x11-repo: Required for TigerVNC and X11 utilities.[6]
2. proot-distro: To manage the Debian installation.[9]
3. python: To run websockify.[25]
4. nodejs-lts: To support the Cloudy CLI environment.
5. tigervnc: Provides the VNC server functionality.[6]
6. tailscale: Provides secure remote networking.[27]
7. pulseaudio: For audio redirection to the remote client.[18]

PRoot Debian Packages:

1. xfce4, xfce4-goodies: A lightweight desktop environment.[6]
2. chromium: The primary web browser for the workspace.[9]
3. dbus-x11: Essential for session bus communication.[7]

4. pulseaudio-utils: To interact with the host's audio server.[29]

Multi-Workspace Isolation Mechanics

Isolation is enforced by defining unique parameters for each workspace, ensuring no overlapping resources.

Parameter	Workspace A	Workspace B	Purpose
Display Number	:1	:2	Graphical session separation [6]
VNC Port	5901	5902	Network separation for VNC protocol [6]
noVNC/WS Port	6080	6081	Separate browser access ports [31]
Base Directory	~/ws_a/	~/ws_b/	Filesystem isolation
Browser Profile	~/ws_a/profile	~/ws_b/profile	Independent cookies and logins [33]
Log File	~/ws_a/logs/vnc.log	~/ws_b/logs/vnc.log	Independent debugging logs [35]
PID File	~/ws_a/run/vnc.pid	~/ws_b/run/vnc.pid	Process management isolation [36]

Each workspace is launched using a dedicated xstartup script that defines the environment variables and session commands. A critical environment variable for browser isolation is the Chromium --user-data-dir flag, which ensures that each instance uses its own sandboxed profile directory.[33]

Implementation Detail: noVNC and Websockify

The noVNC client is a client-side JavaScript application that cannot communicate directly with a VNC server due to browser security restrictions on raw TCP sockets. The websockify proxy bridges this gap by translating WebSocket traffic from the browser into TCP traffic for the VNC server.[32]

noVNC Deployment in Termux

The noVNC static files are cloned into a shared directory on the Termux host. These files are served by the mini-webserver embedded within websockify.[32]

- **Path:** $HOME/gateway/novnc/
- **Source:** https://github.com/novnc/noVNC

Websockify Execution Logic

For each workspace, a separate websockify process is spawned. The security model dictates that websockify should bind to the Tailscale IP or the local loopback 127.0.0.1 to prevent unauthorized LAN access.[32]

- **Workspace A:** websockify -D --web $HOME/gateway/novnc/ 6080 localhost:5901
- **Workspace B:** websockify -D --web $HOME/gateway/novnc/ 6081 localhost:5902

By using the -D flag, websockify runs as a daemon in the background, allowing the automation script to proceed with other tasks.[20]

Security Model and Tailscale Integration

The SwissWorkspaceGateway prioritizes a minimal exposure profile. Direct exposure of VNC ports (5901/5902) to the public internet or even the LAN is discouraged due to the inherent lack of robust encryption in standard VNC protocols.[23]

Zero-Trust Access via Tailscale

Tailscale creates a secure overlay network (Tailnet) that enables peer-to-peer encrypted communication between the tablet and the remote client.[27]

1. **Userspace Networking:** On non-rooted devices, Tailscale in Termux typically operates in userspace mode. This requires the user to set up a SOCKS5 or HTTP proxy if they wish to route traffic through the Tailnet.[27] However, the simplest implementation involves the standalone Tailscale Android app, which creates a system-wide VPN interface.
2. **Binding Addresses:** All graphical services are configured to bind exclusively to 127.0.0.1. This ensures that even if a user is on a public Wi-Fi network, the workspace ports remain invisible to external scans.[6]
3. **Port Access:** The remote user connects to the noVNC port (6080 or 6081) using the tablet's private Tailscale IP (e.g., http://100.x.y.z:6080/vnc.html). Authentication is required at both the Tailscale level (SSO/MFA) and the VNC level (password).[6]

Automation and Lifecycle Management

The architecture employs the **Cloudy CLI** as the primary orchestration tool. Cloudy is a lightweight CLI utility used for spinning up and destroying cloud-like instances, but in this architecture, it is adapted to manage local Termux sessions via YAML task definitions.[40]

File and Folder Blueprint for Repository Additions

The workspace repository should be organized to facilitate idempotent deployment and clear separation of concerns.

```
/swiss-workspace-gateway/
├── automation/
│ ├── install_gateway.sh
│ ├── start_workspace.sh
│ ├── stop_workspace.sh
│ └── healthcheck.sh
├── configs/
│ ├── workspace_a/
│ │ ├── xstartup
│ │ └── vnc_config
│ └── workspace_b/
│ ├── xstartup
│ └── vnc_config
├── scripts/
│ └── cloudy_tasks.yaml
└── data/
├── ws_a/ (generated at runtime)
└── ws_b/ (generated at runtime)
```

Script Pseudocode: install_gateway.sh

This script handles the initial setup of the host and guest environments. It is designed to be idempotent, meaning it can be run multiple times without causing duplicate configurations.

Bash

~~~
#!/bin/bash
# install_gateway.sh
# 1. Update Termux repositories and packages
pkg update && pkg upgrade -y
# 2. Install host dependencies
pkg install x11-repo proot-distro tigervnc python python-pip nodejs-lts git -y
# 3. Setup PRoot Debian
if; then
    proot-distro install debian
fi
# 4. Clone noVNC
if [! -d "$HOME/gateway/novnc" ]; then
    git clone https://github.com/novnc/noVNC "$HOME/gateway/novnc"
fi
# 5. Install guest dependencies inside PRoot
proot-distro login debian -- apt update && proot-distro login debian -- apt install -y xfce4
xfce4-goodies chromium dbus-x11 pulseaudio-utils
``` [6, 9, 41, 42]

### Script Pseudocode: start_workspace.sh <id>
This script initializes the graphical session and the WebSocket proxy for a specific workspace.

```bash
#!/bin/bash
# start_workspace.sh
WS_ID=$1
DISPLAY_NUM=$(( WS_ID == "A"? 1 : 2 ))
WS_DIR="$HOME/gateway/ws_${WS_ID,,}"
mkdir -p "$WS_DIR/run" "$WS_DIR/logs" "$WS_DIR/profile"

# Start the VNC Server inside PRoot with the workspace configuration
# We use a custom Xvnc instance to ensure display separation
proot-distro login debian --shared-tmp -- env DISPLAY=:$DISPLAY_NUM \
    vncserver :$DISPLAY_NUM \
    -localhost \
    -geometry 1280x720 \
~~~

```bash
  -xstartup "$HOME/gateway/configs/workspace_${WS_ID,,}/xstartup" \
  > "$WS_DIR/logs/vnc.log" 2>&1 &

# Store PID for lifecycle management
pgrep -f "Xvnc :$DISPLAY_NUM" > "$WS_DIR/run/vnc.pid"

# Start Websockify
WEB_PORT=$(( WS_ID == "A"? 6080 : 6081 ))
VNC_PORT=$(( 5900 + DISPLAY_NUM ))
websockify -D --web "$HOME/gateway/novnc/" "$WEB_PORT" "localhost:$VNC_PORT" \
  > "$WS_DIR/logs/websockify.log" 2>&1
```

[6, 18, 31, 32]

Script Pseudocode: stop_workspace.sh <id|all>

Handles the graceful termination of sessions.

```bash
#!/bin/bash
# stop_workspace.sh
WS_ID=$1
# Kill VNC server for the specific display
if; then
  proot-distro login debian -- vncserver -kill :1
  pkill -f "websockify.*6080"
elif; then
  proot-distro login debian -- vncserver -kill :2
  pkill -f "websockify.*6081"
fi
```

[30, 36]

Script Pseudocode: healthcheck.sh

A monitoring script to verify that the workspace components are alive.

```bash
#!/bin/bash
# healthcheck.sh
# Check Workspace A
if netstat -tuln | grep -q ":5901" && netstat -tuln | grep -q ":6080"; then
  echo "Workspace A: Running"
else
  echo "Workspace A: Down"
fi
```

[43, 44]

Cloudy CLI Task Configuration (YAML)
The `cloudy_tasks.yaml` file integrates these scripts into the automation framework.

```yaml
# automation/cloudy_tasks.yaml
instances:
  - name: workspace-a
    on_create: "./automation/start_workspace.sh A"
    on_destroy: "./automation/stop_workspace.sh A"
    ssh_command: "proot-distro login debian"
  - name: workspace-b
    on_create: "./automation/start_workspace.sh B"
    on_destroy: "./automation/stop_workspace.sh B"
```

Performance Constraints and Android Pitfalls

Operating a Linux userland on top of the Android kernel presents several unique technical challenges that are not typically encountered in a standard VPS environment.

The Phantom Process Killer (Signal 9)
The most significant hurdle for Termux-based projects is the Phantom Process Killer introduced in Android 12. This mechanism monitors child processes spawned by an app and terminates them if they exceed 32 in total or consume excessive CPU in the background.[45, 46, 47]
*   **Impact:** A workspace session involving a Desktop Environment and a modern browser will easily spawn more than 32 processes, leading to an immediate crash with a "Process completed (signal 9)" message.[46, 47]
*   **Mitigation:** The process killer must be disabled via ADB. This is a one-time setup per device boot.
    *   `adb shell "/system/bin/device_config put activity_manager max_phantom_processes 2147483647"`.[45, 47]
    *   `adb shell "settings put global settings_enable_monitor_phantom_procs false"`.[47]

Battery Optimization and Wake Locks
Android's "Doze Mode" and aggressive battery optimization can suspend Termux processes when the screen is turned off or the app is not in the foreground.[48]
*   **Mitigation:** Users must acquire a "Wake Lock" within the Termux app via the notification drawer. This prevents the CPU from entering deep sleep.[48, 49] Furthermore, battery optimization must be disabled for Termux in the Android System Settings.

Hardware Acceleration (GPU)
Standard PRoot environments lack direct access to the device's GPU, leading to high CPU usage during graphical rendering.
*   **Mitigation:** The architecture can utilize `VirGL` (via `virglrenderer-android`), which provides a

bridge between the guest's OpenGL calls and the host's Adreno GPU drivers.[9, 17, 18] This is particularly useful for smooth browser scrolling and video playback.

Audio Redirection
VNC does not natively support audio streaming. To achieve audio in the remote browser, a PulseAudio bridge is required.
*   **Mechanism:** Start a PulseAudio server on the Termux host with the TCP protocol enabled: `pulseaudio --start --load="module-native-protocol-tcp auth-anonymous=1" --exit-idle-time=-1`.[29, 50]
*   **Guest Config:** Inside the PRoot environment, export `PULSE_SERVER=127.0.0.1`. Audio can then be captured by the remote client using a PulseAudio-to-WebSocket bridge or an experimental noVNC audio plugin.[50, 51, 52]

Operational Guidelines for Switzerland-Based Deployment

Given the focus on a Swiss-based tablet, considerations for network latency and data residency are paramount.

1. **Latency Management:** While Tailscale typically optimizes for the shortest path (direct P2P), users accessing the tablet from outside Switzerland may encounter increased latency. The use of a local DERP (Designated Encrypted Relay for Packets) node or ensuring that the tablet is on a high-speed fiber connection is recommended.
2. **Hardware Selection:** The Samsung Galaxy Tab S8 or S9 series is highly recommended due to the superior performance of the Snapdragon 8 Gen 1/2 processors, which handle the `ptrace` overhead of PRoot more efficiently than mid-range alternatives.[8, 53]
3. **DeX Compatibility:** When the tablet is in Samsung DeX mode, graphical performance and keyboard shortcut handling (like Alt-Tab) can sometimes conflict with the Termux X11 session. It is often more stable to run Termux in regular tablet mode for remote hosting.[54, 55]

Quick Start: MVP Implementation

To deploy the SwissWorkspaceGateway, execute the following commands in the Termux terminal. Ensure that the Phantom Process Killer has been disabled via ADB before starting.

1. `termux-setup-storage` (Grant storage permissions)
2. `pkg update && pkg install x11-repo proot-distro python nodejs-lts git -y`
3. `proot-distro install debian`
4. `pip install websockify`
5. `git clone https://github.com/novnc/noVNC $HOME/gateway/novnc`
6. `proot-distro login debian -- apt update && proot-distro login debian -- apt install -y xfce4 xfce4-goodies chromium tigervnc-standalone-server dbus-x11`
7. `proot-distro login debian -- vncpasswd` (Set your global VNC password)
8. `mkdir -p $HOME/gateway/configs/workspace_a $HOME/gateway/configs/workspace_b`
9. (Create `xstartup` files in the above directories with `xfce4-session &`)
10. `proot-distro login debian --shared-tmp -- env DISPLAY=:1 vncserver :1 -localhost -geometry 1280x720`
11. `websockify -D --web $HOME/gateway/novnc/ 6080 localhost:5901`

12. `proot-distro login debian --shared-tmp -- env DISPLAY=:2 vncserver :2 -localhost -geometry 1280x720`
13. `websockify -D --web $HOME/gateway/novnc/ 6081 localhost:5902`
14. `pkg install tailscale && tailscale up`
15. `termux-wake-lock`

The workspaces are now accessible at `http://<Tailscale_IP>:6080` (Workspace A) and `http://<Tailscale_IP>:6081` (Workspace B).

Risk and Mitigation Checklist

Risk Scenario	Mitigation Strategy
Android kills all Termux processes	Verify Phantom Process Killer is disabled via ADB; Ensure Wake Lock is active; Disable Battery Optimization.[45, 48]
Browser crashes in PRoot	Ensure the `--no-sandbox` flag is used; Check if `dbus` is running; Increase memory limit via `device_config`.[8, 56]
Unauthorized LAN access	Bind all VNC and Websockify listeners to `127.0.0.1`. Use Tailscale for all external connections.[28, 32]
Workspace crosstalk	Strictly enforce separate `--user-data-dir` paths for each Chromium instance in the `xstartup` script.
High graphical latency	Enable VirGL for hardware-accelerated rendering; Ensure direct P2P connection in Tailscale status.
Audio stutters/lag	Increase PulseAudio buffer size; switch to TCP mode with `auth-anonymous=1` on the local network.[29, 50]

Conclusion

The SwissWorkspaceGateway represents a robust and realistic implementation of a multi-session remote workspace on Android hardware. By selecting a PRoot-distro based architecture, the project gains access to a stable package ecosystem and a reliable FHS environment required for modern web browsers. The isolation strategy using distinct display numbers and browser profile directories ensures operational separation, while the security model anchored in Tailscale and localhost binding provides a zero-trust foundation. Through the systematic application of ADB-based process killer overrides and idempotent automation scripts, this architecture offers a production-ready solution that transforms a Samsung tablet into a versatile workspace server. Future development may explore the integration of nested Wayland compositors for even higher graphical performance, but for the current MVP, the Debian PRoot path provides the optimal balance of compatibility and ease of deployment.

Джерела

1. As an alternative to the VNC method suggested here, there's also the Termux-X11 ... - Hacker News, доступ отримано лютого 18, 2026, https://news.ycombinator.com/item?id=36965433
2. termux-x11 vs proot - compare differences and reviews? - LibHunt, доступ отримано лютого 18, 2026, https://www.libhunt.com/compare-termux-x11-vs-termux--proot

3. Headless Chromium, доступ отримано лютого 18, 2026, https://chromium.googlesource.com/chromium/src/+/lkgr/headless/README.md
4. Remote Desktop from Termux? - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1r031if/remote_desktop_from_termux/
5. How to connect it remotely. · Issue #681 · termux/termux-x11 - GitHub, доступ отримано лютого 18, 2026, https://github.com/termux/termux-x11/issues/681
6. Graphical Environment - Termux Wiki, доступ отримано лютого 18, 2026, https://wiki.termux.com/wiki/Graphical_Environment
7. Which Distro are you using ? : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1p7gwtn/which_distro_are_you_using/
8. Diffrence between native desktop and proot desktops : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1gmektd/diffrence_between_native_desktop_and_proot/
9. How to use Termux X11 - The X server on Android phone, доступ отримано лютого 18, 2026, https://ivonblog.com/en-us/posts/termux-x11/
10. The Best Headless Browsers of 2026 - Bright Data, доступ отримано лютого 18, 2026, https://brightdata.com/blog/web-data/best-headless-browsers
11. Isolated multiple Termux environments #11253 - GitHub, доступ отримано лютого 18, 2026, https://github.com/termux/termux-packages/discussions/11253
12. [Release] Termux Sandbox: Isolated Linux environments with native performance (The tool behind my Xfce/Minetest demos) - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1q0d0be/release_termux_sandbox_isolated_linux/
13. How to connect to Chromium Headless using Selenium - Stack Overflow, доступ отримано лютого 18, 2026, https://stackoverflow.com/questions/41836422/how-to-connect-to-chromium-headless-using-selenium
14. [Package]: Best browser in repo · termux termux-packages · Discussion #10227 - GitHub, доступ отримано лютого 18, 2026, https://github.com/termux/termux-packages/discussions/10227
15. Termux has chromium in the tur repo. - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/112jk5s/termux_has_chromium_in_the_tur_repo/
16. Questions about proot and using Chromium under x11 : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/qdo6bz/questions_about_proot_and_using_chromium_under_x11/
17. Simple script shell to set and spin up xfce4 on native Termux android (no root required). - GitHub, доступ отримано лютого 18, 2026,

https://github.com/ar37-rs/xfce4-termux
18. XFCE4 on Termux - GitHub Gist, доступ отримано лютого 18, 2026, https://gist.github.com/avoidik/3147844c90bd7c1680599d4a70e15904
19. sabamdarif/termux-desktop: Install a full Linux desktop ... - GitHub, доступ отримано лютого 18, 2026, https://github.com/sabamdarif/termux-desktop
20. Tips - X11 in an Embedded System - noVNC, доступ отримано лютого 18, 2026, https://di-marco.net/blog/it/2020-02-15-tips-x11_in_an_embedded_system_novnc/
21. Ever wanted to have a full desktop experience of some sites that were cranky on desktop mode on Chrome? Try Firefox in termux native, it's amazing. Also it's kind of private space because only few people would know the command to launch XFCE4. - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1idh3o1/ever_wanted_to_have_a_full_desktop_experience_of/
22. multiple VNC user accounts - Google Groups, доступ отримано лютого 18, 2026, https://groups.google.com/g/novnc/c/oTYG0msGPWM/m/_fnjdEewBwAJ
23. TigerVNC - ArchWiki, доступ отримано лютого 18, 2026, https://wiki.archlinux.org/title/TigerVNC
24. Setting-up Android Emulator + Appium Driver + noVNC Client on Headless Debian 12 Linux VM | by Amit Srivastava | Dec, 2025 | Medium, доступ отримано лютого 18, 2026, https://medium.com/@amitsriv99/setting-up-android-emulator-appium-driver-novnc-on-headless-debian-12-compute-engine-e5811c997310
25. Python - Termux Wiki, доступ отримано лютого 18, 2026, https://wiki.termux.com/wiki/Python
26. How can i install Pip : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1deezh0/how_can_i_install_pip/
27. Magisk/KernelSU module for running Tailscale on rooted Android devices. The easiest, most secure way to use WireGuard and 2FA. - GitHub, доступ отримано лютого 18, 2026, https://github.com/anasfanani/magisk-tailscaled
28. Install Tailscale on Linux, доступ отримано лютого 18, 2026, https://tailscale.com/docs/install/linux
29. pulse audio in chroot/proot · Issue #12289 · termux/termux-packages - GitHub, доступ отримано лютого 18, 2026, https://github.com/termux/termux-packages/issues/12289
30. Small tutorial to use connect to termux in windows using tigervnc - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/lnsuzo/small_tutorial_to_use_connect_to_termux_in/
31. How to Set Up noVNC on Ubuntu 24.04 for Remote Access via Web Browser?, доступ отримано лютого 18, 2026, https://support.stordis.com/hc/en-us/articles/26028329743517-How-to-Set-Up-noVNC-on-Ubuntu-24-04-for-Remote-Access-via-Web-Browser
32. novnc/noVNC: VNC client web application - GitHub, доступ отримано лютого 18, 2026, https://github.com/novnc/noVNC

33. User Data Directory - Chromium Docs, доступ отримано лютого 18, 2026, https://chromium.googlesource.com/chromium/src/+/HEAD/docs/user_data_dir.md
34. chromium-profile` flag is not getting respected. · Issue #1909 · mozilla/web-ext - GitHub, доступ отримано лютого 18, 2026, https://github.com/mozilla/web-ext/issues/1909
35. Tried starting display to vnc : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/1fr8xqp/tried_starting_display_to_vnc/
36. How to make sure that there is not more than one Xvnc process running? : r/termux - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/termux/comments/xpoudh/how_to_make_sure_that_there_is_not_more_than_one/
37. Why ThinLinc instead of noVNC for Linux remote computing through the browser? - Cendio, доступ отримано лютого 18, 2026, https://www.cendio.com/blog/why-thinlinc-instead-of-novnc-for-linux-remote-computing-through-the-browser/
38. Secure Remote Access VPN (Try for Free) | Tailscale, доступ отримано лютого 18, 2026, https://tailscale.com/use-cases/remote-access
39. Tailscale session with SSH on android - Reddit, доступ отримано лютого 18, 2026, https://www.reddit.com/r/Tailscale/comments/11j2l7u/tailscale_session_with_ssh_on_android/
40. cdepillabout/cloudy: CLI tool to easily spin up and control compute instances in various cloud environments - GitHub, доступ отримано лютого 18, 2026, https://github.com/cdepillabout/cloudy