Сборка кастомного Headless Arch ISO с WebUI-инсталлятором

Professional-grade руководство по созданию универсального, автономного загрузочного ISO-образа Arch Linux на базе живой системы (Live-RAM) для «слепой» установки на сервера без интернета и мониторов.

1. Введение и архитектура стенда

- **1.1. Назначение кастомного ISO:** Цели создания автономного Headless-образа, сценарии применения на изолированном оборудовании (Supermicro, HP) без графического вывода.
- **1.2. Схема и конфигурация тестовой среды:** Распределение ролей между эталонным хостом (`tom_1`), виртуальным диском флешки (`arch-flash-3`) и изолированным целевым сервером (`tom_2`).

2. Подготовка эталонного окружения (Хост tom_1)

- **2.1. Актуализация и фиксация ядра:** Полное обновление индексов пакетов и системы, перезагрузка хоста, верификация и фиксация текущей активной версии ядра через `uname -r`.
- **2.2. Установка системных пакетов:** Инсталляция базовых инструментов сжатия и окружения: `squashfs-tools`, `zram-generator`, `nginx`, `php-fpm`, `xorriso`, `samba`.
- **2.3. Конфигурация отказоустойчивой сети:** Создание универсального профиля `systemd-networkd` по маске интерфейсов (`en*`, `eth*`) для автоматического подъема статического IP `192.168.1.150`.

3. Развертывание и настройка веб-интерфейса управления

- **3.1. Системная безопасность и интеграция PHP-FPM:** Отключение контейнерной изоляции службы (`ProtectSystem=false`) и добавление беспарольных прав в `sudoers` для выполнения системных команд через `systemd-run`.
- **3.2. Конфигурация веб-сервера Nginx:** Настройка виртуального хоста бэкенда на порту `5000` с обработкой динамических сценариев через UNIX-сокет PHP-FPM.
- **3.3. Развертывание Samba для Windows-разработки:** Организация общего сетевого ресурса для прямой правки кода, запуск демонов и выставление корректных прав доступа (775 / 664) на файлы сайта для пользователя `http`.

4. Исходный код компонентов WebUI-инсталлятора

- **4.1. Структура каталогов проекта:** Организация файлов и подпапок приложения

внутри директории веб-сервера `/usr/share/nginx/html/`.
- **4.2. Фронтенд-интерфейс панели (index.html):** HTML5-разметка оконного интерфейса в стиле графической утилиты, таблиц данных и модальных форм.
- **4.3. Стили оформления панели (css/style.css):** Стилизация визуальных компонентов интерфейса, адаптивных таблиц, кнопок управления и анимации всплывающих окон.
- **4.4. Клиентские скрипты логики (js/app.js):** JavaScript-обработчик асинхронного обновления таблиц (AJAX/fetch), переключения контекста вкладок, фильтрации на лету и валидации форм.
- **4.5. Серверный обработчик пользователей (api/users.php):** PHP-скрипт парсинга файла `/etc/passwd` и обхода ограничений TTY для выполнения CRUD-операций над аккаунтами ОС.
- **4.6. Серверный обработчик групп (api/groups.php):** PHP-компонент для чтения `/etc/group` и нативного управления системными группами безопасности.

5. Консервация операционной системы в SquashFS

- **5.1. Изоляция дисковой разметки хоста:** Создание резервной копии и полное обнуление содержимого файла `/etc/fstab` во избежание конфликтов UUID и UUID-зависимостей при live-загрузке на стороннем железе.
- **5.2. Генерация монолитного SquashFS-слепка:** Сжатие корневой файловой системы хоста утилитой `mksquashfs` с использованием алгоритма ZSTD и жестким исключением виртуальных директорий ядра.
- **5.3. Немедленное восстановление хоста:** Возврат оригинального `/etc/fstab` на эталонную машину для сохранения её работоспособности и передача прав на сгенерированный файл `airootfs.sfs`.

6. Формирование структуры ISO и UEFI-загрузчика

- **6.1. Создание дерева каталогов загрузчика:** Развертывание строгой иерархии путей EFI и конфигурационных директорий внутри изолированной папки конструктора.
- **6.2. Импорт оригинальных файлов загрузки:** Перенос бинарного загрузчика `BOOTX64.EFI`, оригинального ядра Linux и соответствующего ему `initramfs-linux.img` из живой хост-системы.
- **6.3. Конфигурация загрузчика systemd-boot:** Настройка параметров ядра, привязка к глобальной метке `archisolabel=ARCH_202605`, перенаправление вывода консоли в последовательный COM-порт (ttyS0) и подавление прерываний Hyper-V.

7. Финальная сборка, перенос и тестирование образа

- **7.1. Компиляция универсального ISO:** Упаковка структуры конструктора утилитой `xorriso` с флагами гибридной разметки (Isohybrid GPT) и жестким вшиванием идентификатора тома `-volid`.
- **7.2. Экспорт ISO-образа в Windows:** Быстрое скачивание готового медиа-файла на рабочую станцию администратора средствами встроенной консоли PowerShell через `scp`.
- **7.3. Запись на физический носитель и развертывание в Hyper-V:** Пошаговая инструкция по прожигу образа в Rufus (схема GPT для UEFI) и запуску на изолированной виртуальной машине `tom_2` для верификации WebUI панели.

Сборка кастомного Headless Arch ISO с WebUI-инсталлятором

Это профессиональное руководство по созданию универсального, полностью автономного загрузочного ISO-образа Arch Linux на базе живой системы. Образ предназначен для «слепой» установки ОС на физические сервера (Supermicro, HP) без интернета и мониторов.

1. Вводные данные и Архитектура Стенда

Установка и управление производятся удаленно: доступ к консоли осуществляется по SSH, а сам запуск развертывания — через веб-интерфейс, работающий прямо из оперативной памяти (Live-ОЗУ) флешки.

Конфигурация окружения

* **tom_1** — Эталонная виртуальная машина (хост) с доступом в интернет, где подготавливается слепок системы, настраивается веб-бэкенд и собирается финальный ISO. * **arch-flash-3** — Виртуальный диск (VHDX), на который через Rufus записывается готовый ISO-образ для тестирования. * **tom_2** — Изолированная тестовая виртуальная машина без интернета, на которой имитируется «слепой» сервер. При старте с флешки она автоматически поднимает сеть на IP **192.168.1.150** и открывает порт веб-панели **5000**.

2. Этап 0. Выравнивание версий и подготовка окружения на tom_1

Чтобы предотвратить конфликт драйверов и панику модуля ZRAM на ранних секундах загрузки флешки, ядро загрузчика и модули внутри SquashFS-слепка должны совпадать символ в символ.

Шаг 2.1. Тотальное обновление и фиксация ядра хоста

Зайдите на эталонный хост `tom_1` по SSH и запустите полное обновление индекса пакетов, системных утилит и самого ядра:

```
sudo pacman -Syu --noconfirm
```

Чтобы система полностью приняла новое ядро и зафиксировала его модули в оперативной памяти, отправьте виртуальную машину в перезагрузку:

```
sudo reboot
```

Подождите 30 секунд, подключитесь заново по SSH и проверьте текущую активную версию ядра:

```
uname -r
```

Запомните этот индекс (например, `7.0.9-arch2-1`) — файлы загрузки в конструкторе ISO мы будем брать строго под эту версию.

Шаг 2.2. Установка необходимых пакетов системного окружения

Доставим на хост базовые инструменты сжатия, утилиту генерации разделов подкачки в ОЗУ, а также программную основу нашего будущего веб-инсталлятора (Nginx и PHP-FPM):

```
sudo pacman -S --noconfirm squashfs-tools zram-generator nginx php-fpm
xorriso samba
```

Активируем обработчик процессов PHP-FPM и добавим его в автозагрузку, чтобы при старте флешки сервис запустился сам:

```
sudo systemctl enable php-fpm
```

Шаг 2.3. Создание универсального сетевого конфига флешки

Чтобы один и тот же ISO-образ молча поднимал сеть на любом «зоопарке» физических серверов (где имена карт могут разъехаться на eno1, enp2s0 или eth0), заставим службу `systemd-networkd` применять настройки ко всем проводным интерфейсам по маске.

Перепишем файл конфигурации сети, зашив туда статический адрес `192.168.1.150`, который мы будем пинговать и открывать в браузере на целевом сервере:

```
cat << 'EOF' | sudo tee /etc/systemd/network/20-wired.network > /dev/null
[Match]
Name=en* eth*

[Network]
Address=192.168.1.150/24
Gateway=192.168.1.1
DNS=1.1.1.1
EOF
```

Включим автозапуск сетевых служб и резолвера имён внутри будущего слепка:

```
sudo systemctl enable systemd-networkd systemd-resolved
```

Шаг 2.4. Настройка веб-сервера и системной безопасности

Отключение системной изоляции PHP-FPM

В Arch Linux служба `php-fpm` по умолчанию запущена в изолированном контейнере (ProtectSystem=full). Из-за этого PHP видит системные файлы пользователей только для чтения. Снимем это ограничение, чтобы веб-интерфейс имел физическое право вносить изменения в ОС из ОЗУ.

Откройте переопределение настроек службы:

```
sudo systemctl edit php-fpm
```

Вставьте в открывшееся окно три строки, сохраните (CTRL+O, Enter) и выйдите (CTRL+X):

```
[Service]
ProtectSystem=false
ProtectHome=false
```

Перезапустите службу:

```
sudo systemctl restart php-fpm
```

Конфигурация Nginx под порт 5000

Перепишем конфигурацию Nginx, настроив веб-сервер на порт 5000 и обработку PHP через UNIX-сокет:

```
cat << 'EOF' | sudo tee /etc/nginx/nginx.conf > /dev/null
worker_processes 1;
events { worker_connections 1024; }
http {
    include mime.types;
    default_type application/octet-stream;
    sendfile on;
    keepalive_timeout 65;
    server {
        listen 5000;
        server_name localhost;
        root /usr/share/nginx/html;
        index index.html index.php;
        location / { try_files $uri $uri/ =404; }
        location ~ \.php$ {
            include fastcgi.conf;
            fastcgi_pass unix:/run/php-fpm/php-fpm.sock;
            fastcgi_index index.php;
        }
    }
}
EOF
```

Включите автозапуск веб-сервера:

```
sudo systemctl enable nginx
```

Настройка беспарольного доступа в sudoers

Чтобы PHP-скрипты могли вызывать утилиты управления аккаунтами без ввода пароля и без наличия текстового экрана (TTY), настроим правила безопасности. Команды будут пробрасываться во внешнюю систему через утилиту `systemd-run` для обхода ограничений безопасности PHP.

Откройте конфигурационный файл строго через visudo:

```
sudo EDITOR=nano visudo
```

В самый конец файла добавьте следующие строки:

```
http ALL=(ALL) NOPASSWD: /usr/bin/useradd, /usr/bin/userdel,
/usr/bin/usermod, /usr/bin/chpasswd, /usr/bin/groupadd, /usr/bin/groupdel
http ALL=(ALL) NOPASSWD: /usr/bin/sh, /usr/bin/systemd-run, /usr/bin/sed
Defaults:http !requiretty
```

Шаг 2.5. Настройка Samba-сервера и прав для Windows-разработки

Настройка файлового сервера Samba

Для удобной правки файлов проекта напрямую из Windows через Проводник и Notepad++, настроим файловый сервер Samba.

Создадим чистый конфигурационный файл:

```
sudo nano /etc/samba/smb.conf
```

Вставьте в него следующий рабочий конфиг, который откроет доступ к папке Nginx с автоматическим наследованием безопасных прав доступа (775 для папок и 664 для файлов):

```
[global]
    workgroup = WORKGROUP
    server string = Arch Linux Tom1
    security = user
    map to guest = Bad User
    log file = /var/log/samba/%m.log
    max log size = 50

[nginx_html]
    path = /usr/share/nginx/html
    writable = yes
```

```
    guest ok = yes
    guest only = yes
    force user = http
    create mask = 0664
    directory mask = 0775
</code >

Запустим службу  Samba  и добавим её в автозагрузку системы:
<code>
sudo systemctl enable --now smb
```

Выставление прав доступа на папки сайта

Передаем владение корневым каталогом сайта встроенному веб-пользователю `http`. Задаем права 775 для всех папок (чтобы Samba и Nginx могли создавать файлы) и 664 для файлов (только чтение и запись, без флагов исполнения):

```
sudo chown -R http:http /usr/share/nginx/html/
sudo find /usr/share/nginx/html/ -type d -exec chmod 775 {} +
sudo find /usr/share/nginx/html/ -type f -exec chmod 664 {} +
```

3. Структура каталогов и исходный код Веб-Инсталлятора

Разверните структуру каталогов внутри папки веб-сервера `/usr/share/nginx/html/`. Сделать это можно напрямую из Windows через сетевую Samba-папку.

Исходный код разделен по подпапкам по видам файлов:

```
nginx_html/
├── index.html
├── api/
│   ├── users.php
│   └── groups.php
├── css/
│   └── style.css
└── js/
    └── app.js
```

Шаг 3.1. Главный интерфейс панели (index.html)

Создайте файл `index.html` в корне сайта. Он формирует окно панели управления, вкладки переключения, таблицы и скрытые модальные формы:

```
<!DOCTYPE html>
<html lang="ru">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Control Panel</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <div class="window">
        <header class="window-header">
            <span class="title">Control Panel</span>
            <div class="window-controls">
                <button class="win-btn">?</button>
                <button class="win-btn">—</button>
                <button class="win-btn">□</button>
                <button class="win-btn close">×</button>
            </div>
        </header>
        <div class="window-body">
            <aside class="sidebar">
                <div class="search-box"><input type="text" placeholder="🔍
Search"></div>
                <nav class="menu">
                    <div class="menu-group">^ File Sharing</div>
                    <a href="#" class="menu-item">📁 Shared Folder</a>
                    <a href="#" class="menu-item">⇆ File Services</a>
                    <a href="#" class="menu-item active">👤 User & Group</a>
                    <a href="#" class="menu-item">🖧 Domain/LDAP</a>
                </nav>
            </aside>
            <main class="main-content">
                <div class="tabs">
                    <button class="tab active" id="tab-user">User</button>
                    <button class="tab" id="tab-group">Group</button>
                </div>
                <div class="toolbar">
                    <div class="actions">
                        <button class="btn primary" id="btn-
create">Create</button>
                        <button class="btn" id="btn-edit"
disabled>Edit</button>
                        <button class="btn" id="btn-delete"
disabled>Delete</button>
                    </div>
                    <div class="filter"><input type="text" id="table-filter"
placeholder="∇ Filter"></div>
                </div>
                <div class="table-container">
                    <table id="users-table">
                        <thead>
                            <tr>
                                <th>Name ▲</th>
                                <th>Email</th>
```

```
                                <th>Description</th>
                                <th>2FA Status</th>
                                <th>Status</th>
                            </tr>
                        </thead>
                        <tbody></tbody>
                    </table>
                </div>
                <footer class="table-footer">
                    <span id="items-count">0 items</span>
                    <button id="refresh-btn" class="btn">↻</button>
                </footer>
            </main>
        </div>
    </div>

  <!-- Модальное окно Пользователей -->
    <div class="modal" id="user-modal">
        <div class="modal-content">
            <h3 id="modal-title">Create User</h3>
            <form id="user-form">
                <input type="hidden" id="form-action" value="create">
                <input type="hidden" id="old-username">
                <div class="form-group">
                    <label for="username">Имя пользователя:</label>
                    <input type="text" id="username" required pattern="^[a-
z_][a-z0-9_-]*$">
                </div>
                <div class="form-group">
                    <label for="description">Описание (GECOS):</label>
                    <input type="text" id="description">
                </div>
                <div class="form-group" id="password-group">
                    <label for="password">Пароль:</label>
                    <input type="password" id="password">
                </div>
                <div class="form-buttons">
                    <button type="button" class="btn" id="btn-modal-
cancel">Cancel</button>
                    <button type="submit" class="btn primary">Save</button>
                </div>
            </form>
        </div>
    </div>

  <!-- Модальное окно Групп -->
    <div class="modal" id="group-modal">
        <div class="modal-content">
            <h3>Create Group</h3>
            <form id="group-form">
                <div class="form-group">
```

```
                    <label for="group-name">Имя группы:</label>
                    <input type="text" id="group-name" required
pattern="^[a-z_][a-z0-9_-]*$">
                </div>
                <div class="form-buttons">
                    <button type="button" class="btn" id="btn-group-
cancel">Cancel</button>
                    <button type="submit" class="btn primary">Save</button>
                </div>
            </form>
        </div>
    </div>
    <script src="js/app.js"></script>
</body>
</html>
```

Шаг 3.2. Стили оформления интерфейса (css/style.css)

Создайте файл `css/style.css` в подпапке `css/`. Код задает внешний вид окна приложения, таблиц данных, кнопок управления и всплывающих модальных окон:

```
* { box-sizing: border-box; margin: 0; padding: 0; font-family: -apple-
system, BlinkMacSystemFont, "Segoe UI", Roboto, sans-serif; }
body { background-color: #f0f2f5; display: flex; justify-content: center;
align-items: center; height: 100vh; }
.window { width: 1000px; height: 550px; background: #fff; border-radius:
6px; box-shadow: 0 5px 25px rgba(0,0,0,0.1); display: flex; flex-direction:
column; overflow: hidden; border: 1px solid #dcdcdc; }
.window-header { background: #fff; padding: 12px 15px; display: flex;
justify-content: space-between; align-items: center; border-bottom: 1px
solid #e2e8f0; }
.window-header .title { font-size: 14px; color: #2d3748; font-weight: 500; }
.window-controls .win-btn { border: none; background: none; padding: 4px
8px; cursor: pointer; color: #718096; }
.window-body { display: flex; flex: 1; overflow: hidden; }
.sidebar { width: 230px; background: #f7fafc; border-right: 1px solid
#e2e8f0; padding: 12px; }
.search-box input { width: 100%; padding: 6px 10px; border: 1px solid
#cbd5e0; border-radius: 4px; margin-bottom: 15px; }
.menu-group { font-size: 11px; text-transform: uppercase; color: #a0aec0;
margin: 12px 0 6px 6px; font-weight: 600; }
.menu-item { display: block; padding: 8px 12px; color: #4a5568; text-
decoration: none; font-size: 13px; border-radius: 4px; }
.menu-item.active { background: #ebf8ff; color: #2b6cb0; font-weight: 600; }
.main-content { flex: 1; display: flex; flex-direction: column; padding:
20px; }
.tabs { display: flex; border-bottom: 1px solid #e2e8f0; margin-top: 10px; }
.tab { padding: 10px 20px; border: none; background: none; cursor: pointer;
font-size: 14px; color: #718096; }
.tab.active { color: #3182ce; border-bottom: 2px solid #3182ce; font-weight:
```

```
600; }
.toolbar { display: flex; justify-content: space-between; margin: 15px 0; }
.btn { padding: 6px 14px; border: 1px solid #cbd5e0; background: #fff;
border-radius: 4px; cursor: pointer; font-size: 13px; color: #4a5568; }
.btn:disabled { background: #f7fafc; color: #a0aec0; cursor: not-allowed;
border-color: #e2e8f0; }
.btn:hover:not(:disabled) { background: #f7fafc; }
.btn.primary { background: #3182ce; color: #fff; border-color: #3182ce; }
.filter input { padding: 6px 10px; border: 1px solid #cbd5e0; border-radius:
4px; font-size: 13px; }
.table-container { flex: 1; overflow-y: auto; border: 1px solid #e2e8f0;
border-radius: 4px; }
table { width: 100%; border-collapse: collapse; font-size: 13px; }
th, td { padding: 10px 12px; text-align: left; border-bottom: 1px solid
#edf2f7; user-select: none; }
th { background: #f7fafc; color: #4a5568; font-weight: 600; position:
sticky; top: 0; }
tbody tr { cursor: pointer; }
tbody tr:hover { background: #f7fafc; }
tr.selected-user { background: #e8f0fe !important; }
.status-normal { color: #38a169; }
.status-deactivated { color: #e53e3e; }
.table-footer { display: flex; justify-content: flex-end; align-items:
center; padding: 12px 0; gap: 15px; font-size: 13px; color: #718096; }
.modal { display: none; position: fixed; top: 0; left: 0; width: 100%;
height: 100%; background: rgba(0,0,0,0.4); justify-content: center; align-
items: center; z-index: 1000; }
.modal.open { display: flex; }
.modal-content { background: #fff; padding: 20px; border-radius: 6px; width:
400px; box-shadow: 0 4px 15px rgba(0,0,0,0.2); }
.modal-content h3 { margin-bottom: 15px; color: #2d3748; }
.form-group { margin-bottom: 12px; }
.form-group label { display: block; font-size: 12px; color: #4a5568; margin-
bottom: 4px; }
.form-group input { width: 100%; padding: 8px; border: 1px solid #cbd5e0;
border-radius: 4px; }
.form-buttons { display: flex; justify-content: flex-end; gap: 10px; margin-
top: 18px; }
</code >
```

==== Шаг 3.3. Серверный обработчик пользователей (api/users.php) ====
Создайте файл ''api/users.php'' в подпапке ''api/''. Скрипт обрабатывает GET-запросы для вывода учетных записей (исключая технического ''nobody'') и POST-запросы для выполнения атомарных операций через ''systemd-run'' в обход изоляции:

```
<code php>
<?php
header('Content-Type: application/json; charset=utf-8');
$input = json_decode(file_get_contents('php://input'), true);

if ($_SERVER['REQUEST_METHOD'] === 'GET') {
```

```
    $usersList = [];
    if (is_readable('/etc/passwd')) {
        $lines = file('/etc/passwd', FILE_IGNORE_NEW_LINES |
FILE_SKIP_EMPTY_LINES);
        foreach ($lines as $line) {
            $parts = explode(':', $line);
            if (count($parts) >= 5) {
                $username = $parts[0]; $uid = (int)$parts[2]; $description =
$parts[4];
                if ((($uid === 0 || $uid >= 1000) && $username !== 'nobody')
|| $username === 'guest' || $username === 'admin') {
                    $usersList[] = ['name' => $username, 'email' => '',
'desc' => $description, 'tfa' => 'Disabled', 'status' => 'Normal'];
                }
            }
        }
    }
    echo json_encode($usersList, JSON_UNESCAPED_UNICODE); exit;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $action = $input['action'] ?? '';
    $username = preg_replace('/[^a-z0-9_-]/', '', $input['username'] ?? '');
    $description = $input['description'] ?? '';
    $password = $input['password'] ?? '';

    switch ($action) {
        case 'create':
            $salt = '$1$' . substr(md5(uniqid(rand(), true)), 0, 8) . '$';
            $hashed_password = crypt($password, $salt);
            $uid_gid = rand(1100, 1900);

            $passwd_line =
"{$username}:x:{$uid_gid}:{$uid_gid}:{$description}:/home/{$username}:/bin/b
ash";
            $shadow_line =
"{$username}:{$hashed_password}:19500:0:99999:7:::";
            $group_line = "{$username}:x:{$uid_gid}:";

            $system_cmd = "echo '{$passwd_line}' >> /etc/passwd && echo
'{$shadow_line}' >> /etc/shadow && echo '{$group_line}' >> /etc/group &&
mkdir -p /home/{$username} && chown -R {$uid_gid}:{$uid_gid}
/home/{$username}";
            $cmd = "sudo /usr/bin/systemd-run -G /usr/bin/bash -c " .
escapeshellarg($system_cmd) . " 2>&1";
            exec($cmd, $o, $r);
            echo json_encode(['success' => ($r === 0), 'error' => implode('
', $o)]); break;

        case 'update':
            $old_username = preg_replace('/[^a-z0-9_-]/', '',
```

```
$input['old_username'] ?? '');
            $clean_desc = str_replace('/', '\/', $description);
            $update_cmd = "sed -i -E
's/^({$old_username}:[^:]*:[^:]*:[^:]*):[^:]*(:.*)/\\1:{$clean_desc}\\2/'
/etc/passwd";

            if (!empty($password)) {
                $salt = '$1$' . substr(md5(uniqid(rand(), true)), 0, 8) .
'$';
                $hashed_password = crypt($password, $salt);
                $clean_hash = str_replace('/', '\/', $hashed_password);
                $update_cmd .= " && sed -i -E
's/^({$old_username}:)[^:]*(:.*)/\\1{$clean_hash}\\2/' /etc/shadow";
            }

            $cmd = "sudo /usr/bin/systemd-run -G /usr/bin/bash -c " .
escapeshellarg($update_cmd) . " 2>&1";
            exec($cmd, $o, $r);
            echo json_encode(['success' => ($r === 0), 'error' => implode('
', $o)]); break;

        case 'delete':
            if ($username === 'root') { echo json_encode(['success' =>
false, 'error' => 'Запрещено']); exit; }
            $delete_cmd = "sed -i '/^{$username}:/d' /etc/passwd && sed -i
'/^{$username}:/d' /etc/shadow && sed -i '/^{$username}:/d' /etc/group && rm
-rf /home/{$username}";
            $cmd = "sudo /usr/bin/systemd-run -G /usr/bin/bash -c " .
escapeshellarg($delete_cmd) . " 2>&1";
            exec($cmd, $o, $r);
            echo json_encode(['success' => ($r === 0), 'error' => implode('
', $o)]); break;
    }
    exit;
}
?>
</code >

==== Шаг 3.4. Серверный обработчик групп (api/groups.php) ====
Создайте файл ''api/groups.php'' в подпапке ''api/''. Скрипт парсит системный файл ''/etc/group'', выстраивает связи участников и нативно создаёт/удаляет группы в ОС через ''systemd-run'':

<code php>
<?php
header('Content-Type: application/json; charset=utf-8');
$input = json_decode(file_get_contents('php://input'), true);

if ($_SERVER['REQUEST_METHOD'] === 'GET') {
    $groupsList = [];
    if (is_readable('/etc/group')) {
```

```
        $lines = file('/etc/group', FILE_IGNORE_NEW_LINES |
FILE_SKIP_EMPTY_LINES);
        foreach ($lines as $line) {
            $parts = explode(':', $line);
            if (count($parts) >= 3) {
                $group_name = $parts[0]; $gid = (int)$parts[2]; $users =
$parts[3] ?? '';
                if (($gid === 0 || $gid === 998 || $gid >= 1000) &&
$group_name !== 'nobody') {
                    $groupsList[] = ['name' => $group_name, 'gid' => $gid,
'users' => empty($users) ? '—' : str_replace(',', ', ', $users), 'status' =>
'Normal'];
                }
            }
        }
    }
    echo json_encode($groupsList, JSON_UNESCAPED_UNICODE); exit;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $group_name = preg_replace('/[^a-z0-9_-]/', '', $input['group_name'] ??
'');
    if ($input['action'] === 'create') {
        $cmd = "sudo /usr/bin/systemd-run -G /usr/bin/bash -c " .
escapeshellarg("groupadd {$group_name}") . " 2>&1";
    } else {
        if ($group_name === 'root' || $group_name === 'wheel') { echo
json_encode(['success' => false, 'error' => 'Запрещено']); exit; }
        $cmd = "sudo /usr/bin/systemd-run -G /usr/bin/bash -c " .
escapeshellarg("groupdel {$group_name}") . " 2>&1";
    }
    exec($cmd, $o, $r);
    echo json_encode(['success' => ($r === 0), 'error' => implode(' ',
$o)]); exit;
}
?>
</code >

==== Шаг 3.5. Клиентские скрипты логики (js/app.js) ====
Создайте файл ''js/app.js'' в подпапке ''js/''. Скрипт управляет асинхронным обновлением таблиц (fetch), переключением контекста вкладок, фильтрацией на лету и валидацией полей ввода:

<code javascript>
document.addEventListener('DOMContentLoaded', () => {
    const tableBody = document.querySelector('#users-table tbody');
    const itemsCount = document.getElementById('items-count');
    const refreshBtn = document.getElementById('refresh-btn');
    const filterInput = document.getElementById('table-filter');

    const btnCreate = document.getElementById('btn-create');
```

```
    const btnEdit = document.getElementById('btn-edit');
    const btnDelete = document.getElementById('btn-delete');

    const userModal = document.getElementById('user-modal');
    const userForm = document.getElementById('user-form');
    const modalTitle = document.getElementById('modal-title');
    const btnModalCancel = document.getElementById('btn-modal-cancel');

    const tabUser = document.getElementById('tab-user');
    const tabGroup = document.getElementById('tab-group');
    const tableHeader = document.querySelector('#users-table thead tr');

    const groupModal = document.getElementById('group-modal');
    const groupForm = document.getElementById('group-form');
    const btnGroupCancel = document.getElementById('btn-group-cancel');

    let currentTab = 'user', selectedUsername = null, selectedGroupName =
null, selectedUserRow = null;

    async function loadUsers() {
        try {
            const r = await fetch('api/users.php'); const d = await
r.json(); renderTable(d); resetSelection();
        } catch (e) { alert('Ошибка сети'); }
    }

    function renderTable(users) {
        tableBody.innerHTML = '';
        users.forEach(user => {
            const tr = document.createElement('tr'); tr.dataset.username =
user.name; tr.dataset.desc = user.desc;
            tr.innerHTML =
`<td><b>${escapeHtml(user.name)}</b></td><td>${escapeHtml(user.email)}</td><
td>${escapeHtml(user.desc)}</td><td>${escapeHtml(user.tfa)}</td><td
class="${user.status === 'Normal' ? 'status-normal' : 'status-
deactivated'}">${escapeHtml(user.status)}</td>`;
            tr.addEventListener('click', () => {
                const active = tableBody.querySelector('.selected-user'); if
(active) active.classList.remove('selected-user');
                if (selectedUsername === user.name) { resetSelection(); }
else {
                    selectedUsername = user.name; selectedUserRow = tr;
tr.classList.add('selected-user');
                    btnEdit.disabled = false; btnDelete.disabled =
(user.name === 'root');
                }
            });
            tableBody.appendChild(tr);
        });
        itemsCount.textContent = `${users.length} items`;
    }
```

```
    async function loadGroups() {
        try { const r = await fetch('api/groups.php'); const d = await
r.json(); renderGroupsTable(d); } catch (e) { alert('Ошибка'); }
    }

    function renderGroupsTable(groups) {
        tableBody.innerHTML = '';
        groups.forEach(group => {
            const tr = document.createElement('tr');
            tr.innerHTML =
`<td><b>${escapeHtml(group.name)}</b></td><td>${group.gid}</td><td>${escapeH
tml(group.users)}</td><td class="status-normal">${group.status}</td>`;
            tr.addEventListener('click', () => {
                const active = tableBody.querySelector('.selected-user'); if
(active) active.classList.remove('selected-user');
                if (selectedGroupName === group.name) { selectedGroupName =
null; btnDelete.disabled = true; } else {
                    selectedGroupName = group.name;
tr.classList.add('selected-user');
                    btnEdit.disabled = true; btnDelete.disabled =
(group.name === 'root' || group.name === 'wheel');
                }
            });
            tableBody.appendChild(tr);
        });
        itemsCount.textContent = `${groups.length} items`;
    }

    function resetSelection() { selectedUsername = null; selectedGroupName =
null; selectedUserRow = null; btnEdit.disabled = true; btnDelete.disabled =
true; }

    tabUser.addEventListener('click', () => {
tabGroup.classList.remove('active'); tabUser.classList.add('active');
currentTab = 'user'; tableHeader.innerHTML = `<th>Name
▲</th><th>Email</th><th>Description</th><th>2FA Status</th><th>Status</th>`;
resetSelection(); loadUsers(); });
    tabGroup.addEventListener('click', () => {
tabUser.classList.remove('active'); tabGroup.classList.add('active');
currentTab = 'group'; tableHeader.innerHTML = `<th>Group Name
▲</th><th>GID</th><th>Members (Users)</th><th>Status</th>`;
resetSelection(); loadGroups(); });

    filterInput.addEventListener('input', (e) => {
        const v = e.target.value.toLowerCase();
        Array.from(tableBody.querySelectorAll('tr')).forEach(tr => {
tr.style.display = tr.textContent.toLowerCase().includes(v) ? '' : 'none';
});
    });
```

```
    btnCreate.addEventListener('click', () => {
        if (currentTab === 'user') { userForm.reset();
document.getElementById('form-action').value = 'create';
document.getElementById('username').disabled = false;
document.getElementById('password-group').style.display = 'block';
modalTitle.textContent = 'Create User'; userModal.classList.add('open'); }
        else { groupForm.reset(); groupModal.classList.add('open'); }
    });

    btnEdit.addEventListener('click', () => {
        if (currentTab !== 'user' || !selectedUsername) return;
        userForm.reset(); document.getElementById('form-action').value =
'update'; document.getElementById('old-username').value = selectedUsername;
        const uInput = document.getElementById('username'); uInput.value =
selectedUsername; uInput.disabled = true;
        document.getElementById('description').value =
selectedUserRow.dataset.desc || '';
        modalTitle.textContent = 'Edit User';
userModal.classList.add('open');
    });

    btnDelete.addEventListener('click', async () => {
        if (currentTab === 'user' && selectedUsername) { if
(confirm(`Удалить ${selectedUsername}?`)) await sendAction({ action:
'delete', username: selectedUsername }); }
        if (currentTab === 'group' && selectedGroupName) { if
(confirm(`Удалить группу ${selectedGroupName}?`)) await sendGroupAction({
action: 'delete', group_name: selectedGroupName }); }
    });

    userForm.addEventListener('submit', async (e) => {
        e.preventDefault();
        await sendAction({ action: document.getElementById('form-
action').value, username: document.getElementById('username').value,
description: document.getElementById('description').value, password:
document.getElementById('password').value, old_username:
document.getElementById('old-username').value });
        userModal.classList.remove('open');
    });

    groupForm.addEventListener('submit', async (e) => {
        e.preventDefault(); await sendGroupAction({ action: 'create',
group_name: document.getElementById('group-name').value });
groupModal.classList.remove('open');
    });

    async function sendAction(data) {
        const r = await fetch('api/users.php', { method: 'POST', headers: {
'Content-Type': 'application/json' }, body: JSON.stringify(data) });
        const res = await r.json(); if (res.success) loadUsers(); else
alert('Ошибка:' + res.error);
```

```
    }

    async function sendGroupAction(data) {
        const r = await fetch('api/groups.php', { method: 'POST', headers: {
'Content-Type': 'application/json' }, body: JSON.stringify(data) });
        const res = await r.json(); if (res.success) loadGroups(); else
alert('Ошибка:' + res.error);
    }

    btnModalCancel.addEventListener('click', () =>
userModal.classList.remove('open'));
    btnGroupCancel.addEventListener('click', () =>
groupModal.classList.remove('open'));
    refreshBtn.addEventListener('click', () => { if (currentTab === 'user')
loadUsers(); else loadGroups(); });

    function escapeHtml(t) { if (!t) return ''; return
t.toString().replace(/&/g, "&amp;").replace(/</g, "&lt;").replace(/>/g,
"&gt;"); }

    loadUsers();
});
</code >

===== 4. Подготовка слепка системы и консервация в SquashFS =====

Чтобы живой Live-образ без ошибок загружался целиком в оперативную память на любом стороннем оборудовании, мы должны временно очистить конфигурацию точек монтирования хоста перед сборкой, исключив привязку к его локальным жестким UUID.

==== Шаг 4.1. Временное обнуление таблицы разделов (fstab) ====
Сделайте резервную копию рабочей таблицы разделов вашего хоста ''tom_1'':
<code>
sudo cp /etc/fstab /etc/fstab.bak
```

Полностью очистите оригинальный конфигурационный файл /etc/fstab, чтобы инициализация systemd Live-образа не спотыкалась о чужую дисковую разметку:

```
sudo truncate -s 0 /etc/fstab
```

Убедитесь, что оригинальный файл гарантированно стал пустым:

```
cat /etc/fstab
```

Команда должна вернуть абсолютно пустую строку, подтверждая успешность операции.

Шаг 4.2. Запаковка файловой системы в SquashFS

Запустите ресурсоемкую команду создания монолитного сжатого слепка системы. Утилита

полностью проигнорирует виртуальные папки (`/proc`, `/sys`), временные ресурсы и файлы резервных копий дисков:

```
sudo mksquashfs / ~/custom_iso/arch/x86_64/airootfs.sfs -e /proc /sys /dev
/run /tmp /mnt /media /lost+found ~/archlinux-x86_64.iso ~/custom_iso -comp
zstd -b 1M
```

Дождитесь полного завершения операции, пока на экране снова не появится приглашение командной строки вашего пользователя.

Шаг 4.3. Немедленное восстановление хоста

Как только упаковщик `mksquashfs` завершит свою работу, незамедлительно верните оригинальную таблицу разделов на место, чтобы хост `tom_1` сохранил работоспособность при следующем перезапуске:

```
sudo mv /etc/fstab.bak /etc/fstab
```

Проверьте, что Btrfs-субтома и UEFI-разделы хоста вернулись на свои места:

```
cat /etc/fstab
```

Передайте права владения на созданный файл слепка текущему рабочему пользователю:

```
sudo chown eva:eva ~/custom_iso/arch/x86_64/airootfs.sfs
```

5. Формирование структуры конструктора ISO и Загрузчика UEFI

Чтобы материнские платы Supermicro, HP и ноунеймы поняли, как запускать наш кастомный образ, внутри папки конструктора должна лежать строгая структура файлов UEFI-загрузчика systemd-boot. Копирование файлов ядра и initramfs производится строго из родного каталога самого обновленного хоста.

Шаг 5.1. Создание структуры каталогов загрузчика

Создаем дерево изолированных путей для ядра и конфигурационных файлов UEFI внутри папки конструктора:

```
mkdir -p ~/custom_iso/EFI/BOOT/ ~/custom_iso/loader/entries/
~/custom_iso/arch/boot/x86_64/
```

Шаг 5.2. Копирование родных файлов загрузки и ядра

Копируем сам бинарник загрузчика, родное ядро Linux и сгенерированный под него виртуальный диск (initramfs) из живой системы `tom_1` напрямую в конструктор:

```
cp /boot/EFI/BOOT/BOOTX64.EFI ~/custom_iso/EFI/BOOT/BOOTX64.EFI
cp /boot/vmlinuz-linux ~/custom_iso/arch/boot/x86_64/vmlinuz-linux
cp /boot/initramfs-linux.img ~/custom_iso/arch/boot/x86_64/initramfs-
linux.img
```

Шаг 5.3. Перезапись конфигурации загрузчика по логике LABEL

Создаем конфигурационный файл загрузки ядра. Зашиваем туда жесткую глобальную метку тома `archisolabel=ARCH_202605`, чтобы ядро искало флешку не по уникальному UUID, а по имени.

Помня про отсутствие монитора на реальных серверах, сразу активируем вывод ядерной консоли в последовательный COM-порт и глушим ложные прерывания виртуализации Hyper-V флагом `unknown_nmi_panic=0` для стабильного прохождения тестов:

```
cat << 'EOF' > ~/custom_iso/loader/entries/01-archiso-linux.conf
title Arch Linux install medium (x86_64, UEFI)
linux /arch/boot/x86_64/vmlinuz-linux
initrd /arch/boot/x86_64/initramfs-linux.img
options archisobasedir=arch archisolabel=ARCH_202605 console=tty0
console=ttyS0,115200 unknown_nmi_panic=0
EOF
```

Параллельно создаем глобальный конфигурационный файл загрузчика:

```
cat << 'EOF' > ~/custom_iso/loader/loader.conf
timeout 3
default 01-archiso-linux.conf
EOF
```

6. Финальная сборка и развертывание кастомного ISO

Структура папок конструктора полностью готова к финальной упаковке. Осталось запустить утилиту `xorriso`, чтобы собрать папки и настроенный слепок в один готовый загрузочный файл.

Шаг 6.1. Упаковка универсального образа через xorriso

Запустите команду сборки. В ней мы жёстко привязываем идентификатор тома к нашей глобальной метке тома (параметр `-volid «ARCH_202605»`), чтобы загрузчик ядра не потерял флешку при запуске «вслепую»:

```
xorriso -as mkisofs \
```

```
  -iso-level 3 \
  -full-iso9660-filenames \
  -volid "ARCH_202605" \
  -eltorito-alt-boot \
  -e "EFI/BOOT/BOOTX64.EFI" \
  -no-emul-boot \
  -isohybrid-gpt-basdat \
  -output ~/arch_custom.iso \
  ~/custom_iso
```

Убедитесь, что утилита вывела сообщение o為 успешном завершении, и проверьте точный физический размер созданного образа (он должен быть в районе 1.6–1.8 ГБ):

```
ls -lh ~/arch_custom.iso
```

Шаг 6.2. Перенос ISO в среду Windows

Чтобы забрать готовый файл образа на рабочую Windows-машину без использования стороннего софта, откройте встроенную консоль **PowerShell** на вашем ПК и выполните команду безопасного копирования (замените IP на текущий адрес вашей ВМ `tom_1`):

```
scp eva@192.168.1.72:~/arch_custom.iso $home\Downloads\arch_custom.iso
```

Введите пароль пользователя `eva`. Образ моментально скачается в вашу системную папку «Загрузки».

Шаг 6.3. Запись образа на флешку и запуск в Hyper-V

1. Откройте утилиту **Rufus** на Windows. Выберите скачанный файл `arch_custom.iso`. 2. Выставили параметры разметки: Схема раздела — **GPT**, Целевая система — **UEFI (не-CSM)**. Метка тома автоматически встанет как **ARCH_202605**. Нажмите «Старт» для записи. 3. Отсоедините виртуальный диск флешки через окно «Управление дисками» Windows (Правый клик по плашке диска → Отсоединить виртуальный жесткий диск). 4. Зайдите в параметры изолированной ВМ `tom_2` в Диспетчере Hyper-V, добавьте жесткий диск, указав файл вашей флешки, и поднимите его на первое место в списке загрузки (Firmware). Галочка Secure Boot должна быть снята.

Запустите машину `tom_2`. Система загрузится из Live-ОЗУ флешки, поднимет статику **192.168.1.150**, и вы сможете управлять её пользователями и группами с любого устройства в локальной сети, просто открыв в браузере адрес: **http://192.168.1.150:5000**