

Rapor 2 - Geliştirme Ortamının Hazırlanması

Gelişmiş Kurulum ve Ağ Yapılandırması

Raporun Amacı: Bu belge, "Bitirme Projesi Rapor-1" belgesinin 5.1. Bölümünde tanımlanan 1. Hafta görevlerini detaylandırmaktadır. Projenin 2.2. Bölümünde tanımlanan "Uzaktan Geliştirme" akışını (CLion ile SSH bağlantısı) sağlamak amacıyla, temel sanal makine kurulumu ve bu kurulum sırasında karşılaşılan kritik ağ yapılandırma sorunları, teşhis ve çözüm adımları burada belgelenmiştir.

1. Temel Geliştirme Ortamı ve Yapılandırma

"Uzaktan Geliştirme" hedefine ulaşmak için, GNOME Boxes'ın kısıtlamalarından kaçınarak, tam kontrol sağlayan virt-manager (Sanal Makine Yöneticisi) ile sıfırdan bir geliştirme ortamı kurulmuştur.

1.1. Sanallaştırma Platformu ve Hizmet Seviyesi

- **Platform:** virt-manager (QEMU/KVM)
 - **Gerekçe:** GNOME Boxes'ın soyutladığı ve kısıtladığı gelişmiş ağ ve donanım ayarları üzerinde tam kontrol sağlamak için tercih edilmiştir. Proje raporundaki "Anlık Görüntü" (Snapshot) yeteneğini korumaktadır.
- **Hizmet Seviyesi:** qemu:///system (Sistem)
 - **Gerekçe:** Kısıtlı qemu:///session (Kullanıcı) hizmetinin aksine, bu yetkili hizmet, "Sanal ağ 'default': NAT" gibi paylaşılan sistem kaynaklarını yönetebilir. Bu, SSH bağlantısı için zorunludur.

1.2. İşletim Sistemi ve Kurulum Metodolojisi

- **İşletim Sistemi:** Debian 13 "Trixie"
- **Kurulum Metodolojisi:** Proje raporundaki "arka plan gürültüsünü en aza indirme" hedefine uygun olarak, kurulum sihirbazında "Debian desktop environment" seçeneği **işaretlenmeden**, sadece "**SSH server**" ve "**standard system utilities**" seçenekleri ile minimal bir Komut Satırı Arayüzü (CLI) kurulumu yapılmıştır.

1.3. Kaynak Tahsisi ve Gerekli Paketler

- **Kaynak Tahsisi:**
 - **CPU:** 4 Çekirdek
 - **Bellek (RAM):** 4096 MiB
 - **Disk:** 32 GiB
 - **Gerekçe:** Kaynaklar, clang/llvm ile yapılacak eBPF derlemelerinin ve CLion'ın uzaktan

indeksleme işlemlerinin ihtiyaçlarını karşılamak üzere ayarlanmıştır.

- **Gerekli Temel Paketler:**

- build-essential, gcc: Kullanıcı alanı C ajanını derlemek için.
- clang, llvm, libelf-dev, zlib1g-dev: eBPF çekirdeği alanı kodunu derlemek için zorunlu araç zinciri.
- libbpf-dev: Kullanıcı alanından eBPF programlarını yönetmek için gereken modern kütüphane.
- linux-headers-\$(uname -r): eBPF programlarının etkileşime gireceği çekirdek yapılarını tanıması için.
- bpftool: eBPF programlarını incelemek ve iskelet kod üretmek için yardımcı araç.
- cmake, gdb: CLion'ın "Uzaktan Geliştirme" özelliği ile projeleri yönetmesi ve hata ayıklaması için zorunlu paketler.

2. Ağ Sorununun Tanımlanması: Ağ İzolasyonu ve ping Başarısızlığı

Proje planına uygun olarak ana makine (Host) ile sanal makine (Guest) arasında SSH bağlantısı kurmak için yapılan ilk ping testi, %100 paket kaybı ile sonuçlanmıştır.

Analiz:

Sanal makinenin ip a komut çıktısı, 10.0.2.15 IP adresini aldığını göstermiştir. Bu IP bloğu, QEMU tarafından kullanılan varsayılan "Kullanıcı modu ağı" (User Mode Networking) tarafından atanır. Bu ağ modu, doğası gereği tam bir izolasyon sağlar: Sanal makinenin dışarıdaki internete erişmesine izin verirken, ana makinenin sanal makineye erişmesini engeller.

Bu durum, projenin "modern akış" olarak tanımladığı uzaktan geliştirme metodolojisiyle doğrudan çelişmektedir.

3. Ağ Hatası Analizi ve Kök Neden Tespiti

qemu:///system (Sistem) hizmeti üzerinden virt-manager ile kurulum yapılırken, CLion bağlantısı için zorunlu olan "**Sanal ağ 'default' : NAT**" seçeneğinin "**(Etkin değil)**" olduğu görüldü.

Bu ağı manuel olarak sudo virsh net-start default komutuyla başlatma denemesi, error: Unable to find 'dnsmasq' binary in \$PATH hatasıyla başarısız oldu.

Kök Neden: libvirt'nin default NAT ağını (DHCP/DNS) yönetebilmesi için gereken **dnsmasq** paketinin ana makinede (Host) eksik olduğu tespit edildi. Bu bağımlılık eksikliği, ağın etkinleştirilmesini engelliyordu.

4. Çözüm ve Sonuç

Belirlenen kök neden doğrultusunda Eylem Planı uygulanmıştır:

1. **Bağımlılık Giderildi:** Ana geliştirme makinesine (Host) eksik olan dnsmasq paketi kuruldu.
2. **Hizmetler Yapılandırıldı:** libvirtd servisi yeniden başlatıldı. default ağı sudo virsh net-start default ve sudo virsh net-autostart default komutları kullanılarak başarıyla başlatıldı ve kalıcı olarak etkinleştirildi.
3. **Temiz Kurulum Tamamlandı:** Sanal makine kurulumu, virt-manager ve qemu:///system (Sistem) hizmeti kullanılarak, "Sanal ağ 'default': NAT" seçeneğiyle (artık etkin durumdaydı) sıfırdan gerçekleştirildi.

Sonuç: Sanal makine, 192.168.122.XXX IP bloğundan bir adres almıştır. Ana makineden ping testi başarılıdır ve SSH bağlantısı kurulabilmektedir. Geliştirme ortamı, proje raporunda tanımlanan "Uzaktan Geliştirme" akışına (CLion bağlantısı) hazırdır.

5. Uzaktan Geliştirme Akışı (CLion) Yapılandırması ve Testi

Altyapı hazırlandıktan sonra, proje raporundaki "modern akış" hedefi CLion IDE üzerinde yapılandırılmış ve test edilmiştir.

5.1. CLion Uzak Araç Zinciri (Remote Toolchain)

1. **Bağlantı:** CLion içerisinde File -> Settings -> Toolchains menüsünden yeni bir Remote Host araç zinciri eklendi.
2. **Kimlik Bilgileri:** Sanal makinenin 192.168.122.XXX IP adresi, kullanıcı adı (developer) ve parolası girilerek SSH bağlantısı test edildi ve onaylandı.
3. **Bağımlılık Tespiti:** Bu aşamada, CLion'ın dosya senkronizasyonu için sanal makinede rsync paketine ihtiyaç duyduğu tespit edildi. sudo apt install rsync komutuyla bu bağımlılık giderildi ve araç zinciri başarıyla doğrulandı.

5.2. Proje Profili (CMake) Yapılandırması

1. **Proje Oluşturma:** merhaba-ebpf adında yeni bir C projesi oluşturuldu.
2. **Profil Yönlendirme:** Projenin Settings -> CMake menüsünde, Debug profili için "Toolchain" ayarı, varsayılan "Local" yerine Remote Host (uzak araç zinciri) olarak değiştirildi.
3. **Sonuç:** Bu değişikliğin ardından "Apply" tuşuna basıldığında, CLion proje dosyalarını rsync ile sanal makineye senkronize etmiş ve cmake işlemini SSH üzerinden uzaktan çalıştırarak projeyi başarıyla yüklemiştir.

5.3. Akışın Doğrulanması (Hostname Testi)

"Modern akış"ın tam olarak beklendiği gibi çalıştığını doğrulamak için basit bir C programı kullanıldı.

- **Test Kodu:** gethostname() fonksiyonunu (unistd.h kütüphanesinden) çağıran bir main.c dosyası hazırlandı.
- **Çalıştırma:** CLion arayüzündeki "Run" düğmesine basıldı.

- **Doğrulama:** Program, beklendiği gibi sanal makinenin hostname (ebpf-dev) bilgisini CLion'ın ana makinedeki "Run" konsoluna başarıyla yazdırdı. Bu test, kodun sanal makinede derlendiğini, sanal makinada çalıştığını ve çıktısının (stdout) SSH tüneli üzerinden IDE'ye doğru bir şekilde yönlendirildiğini kanıtlamıştır.