فاز اول پروژه

پاسخ‌دهنده: معین آعلی - ۴۰۱۱۰۵۵۶۱

۱ مقدمه‌ای بر پروژه

پروژه‌ای که قرار است توسعه داده شود، مربوط به طراحی و پیاده‌سازی یک سیستم مدیریت و اجرای محفظه‌ها (Containers) مشابه سیستم‌های شناخته‌شده‌ای همچون Docker یا Podman است. این پروژه، بدون نیاز به معماری مبتنی بر دیمون (Daemonless) اجرا می‌شود که باعث افزایش پایداری و جلوگیری از نقطه تکی شکست (Single-point-of-failure) خواهد شد.

هدف اصلی این پروژه، ایجاد و مدیریت محیط‌های منزوی برای اجرای برنامه‌هاست که از طریق تکنولوژی‌هایی همچون namespace ، cgroups ، chroot و filesystem union محقق می‌شود. همچنین از تکنولوژی eBPF برای نظارت عمیق‌تر بر تعامل برنامه‌ها با سیستم عامل لینوکس استفاده خواهد شد.

۲ اهداف و ویژگی‌های پروژه

۱.۲ ایجاد محیط‌های منزوی (Isolation)

- استفاده از namespace ها جهت تفکیک فضای PID ، UID ، GID ، Hostname و فایل سیستم از محیط میزبان.
- محدود کردن دسترسی به فایل‌ها و دایرکتوری‌ها با chroot .
- پیاده‌سازی سیستم فایل Union-filesystem (مانند overlayfs) برای جداسازی پویا و ترکیب دایرکتوری‌ها.

۲.۲ مدیریت منابع با استفاده از cgroups

- مدیریت منابع شامل حافظه، پردازنده، شبکه و I/O برای هر محفظه.
- کنترل و اعمال محدودیت‌های دقیق بر منابع اختصاص‌یافته به هر محفظه.

۳.۲ تعامل با هسته لینوکس

- استفاده از قابلیت eBPF برای نظارت بر فراخوانی‌های سیستمی و ایجاد رفتار پویا در فضای هسته.

۴.۲ ارتباط بین محفظه‌ها

- پشتیبانی از Inter-process-communication(IPC) برای ارتباط بین محفظه‌ها در شرایط خاص.
- امکان استفاده از mount های اشتراکی (propagate) جهت انتشار mount ها بین محفظه‌ها.

۵.۲ کنترل و مدیریت اجرای محفظه‌ها

- ایجاد رابط کاربری ساده و کاربردی با فرمان‌هایی نظیر ،run ،list start و status برای مدیریت محفظه‌ها.
- قابلیت ایجاد، متوقف‌سازی، راه‌اندازی مجدد و حذف محفظه‌ها.

۳ قابلیت‌های اضافی

- پیاده‌سازی قابلیت توقف (freeze) محفظه‌ها و ادامه دادن کار از نقطه متوقف شده، با استفاده از cgroups-freezer .
- ایجاد قابلیت‌هایی شبیه Docker-images برای استفاده از سیستم فایل Union-filesystem .

۴ ابزارها و زبان‌های برنامه‌نویسی پیشنهادی

۱.۴ زبان برنامه‌نویسی پیشنهادی:

- Go (Golang) : پیشنهاد اول به دلیل پشتیبانی قوی از قابلیت‌های سطح پایین لینوکس، وجود کتابخانه‌های آماده برای namespace و cgroups ، و استفاده گسترده در ابزارهای مشابه Docker و Kubernetes و Podman .
- Rust : پیشنهاد دوم برای اطمینان از ایمنی حافظه و تعامل قدرتمند با سیستم عامل.

۲.۴ ابزارهای مورد نیاز:

- Linux-Kernel (هسته لینوکس): ضروری برای استفاده از namespace و cgroups .
- eBPF-Toolchain: شامل BCC یا libbpf برای پیاده‌سازی بخش‌های نظارتی پروژه.
- OverlayFS : جهت پیاده‌سازی Union-Filesystem .
- systemd و ابزارهای مرتبط: برای مدیریت پردازش‌ها (اختیاری اما کاربردی در محیط‌های لینوکسی).

۵ پیش‌نیازهای فنی پروژه

برای اجرای موفق این پروژه نیاز به پیش‌نیازهای زیر دارید:

- دانش قوی از ساختار سیستم عامل لینوکس به خصوص namespace ، cgroups و chroot .
- درک کافی از مدیریت منابع سیستم و تعامل با هسته لینوکس.
- آشنایی با اصول طراحی سیستم‌های Daemonless .
- تسلط نسبی به eBPF برای ایجاد قابلیت نظارتی قدرتمند.
- آشنایی با مفاهیم پایه‌ای شبکه و IPC .

۶ توضیحات اهداف پروژه

۱.۶ آشنایی با Namespace ها :

Namespace در لینوکس ابزاری است که منابع مختلف سیستم (مثل PID فرایندها، شبکه، کاربران، فایل سیستم و...) را از دید برنامه‌ها جداسازی کرده و به هر برنامه محیطی مجزا و منزوی ارائه می‌دهد.

۲.۶ مدیریت منابع با cgroups :

Control-Groups یا cgroups به شما امکان می‌دهد میزان استفاده از منابع سیستم را برای هر محفظه یا برنامه محدود کرده و دقیقاً کنترل کنید تا مصرف بی‌رویه منابع توسط برنامه‌ها رخ ندهد.

۳.۶ منزوی‌سازی محیط‌ها با chroot :

Chroot ابزاری است که به کمک آن دسترسی یک برنامه به سیستم فایل اصلی محدود می‌شود؛ درواقع برنامه در محیطی محدود اجرا می‌شود و نمی‌تواند به فایل‌ها و دایرکتوری‌های خارج از آن دسترسی پیدا کند.

۴.۶ ساخت سیستم‌های فایل Union :

Union-Filesystem (مانند overlayfs) روشی برای ترکیب چند دایرکتوری به صورت پویا است که به ایجاد فایل‌سیستم‌های جداگانه برای هر محفظه کمک می‌کند؛ به این ترتیب هر محفظه یک دید متفاوت و مجزا به فایل‌ها دارد.

۵.۶ ایجاد یک سیستم مدیریت محفظه:

هدف این است که سیستمی مشابه ابزارهایی مانند Podman طراحی شود که وظیفه ایجاد، مدیریت، اجرا و نظارت بر محفظه‌ها را دارد و یک رابط ساده برای کنترل آن‌ها ارائه می‌دهد.

۶.۶ تعامل با سیستم‌عامل لینوکس:

در این هدف لازم است مستقیماً با هسته لینوکس ارتباط برقرار کرده و با استفاده از فراخوانی‌های سیستمی و کدنویسی سطح پایین، قابلیت‌های عمیق‌تری در سیستم ایجاد شود.

۷.۶ آشنایی با eBPF :

eBPF تکنولوژی قدرتمندی است که اجازه می‌دهد رفتار دلخواه و پویایی در هسته سیستم‌عامل ایجاد کرده و به شکل مؤثر و ایمن فعالیت‌های سیستم عامل و برنامه‌ها را مانیتور یا کنترل کنید.