Von PXELinux zu iPXE im Rechenzentrum

Wie man für >400.000 Systeme eine Firmware wechselt

sinuscosinustan, CLT 2025

HETZNER

Für was braucht Hetzner Netzwerkboot?

HETZNER

Netzwerkboot Use Case

- Schnelle Bereitstellung von Servern in wenigen Minuten möglich
- Autoprovisionierung ohne menschliche Interaktion
- Rescue-System

HETZNER

Was ist iPXE?

- Open Source network boot firmware (GNU GPL licensed)
- Entwickelt durch Fen Systems
- Scriptable Firmware mit Bootmöglichkeiten für iSCSI, InfiniBand, Fibre Channel (over Ethernet) und mehr

HETZNER

Vorteile iPXE vs PXELinux

- Kein `pxelinux.cfg`
- Dynamische Configs dank eigener Scriptsprache
- SAN Boot möglich
- Windows Installationen mit `wimboot` einfach möglich
- Gleiche Config kann für verschiedene Deployments verwendet werden
  - UEFI, Legacy, KVM (iPXE ist das default Option ROM bei QEMU fuer SeaBIOS)

HETZNER

Vorteile iPXE vs PXELinux

- Kann über PXE geladen oder als Option ROM auf der NIC geflashed werden
- TFTP ist meh - Latenzabhängig und langsam bei grossen Dateien wie WinPE oder initramfs
  -> HTTP is the way

HETZNER

Beispielconfig kea-dhcp

/etc/kea/kea-dhcp4.conf

```
{
  "Dhcp4": {
    "client-classes": [
      {
        "name": "UEFI x86_64",
        "test": "not(substring(option[77].hex,0,4) == 'iPXE') and (substring(option[93].hex,0,2) == 0x0007)",
        "next-server": "10.255.255.1",
        "boot-file-name": "snponly-amd64.efi"
      },
      {
        "name": "UEFI arm64",
        "test": "not(substring(option[77].hex,0,4) == 'iPXE') and (substring(option[93].hex,0,2) == 0x000b)",
        "next-server": "10.255.255.1",
        "boot-file-name": "snponly-arm64.efi"
      },
      {
        "name": "iPXE",
        "test": "(substring(option[77].hex,0,4) == 'iPXE')",
        "next-server": "10.255.255.1",
        "boot-file-name": "http://10.255.255.1/ipxe/index.ipxe"
      },
      {
        "name": "PXE",
        "test": "not(substring(option[77].hex,0,4) == 'iPXE') and not(substring(option[93].hex,0,2) == 0x0007)",
        "next-server": "10.255.255.1",
        "boot-file-name": "undioly.kpxe"
      }
    ]
  }
}
```

HETZNER

Unterschied zwischen ipxe.efi, snp.efi und snponly.efi

ipxe.efi	snp.efi	snponly.efi
Alle aktivierten Treiber einkompiliert	Ausschließlich SNP und NII Treiber vorhanden (ähnlich wie das Legacy undionly.kpxe Binary)	
“Großes” Binary (~1MiB)	Kleines Binary (~ 200 KiB)	
Alle verfügbaren Interfaces können zum Boot verwendet werden		Netboot ausschließlich vom gestarteten Interface möglich
Nicht alle Treiber implementiert - Keine Garantie für arm64	Solange SNP / NII vorhanden ist (was meistens der Fall ist) - Funktioniert in der Regel für arm64	

HETZNER

Unterschied zwischen ipxe.pxe, undionly.kpxe und undionly.kkpxe

ipxe.pxe	undionly.kpxe	undionly.kkpxe
Entlädt PXE und UNDI beim Init	Entlädt nur PXE Stack beim Init	Behält PXE und UNDI im RAM
Binary ~ 300 KiB	Binary ~ 100 KiB	
Alle verfügbaren Interfaces können zum Boot verwendet werden	Netboot ausschließlich vom gestarteten Interface möglich	
Empfohlen für Produktion		Nicht empfohlen - nur bei buggy Systemen

HETZNER

Configvergleich iPXE - PXELinux

iPXE

```
#!ipxe

menu Hetzner Online GmbH :: PXE Boot-System
item --key l exit         Boot from first hard disk (default)
item --key 1 rescueamd64  (1) Rescue-System amd64 (bookworm)
item --key m memtest      (m) Memtest86+
choose --timeout 10 --default exit selection || goto exit
goto ${selection} # default selection is exit

:rescueamd64
kernel http://10.255.255.1/rescue/vmlinuz \
       initrd=initrd.img BOOTIF=01-${netX/mac:hexhyp}
initrd --name initrd.img \
       http://10.255.255.1/rescue/initramfs.cpio.zst
boot

:memtest
kernel http://10.255.255.1/tools/memtest86
boot

:exit
iseq ${platform} pcbios && sanboot --drive 0x80 || exit 1
```

Command - Label - Argument - Comment

PXELinux

```
TIMEOUT 100

MENU TITLE Hetzner Online GmbH - PXE Boot-System
MENU ROWS 10
MENU WIDTH 80
MENU MARGIN 5
MENU CMDLINEROW 23
MENU ENDROW 25

LABEL local1
  MENU LABEL ^L. Boot from first hard disk  (TAB to edit)
  MENU DEFAULT
  LOCALBOOT 0

LABEL rescueamd64
  MENU LABEL ^1.  -> Rescue-System amd64 (bookworm)
  KERNEL  rescue/vmlinuz
  APPEND  initrd=rescue/initramfs.cpio.zst
  IPAPPEND 2

LABEL memtest
  MENU LABEL ^6.  Memory Test 86+
  KERNEL tools/memtest86+p
```

HETZNER

Lab war gut, also Staged Rollout?

HETZNER

Erste Stufe vom Staged Rollout

- Einige Systemkonfigurationen sind nicht mehr ins OS gebootet
- Bootloops wegen Crash durch Memory Corruption
- Workarounds für Buggy Systeme griffen nicht zuverlässig
- Testweise auf neusten Git Commit, aber war nicht besser
- Abbruch des Rollouts und alle zurück zu PXELinux

HETZNER

Problemuntersuchung

Warum geht es im Lab, aber nicht bei allen Kunden?

- Mischen von NVMe und SATA problematisch
- Bootloader nicht auf allen Disks installiert -> sollte die nächste Disk nutzen, hat es aber nicht
- FreeBSD startet nicht mehr
  - Grund: Legacy IRQ 8 (RTC) wurde ohne Prüfung deaktiviert
    - Wurde letzte Woche in master durch Report von Hetzner gefixed 🥳 (Thanks mcb30 for fixing the issue!)

HETZNER

Problemuntersuchung

Warum geht es im Lab, aber nicht bei allen Kunden?

- Nicht-Standard Configs von Kundensystemen nachgebaut, teilweise reproduzierbare Bugs
  - Mit `undionly.kkpxe` lief es
    - Aber: Boot von NVMe mit `sanboot --drive 0x80` unzuverlässig `exit` zuverlässiger

HETZNER

Staged Rollout die zweite

HETZNER

Staged Rollout die zweite

Nicht so schlimm wie man denkt

- iPXE kann kein DHCP machen und verhindert Diskboot
  - Betrifft primär Intel e1000e-basierte PCIe Steckkarten mit Firmware von 2008
    - NIC tauschen -> solved
- Dell AMD EPYC Systeme frieren bei `iPXE Initialising devices…` mit PCIe Steckkarte ein
  - Passiert nur wenn **keine** OCP NIC installiert ist - mit OCP NIC klappt es
  - BIOS Bug - Problem wird aktuell noch mit Dell untersucht

HETZNER

Staged Rollout die zweite

Nicht so schlimm wie man denkt

- Custom Mainboard Serie friert nach `exit` ein
  - Force Flash vom BIOS behebt das Problem - wird aktuell untersucht
- Intel Mainboard kommt mit Intel OpROM nicht klar wenn `kkpxe` verwendet wird (lol)
  - Workaround: Assembly `INT 18h` statt `exit` wenn Board erkannt wird

HETZNER

Staged Rollout die zweite

Nicht so schlimm wie man denkt

- DHCP Option 82 kommt nicht am DHCP Server an
  - Zu große `DHCPDISCOVER` Pakete durch iPXE's eigenes Option-Set
  - Festgestellt mit MikroTik RouterOS Switch
  - MikroTik RouterOS Bug - kann auch den Switch crashen
    - Bugreport offen, aber bis jetzt nicht behoben (ROS 7.18.2)
  - Workarounded indem unnötige Optionen in iPXE entfernt werden

HETZNER

What we learned

Do and don’ts

- Architekturen (x86, x86_64, arm64) so früh wie möglich im Script prüfen
- “Unnötige” Features wie VLAN Support nicht mit `undionly`
  - Grund: UNDI Stack in NIC Firmware oft buggy oder unvollständig
- Keine Hintergrundbilder! - GFX Framebuffer unter Legacy / CSM kann System crashen
- `BOOTIF` von PXELinux kann mittels `01-${netX/mac:hexhyp}` wiederhergestellt werden

HETZNER

What we learned
Do and don’ts

- `exit 1` statt `exit` für UEFI
  - Grund: Einige Hersteller haben die UEFI Spec richtig implementiert
- HTTPS vermeiden
  - CA Trust durch ROM Limitierung sehr “kreativ” gelöst
  - Cross-Signing der Root CAs notwendig wenn keine externe Abhängigkeit gewünscht -> github.com/hetznercloud/ansible-role-ipxe-ca
  - Nur ältere / unsichere Ciphers unterstützt
  - Standardmäßig externe Dependency auf iPXE Infrastruktur -> No-Go für uns
  - Wenn der RNG im BIOS falsch implementiert ist -> System Crash

HETZNER

What we learned

Hardware is hard, firmware even harder

- System Freezes wegen fehlerhafter IRQ Implementierung in der NIC Firmware oder BIOS
- Bestimmte BIOS Settings können OS Boot verhindern (z.B. SR-IOV)
- Bei Problemen -> Erst neuster iPXE Commit, NIC Firmware und BIOS updaten, dann Bug Report
- Base Commit pinnen und nur nötige Commits selbst backporten

HETZNER

Q&A

HETZNER