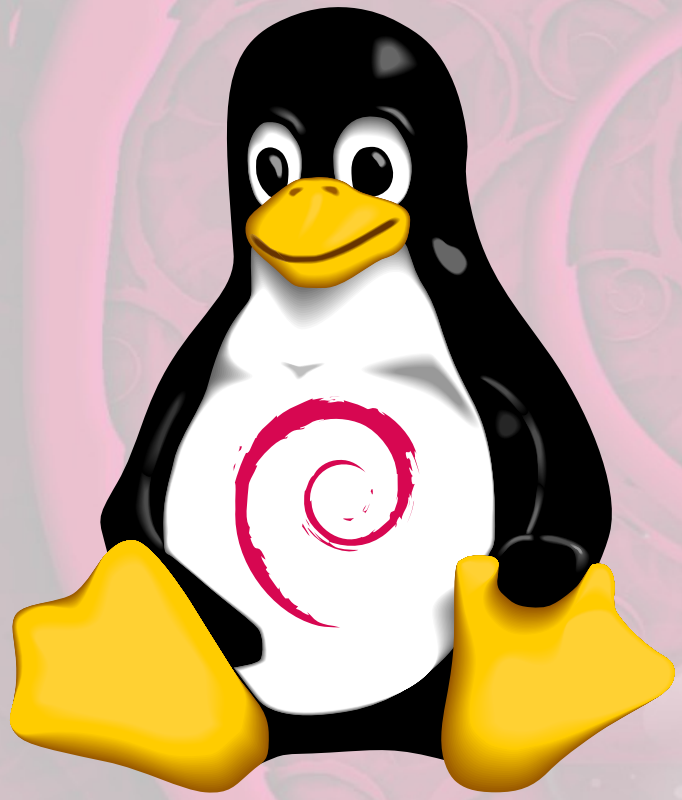
Secure Boot vs the Debian linux package



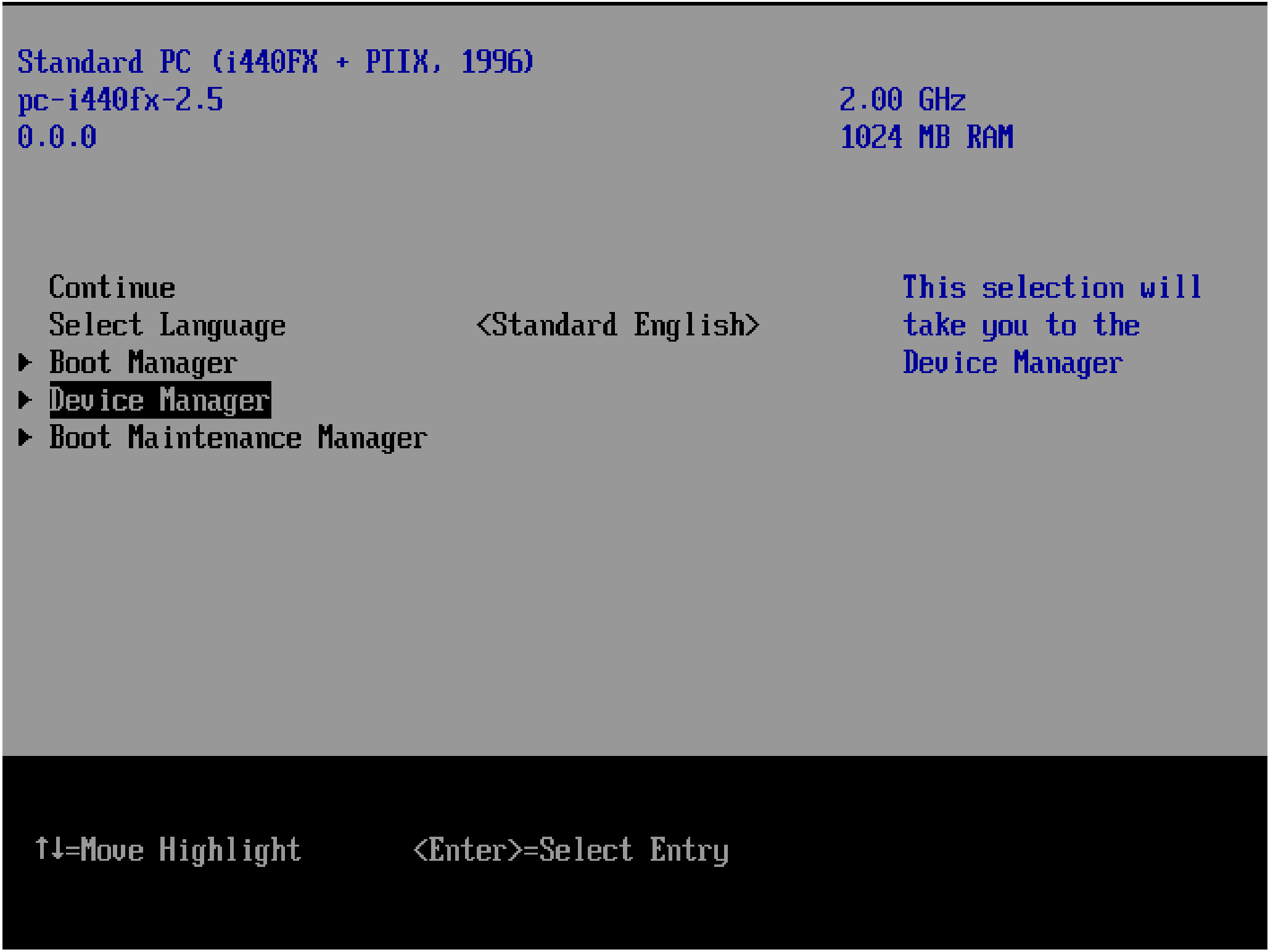
Ben Hutchings

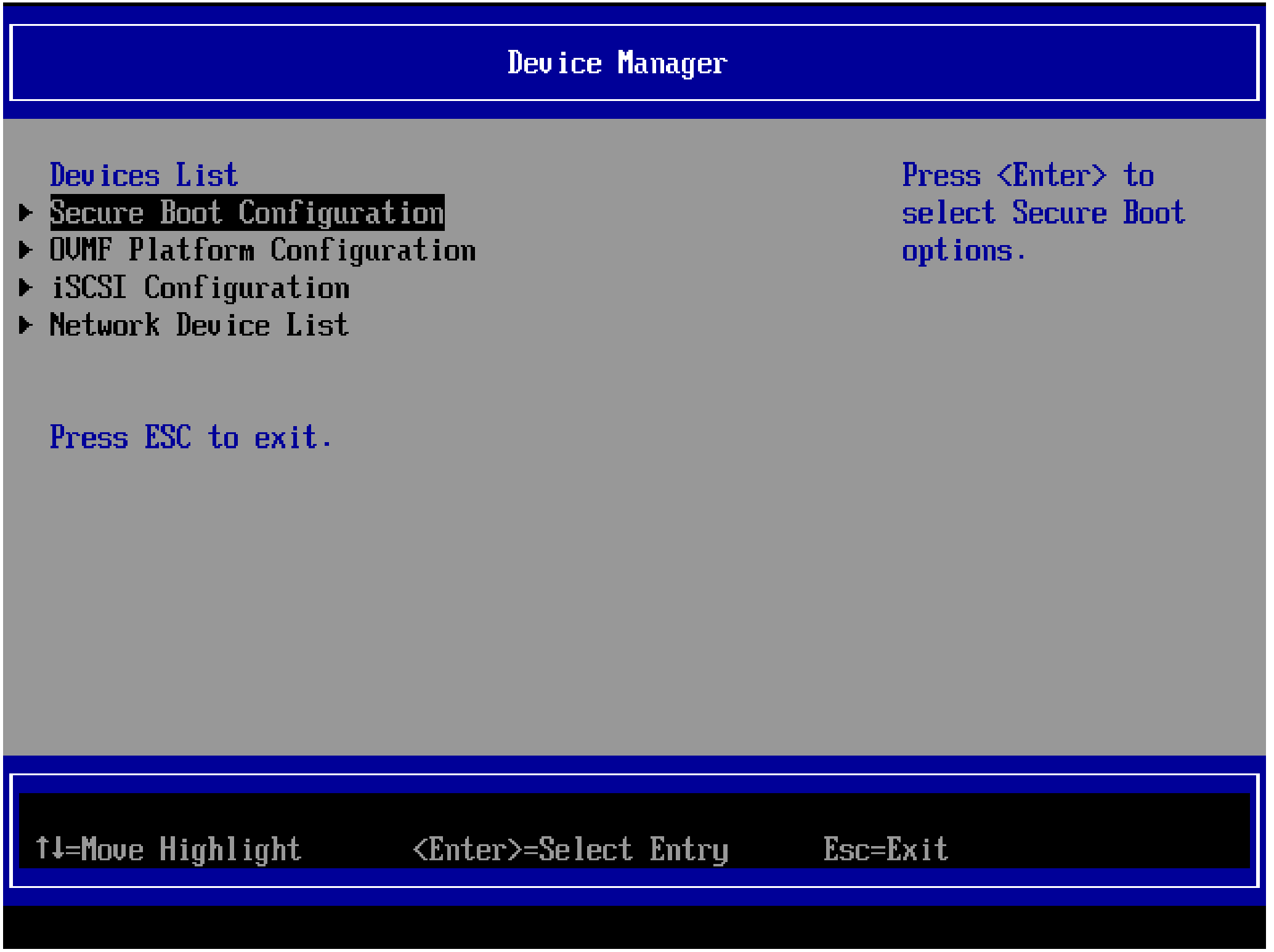
Mini-DebConf Vienna, 2016

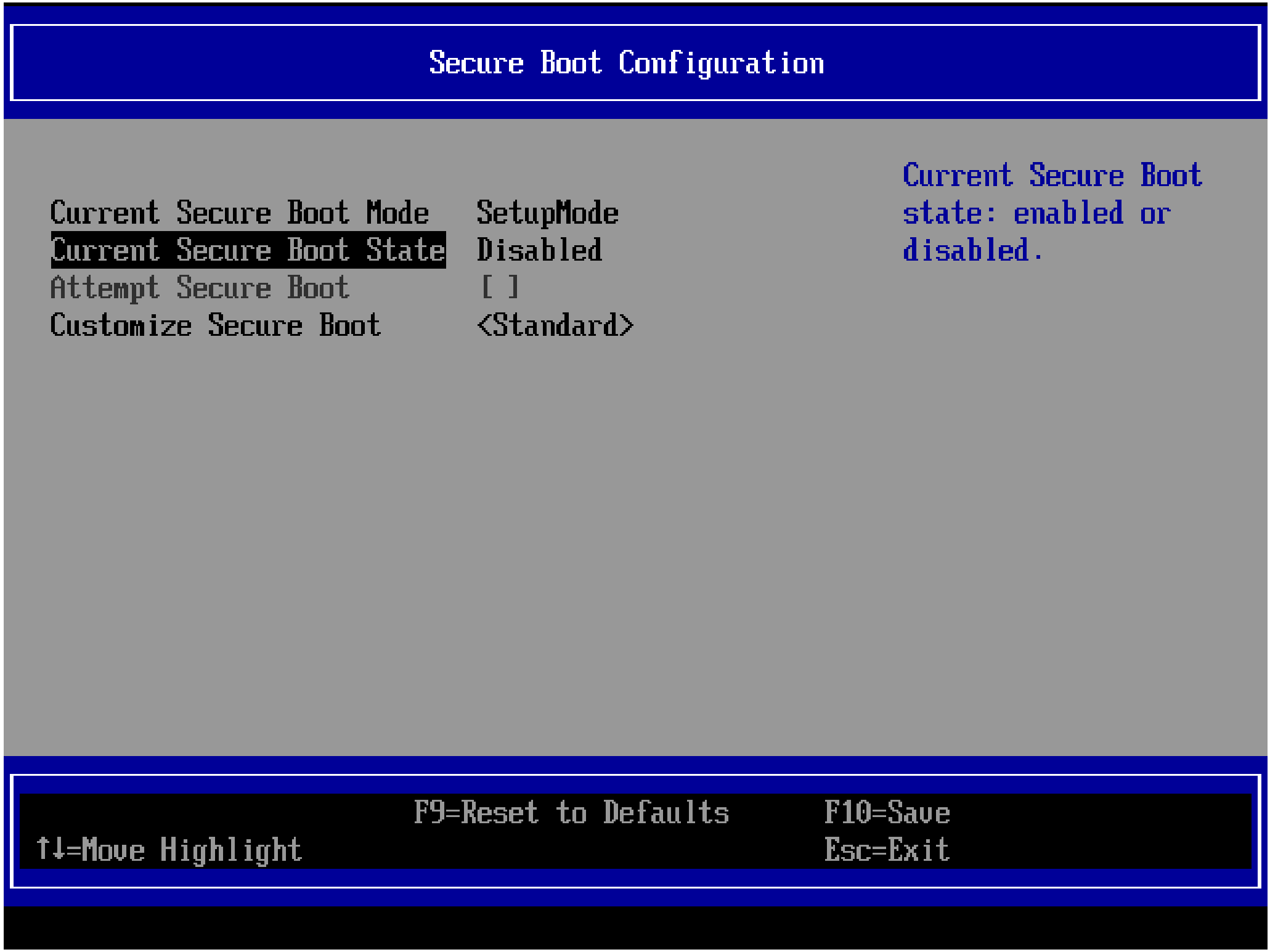
Ben Hutchings

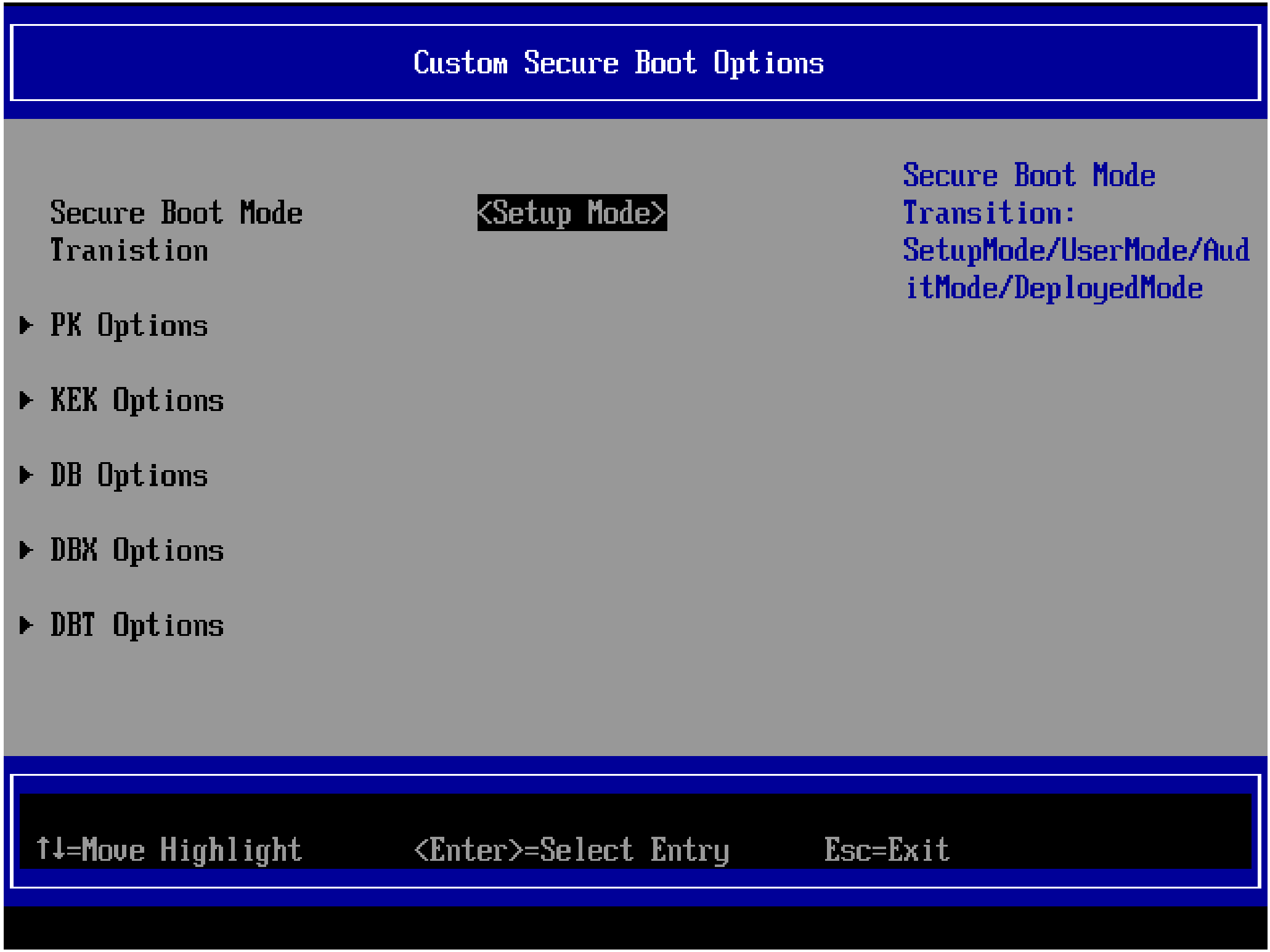
- Professional software engineer by day, Debian developer by night (or sometimes the other way round)
- Regular Linux contributor in both roles since 2008
- Working on various drivers and kernel code in my day job
- Debian kernel and LTS team member, now doing most of the kernel maintenance aside from ports
- Maintaining Linux 3.2.y and 3.16.y stable update series on kernel.org


ACCESS
DENIED


Standard PC (i440FX + PIIX, 1996)
pc-i440fx-2.5
0.0.0
2.00 GHz
1024 MB RAM
Continue
Select Language
<Standard English>
This selection will take you to the Device Manager
Boot Manager
Device Manager
Boot Maintenance Manager
↑↓=Move Highlight
<Enter>=Select Entry


Device Manager
Devices List
Secure Boot Configuration
OVMF Platform Configuration
iSCSI Configuration
Network Device List
Press ESC to exit.
Press <Enter> to select Secure Boot options.
↑↓=Move Highlight
<Enter>=Select Entry
Esc=Exit


Secure Boot Configuration
Current Secure Boot Mode SetupMode
Current Secure Boot State Disabled
Attempt Secure Boot [ ]
Customize Secure Boot <Standard>
Current Secure Boot state: enabled or disabled.
F9=Reset to Defaults
F10=Save
↑↓=Move Highlight
Esc=Exit


Custom Secure Boot Options
Secure Boot Mode Tranistion
<Setup Mode>
Secure Boot Mode Transition: SetupMode/UserMode/AuditMode/DeployedMode
PK Options
KEK Options
DB Options
DBX Options
DBT Options
↑↓=Move Highlight
<Enter>=Select Entry
Esc=Exit

Secure Boot

- Optional feature in UEFI - uses certificate store to validate boot loader, UEFI drivers, system firmware updates
- Protects against persistent malware (bootkit / kernel rootkit) if implemented correctly
- Required in 'Designed for Windows' systems since Windows 8 (2012)
- Only common trusted certificates on PCs are for Microsoft signing keys
  - MS *will* sign PC boot loaders for a small fee, and the certificate store is normally editable on PCs
  - ARM-based Windows systems are completely locked down

GNU/Linux under Secure Boot

- First stage needs MS signature – manual submission process
  - Most distributions introduced 'shim' as first stage boot loader that won't need updating often
- MS expects boot loader and kernel to validate code they load – and it's a good idea anyway
  - For later stages, we control certificates and keys – certificates can be embedded in 'shim'
  - GRUB needs to validate its modules and kernels
  - Linux kernel needs to validate its modules and any other code that runs in kernel mode

Securing the Linux kernel

- Root user, including malware running as root, can modify kernel without using any security bugs
- MS signing requires kernel to validate code it runs – and it's also a good idea in general
- Using Matthew Garrett's patchset to add 'securelevel' feature, activated when booted under SB:
  - Module signatures are mandatory
  - Kexec is disabled – but images *could* be signed and validated
  - Hibernation is disabled – but images *could* be signed and validated using per-machine key
  - Other kernel APIs that allow peek/poke are disabled
  - See Documentation/security/securelevel.txt

The signature problem

- We don't want to expose signing keys to buildds
- Reproducible builds can't depend on anything secret
- So we can't *auto-build* signed binaries in single step

Solution requires an extra source package:

1. Build unsigned binaries from first source package
2. Sign 'offline' and put detached signatures in second source package
3. Build signed binaries from second source package

Introducing linux-signed

- The second source package for linux
- Contains signing script and detached signatures for specific kernel version
- Builds binary packages linux-image-*version*-*flavour*-signed containing the detached signatures
- On installation, binary package attaches signature to make signed kernel image (on architectures where UEFI boot is supported)
- Module signatures are not currently attached on disk – problem to be solved

Module signatures

- Modules loaded and passed to kernel by kmod/libkmod which always looks in /lib/modules/*version*/...
- Must not delete unsigned modules, so how do we make kmod load signed modules instead of them?
- Current implementation: make kmod look for detached signatures and attach them in memory
  - Also requires changes to initramfs-tools, dracut, kernel-wedge, etc. to include signatures in initramfs and installer
- Alternative: install in subdirectory and change module search path
- Alternative: divert unsigned modules and replace with signed (does dpkg-divert scale to thousands of files?)

We're not ready for SB yet

- Bug #820036 tracks work to be done
  - Archive infrastructure for signed packages
  - shim package
  - GRUB signed build and validation of next stage
  - Signed binaries in installer and live images
  - Support for detached module signatures in kmod (or alternative solution)
- Some info at https://wiki.debian.org/SecureBoot
- BoF at DebConf 16
- Ready for stretch freeze (Jan 2017)?