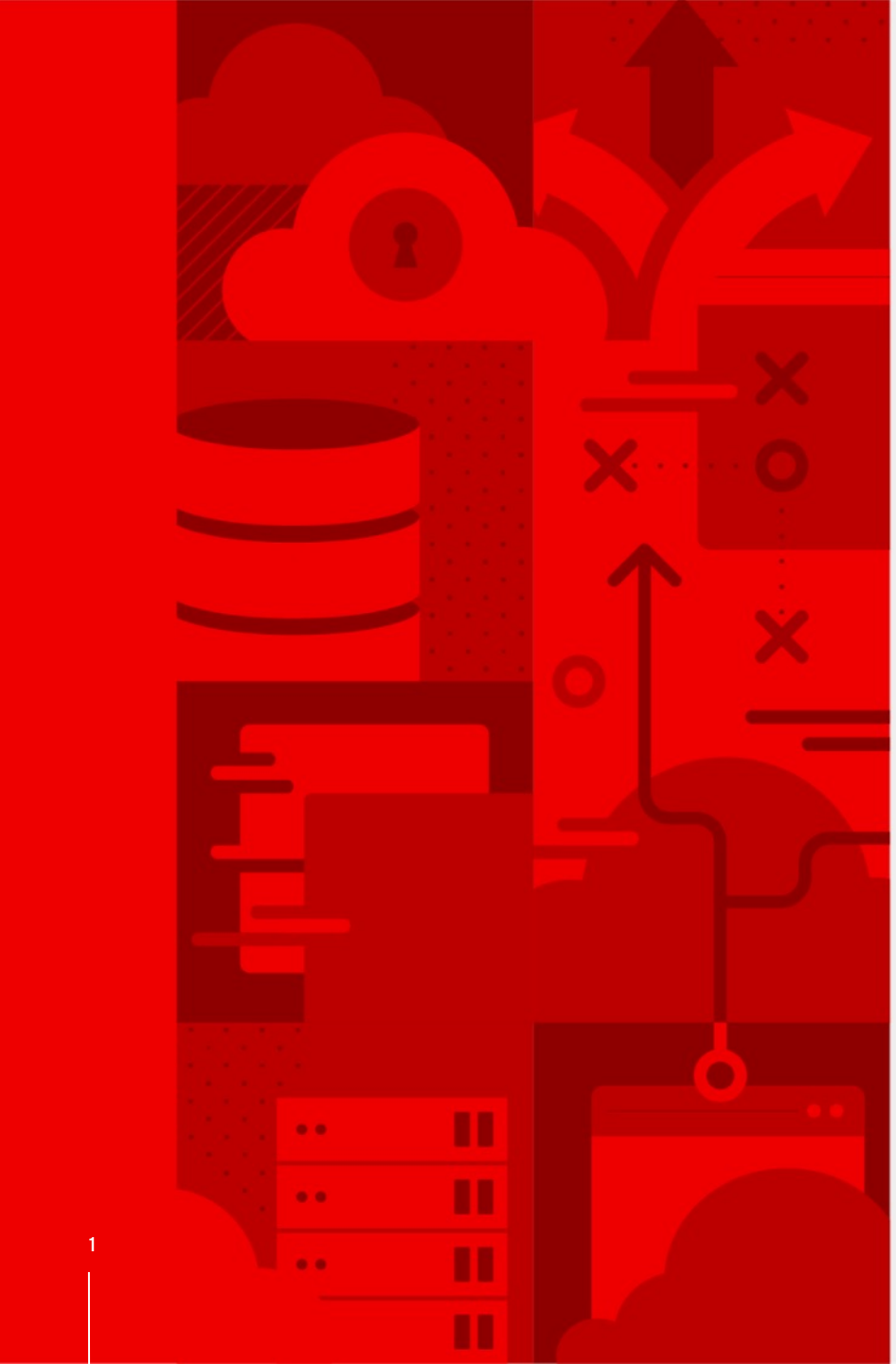


Firmware SBoM

Let’s add a Software Bill of Materials to firmware images.

Richard Hughes, Senior Principal Engineer, Red Hat
Martin Fernandez, Software Engineer, Eclypsium

Red Hat

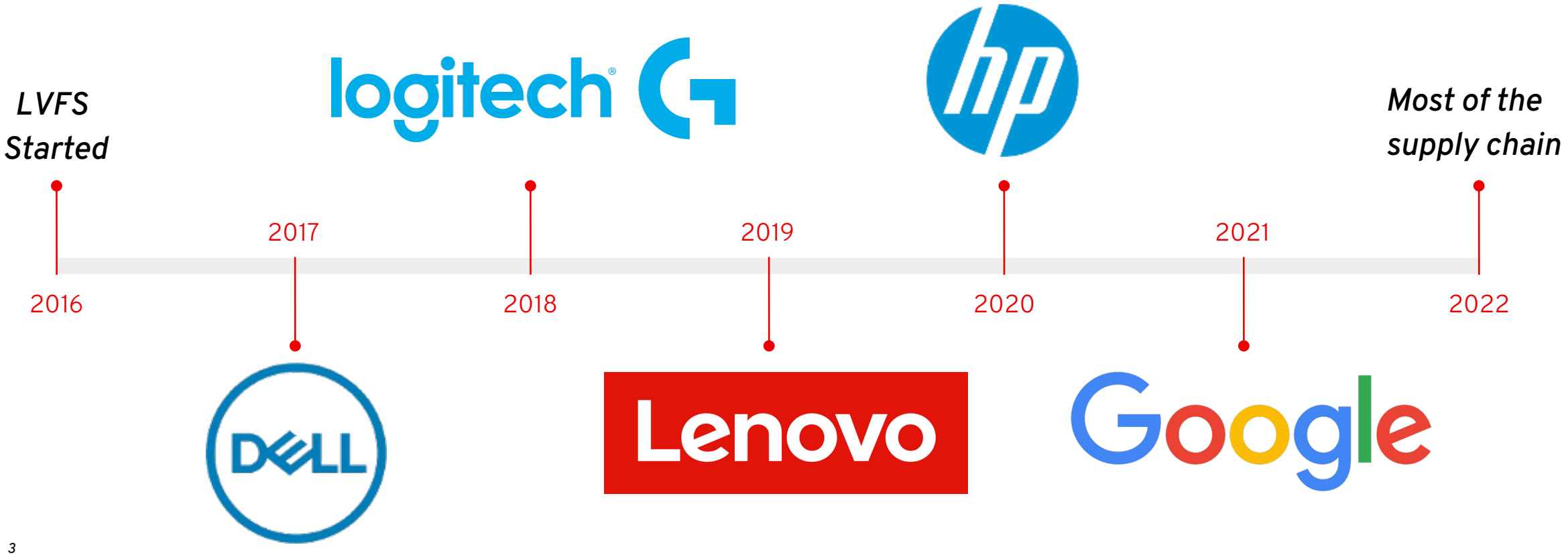
Who are we?

I've been building Open Source for **over 20 years**, 15 of which at Red Hat.

I built fwupd and the Linux Vendor Firmware Service.

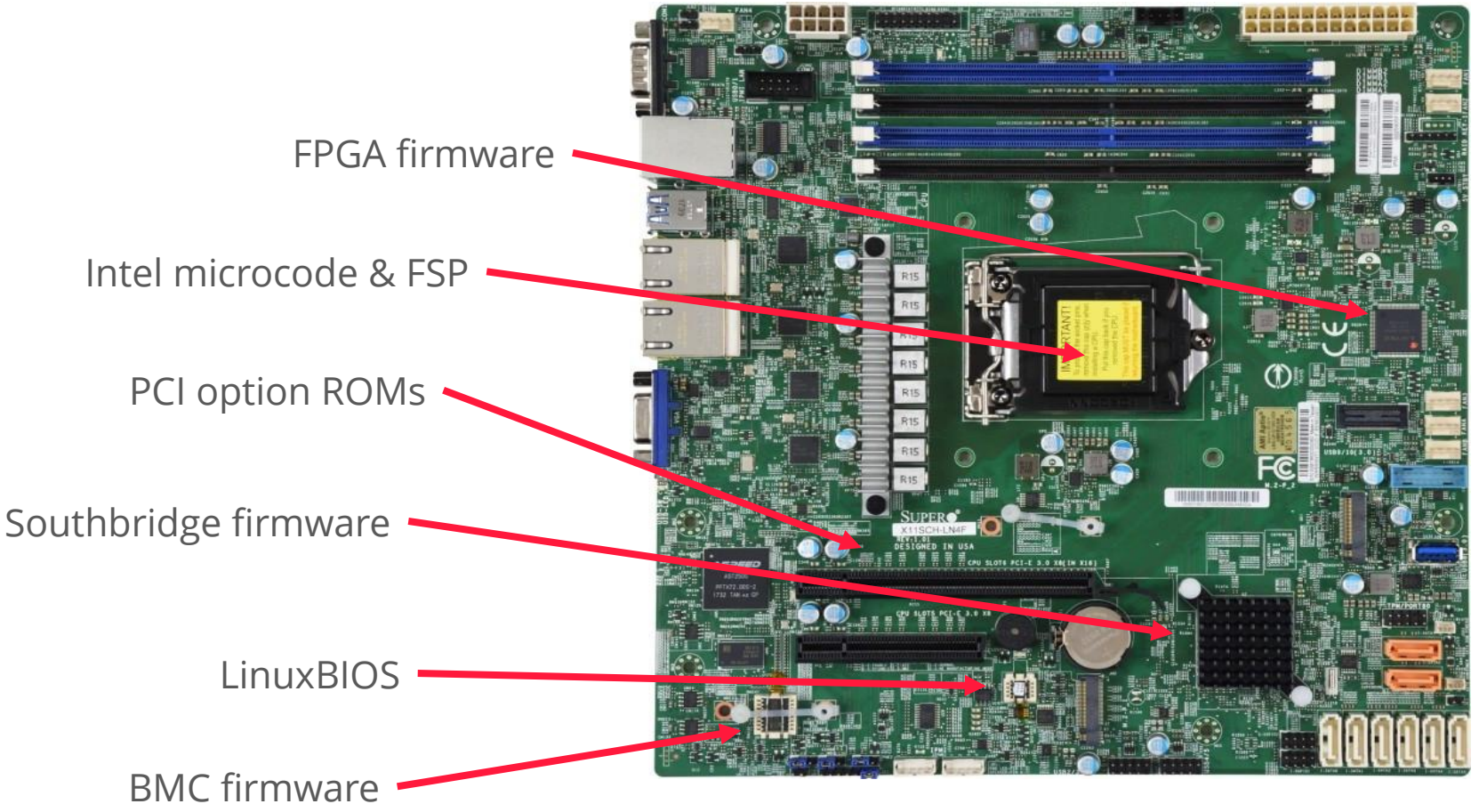
Martin has been working on SBoM at Eclypsium for ~2 years.

Over 140 OEMs, ODMs, IBVs and IHVs use the LVFS



Red Hat

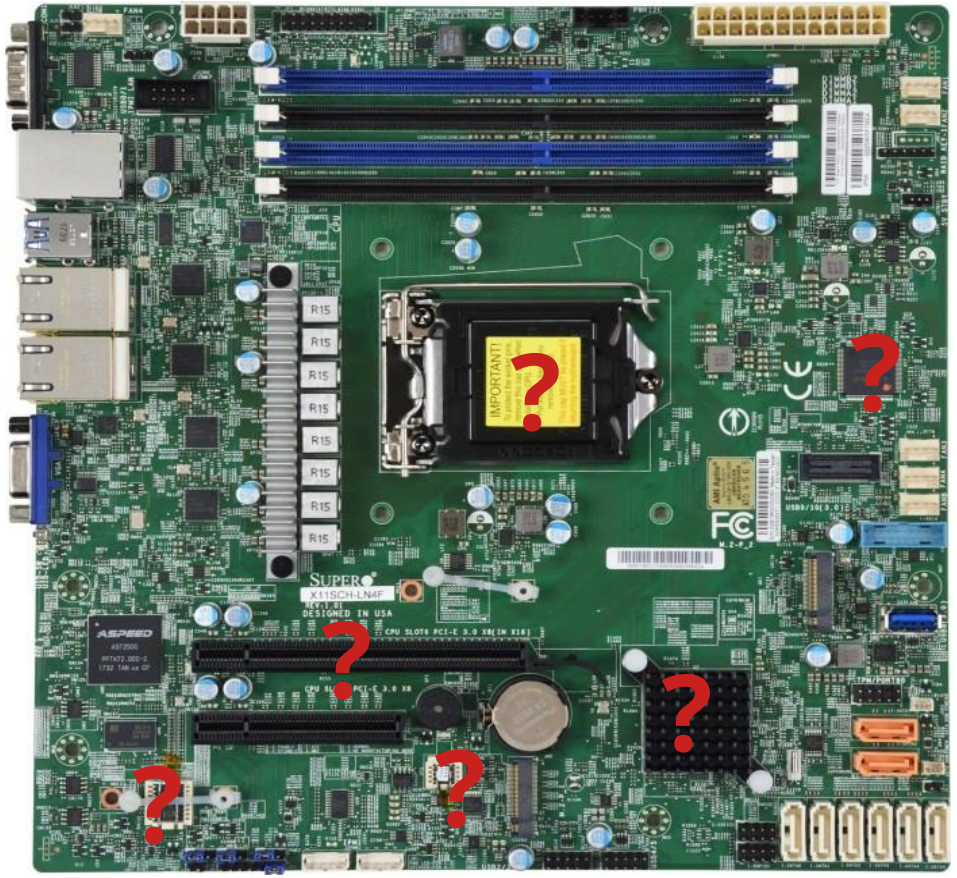
We have more than one firmware?



Red Hat

Where did each firmware come from?

- Who built them?
- When did they build it?
- What OpenSSL did they use?
- What is the licence?
- What is the version?
- What were the source hashes?



Red Hat

UEFI source trees are normally shared between IBV→ODM/OEM

`TagCreator, SoftwareCreator`

`Licensor`

`Distributor`

EFI binaries are also copied from IBV/IHV→ODM/OEM

odm_auth.efi

IBV

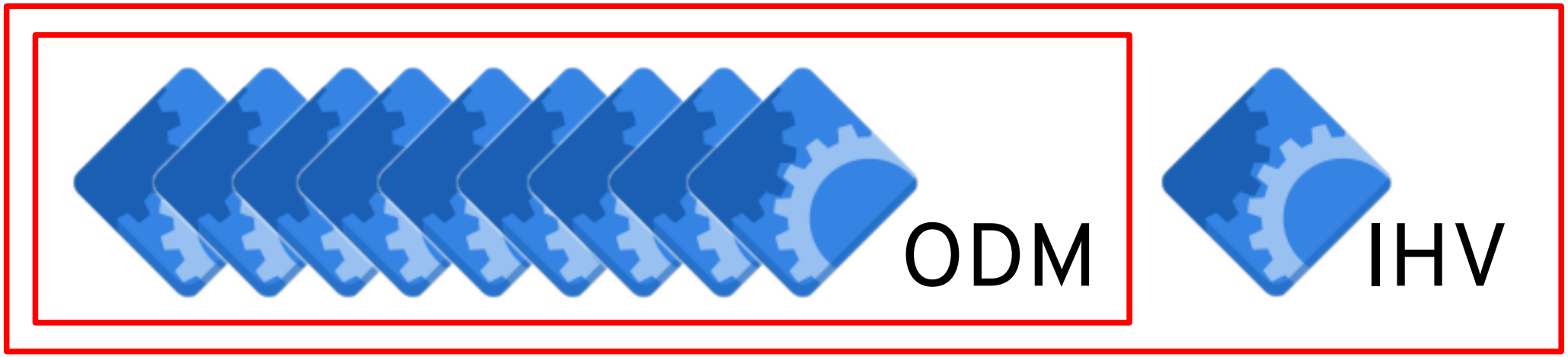
odm_auth_NEW.efi

ODM

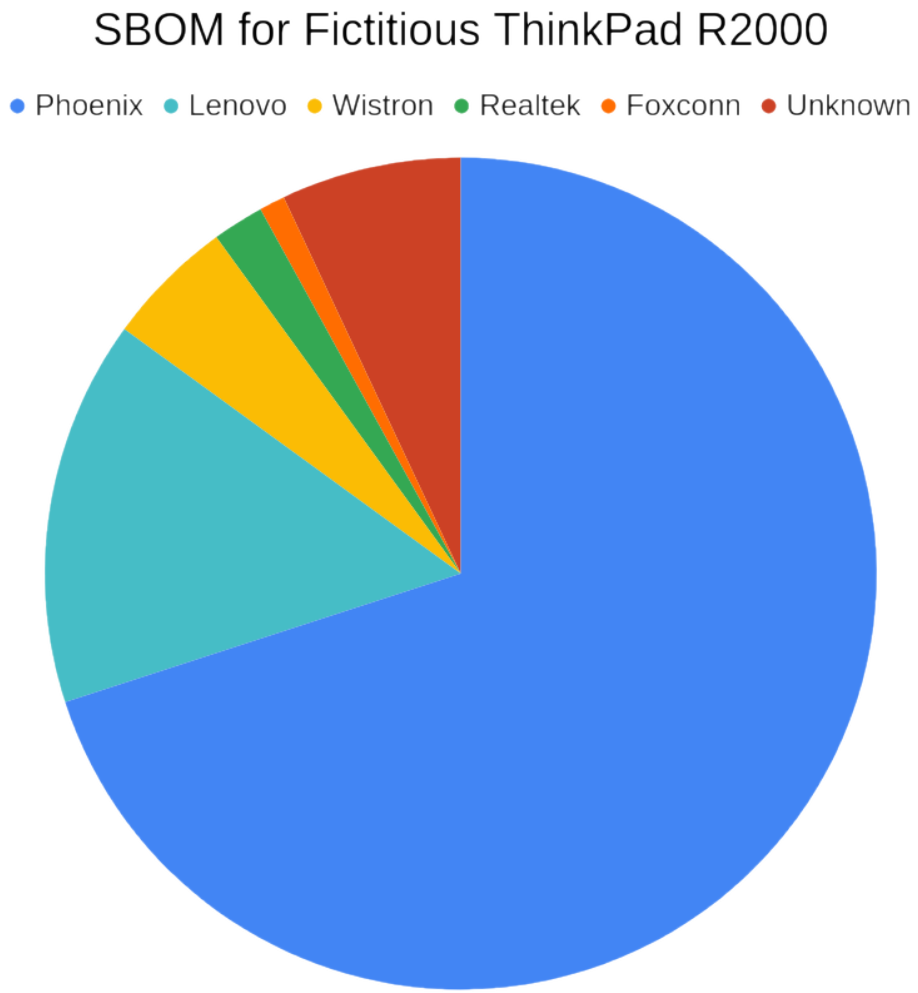
odm_auth_NEW_FINAL.efi

OEM

EFI binaries get added to hierarchical FVs



SBOM via uSWID (for EDK2ish…)



Embed the SBOM data into a SBOM COFF section

- Means it doesn't get stripped
- Which allows the LVFS to extract from Fvs
- **It is always up to date and correct – no validation**
- We don't have to host the data **online** for the next 20 years

Include "detached" metadata for immutable blobs

- Use a magic signature to find uSWID data – anywhere!

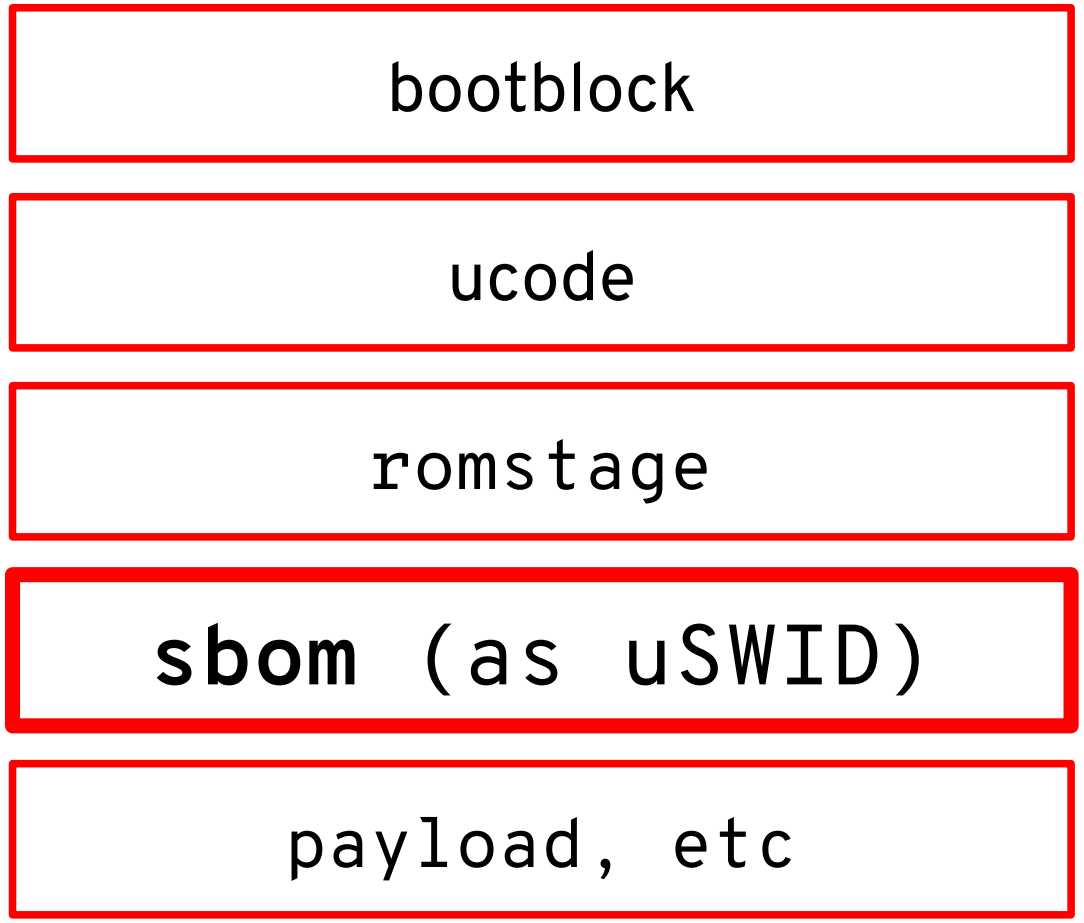
Allow entity "patching" using a simple .ini format

```
[uSWID-Entity:Distributor]
name = OEM Vendor
```

`https://github.com/hughsie/python-uswid`

Red Hat

A New CBFS section for coreboot

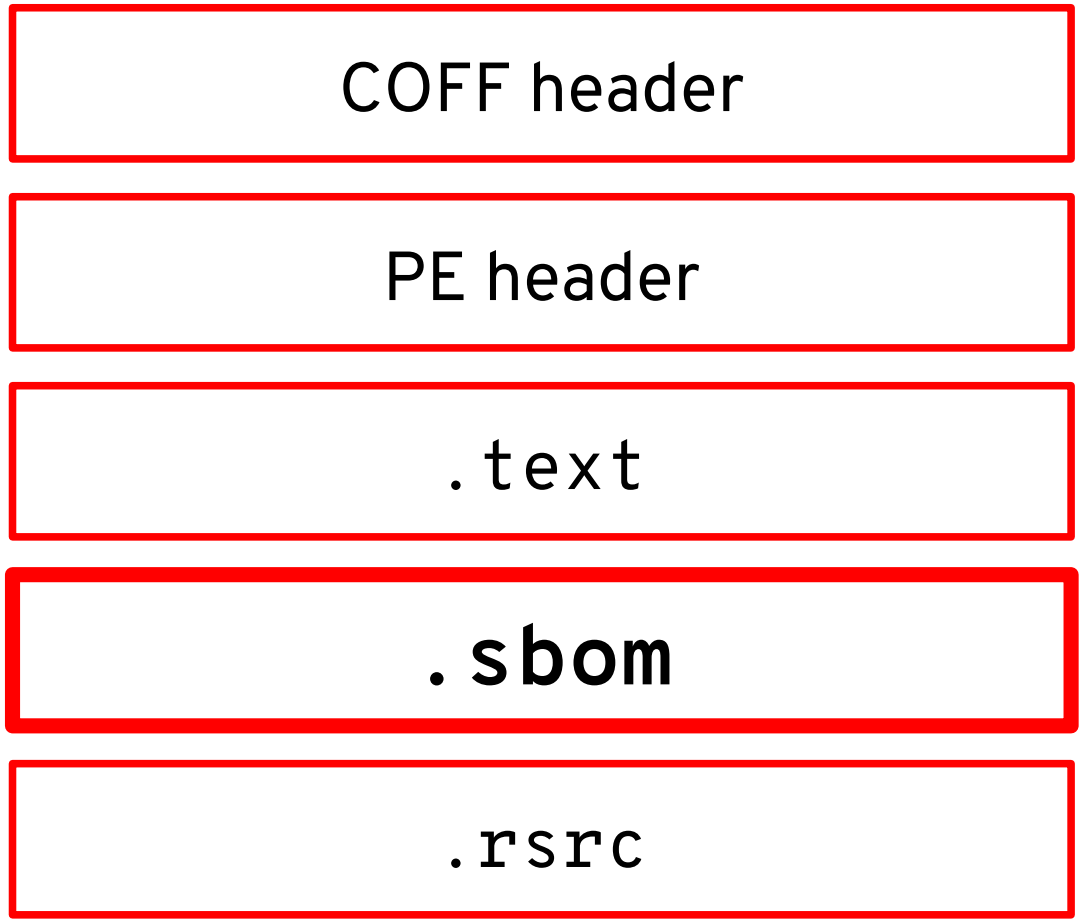


Red Hat

Coreboot community

- **Mar 2022**: Started work on coreboot patch
- **Aug 2022**: Initial patches merged
- **Oct 2022**: Coreboot 4.20 released with uSWID support

A New Optional COFF Section for EDK2



What we have: EDK2 Metadata `.inf`

```
## @file
#  This driver installs SMBIOS information for OVMF
#
#  Copyright (c) 2011, Bei Guan <gbtju85@gmail.com>
#  Copyright (c) 2011 - 2018, Intel Corporation. All rights reserved.<BR>
#
#  SPDX-License-Identifier: BSD-2-Clause-Patent
#
##

[Defines]
  INF_VERSION                    = 0x00010005
  BASE_NAME                      = SmbiosPlatformDxe
  FILE_GUID                      = 4110465d-5ff3-4f4b-b580-24ed0d06747a
  MODULE_TYPE                    = DXE_DRIVER
  VERSION_STRING                 = 1.0

  ENTRY_POINT                    = SmbiosTablePublishEntry
```

What we need to add: IBV & IHV Metadata

```
[uSWID]
tag-id = acbd84ff-9898-4922-8ade-dd4bbe2e40ba
software-name = oem_auth.efi
software-version = 1.2.3
product = Authentication Module
summary = Hughski Super-Secret-Sauce Authentication Module
colloquial-version = b2ed6f1ed8587bf01a2951d74512a70f1a512d38
```

Overriding: ODM & OEM Metadata

```
[uSWID-Entity:Distributor]
name = Richard Hughes
regid = hughsie.com
extra-roles = Licensor
```

```
$ pip install uswid
$ uswid --inifile oem.ini --binfile ./odm_auth_NEW.efi
```

EDK2 is [essentially] Abandonware

- **`Jan 2022`**: Started work: `https://github.com/mefff/edk2/tree/sbom`
- **`Feb 2022`**: Met with AMI and we showed them our patch.
- **`Jun 2022`**: Sent the first patch to edk-devel with good feedback overall.
- **`Jul 2022`**: Met with EDK dev team to discuss. Waited for direct feedback from engineers.
- **`Oct 2022`**: Intel confirmed SBoM support is “on the radar” and would be addressed it in a special meeting.

This meeting never happened and nobody at Intel wants to talk about SBoM.

EDK2 is [essentially] Abandonware (2)

The EDK build system is bespoke, complicated and confusing.

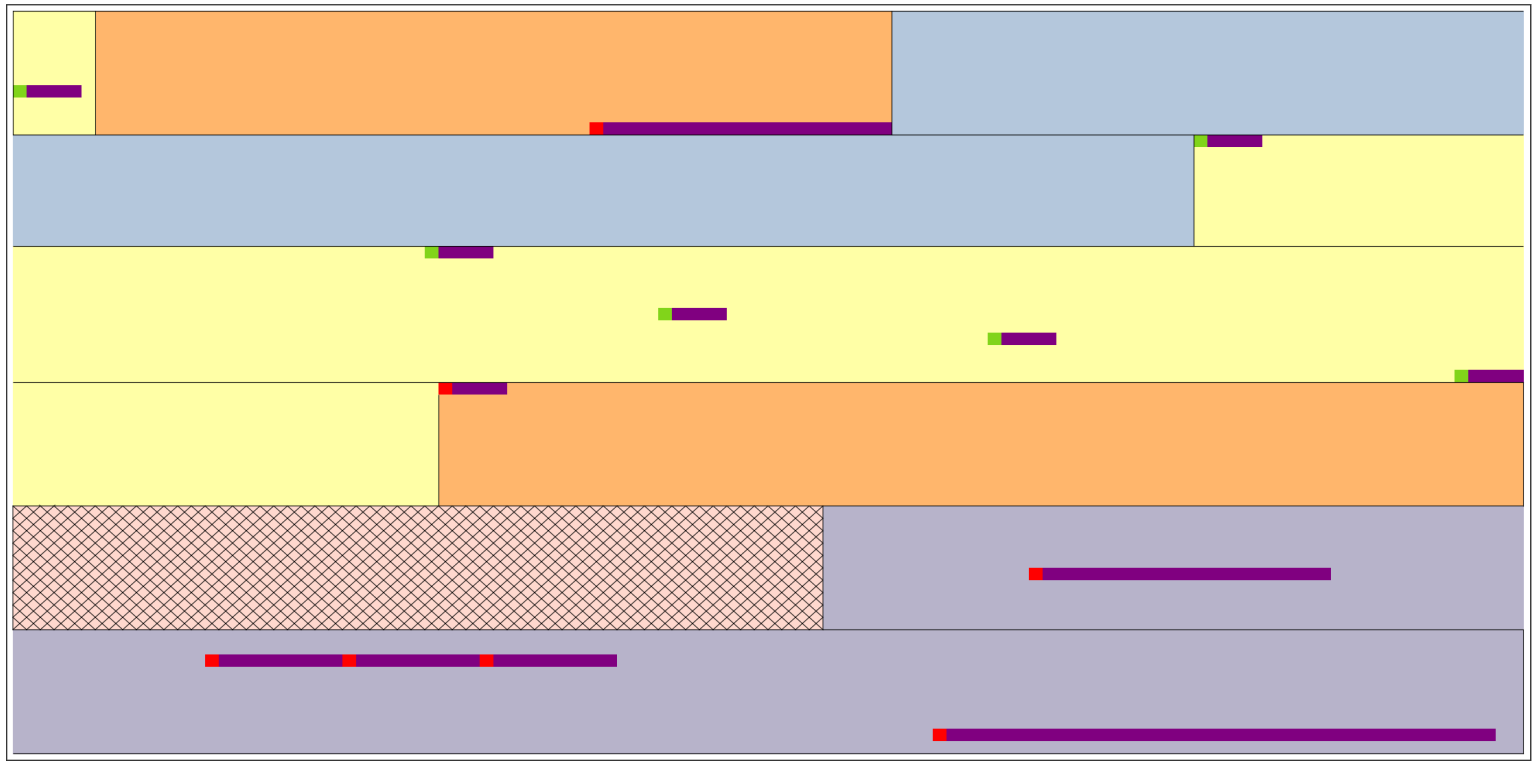
It's split into two parts:

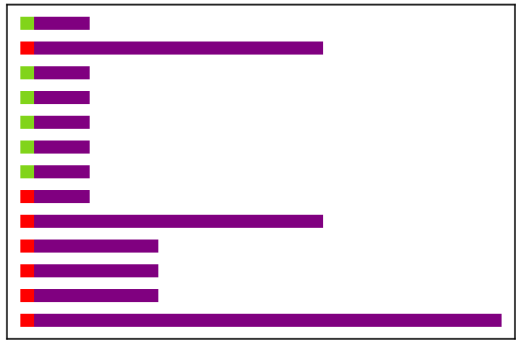
1. A file with a bunch of rules very similar to a Makefile.
2. A framework in python that takes rules, generates files, source files, and runs rules.

We can use those metadata files to generate the SBoM, but we're stuck and nobody knows how it's supposed to work. Most IBVs don't use it.

This is why we're approaching IBVs and ODMs directly now.

Being Pragmatic: Scattering the SBoM is OK!





uSWID sections have no address mapping

- Artifacts like microcode can be detached
- A magic signature is used for uSWID blobs
  - This works even in “empty sections”
- A dedicated section is used for PE/coreboot

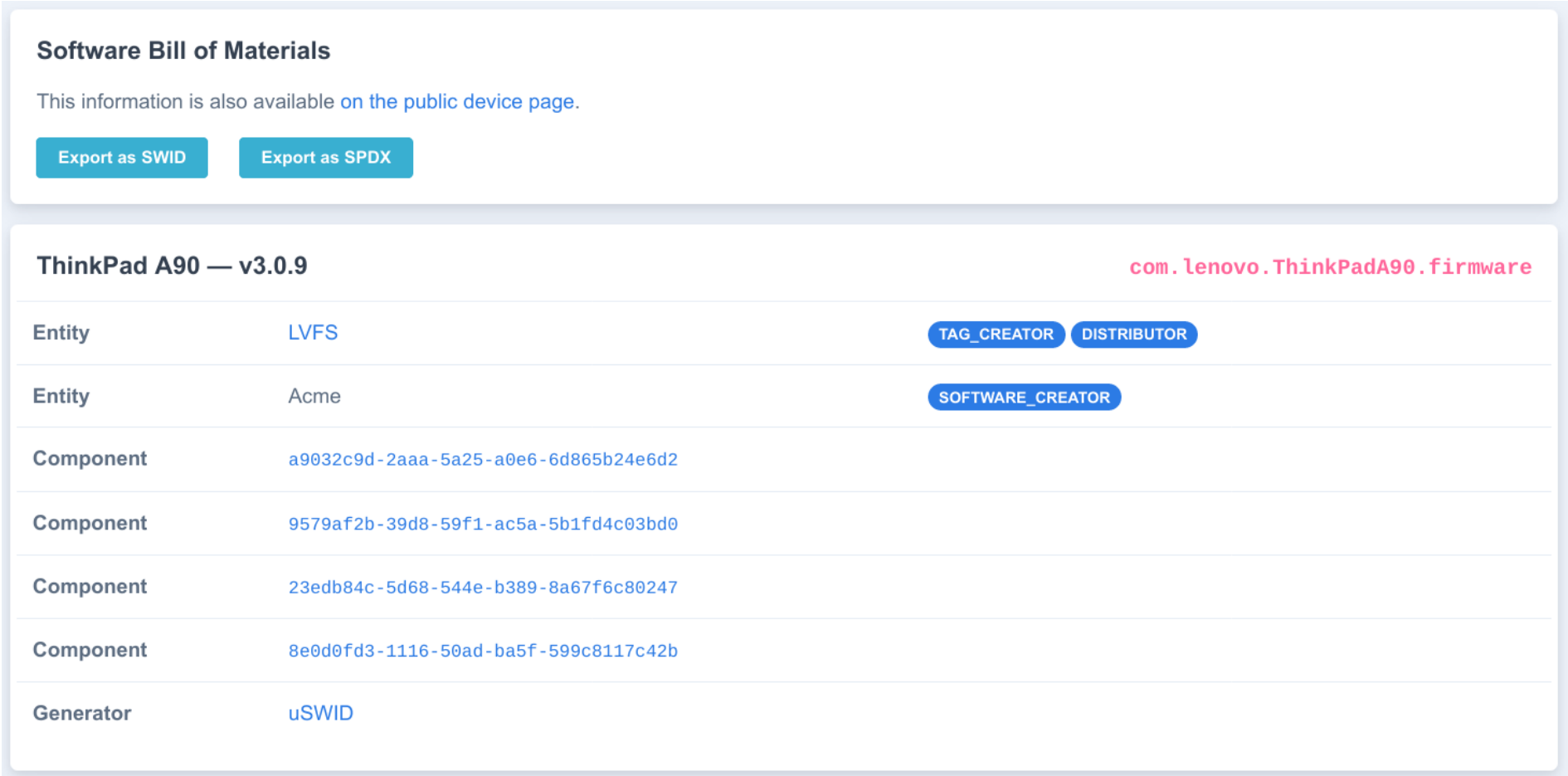
python-uswid and goswid

```
[hughsie@fedora uswid (main %)]$ uswid --verbose --load uswid.ini
Loaded:
uSwidContainer([uSwidIdentity(acbd84ff-9898-4922-8ade-dd4bbe2e40ba,0,HughskiColorHug.efi,1.2.3):
uSwidLink(https://spdx.org/licenses/LGPL-2.1-or-later.html,license)
uSwidEntity(Hughski Limited,hughski.com->TAG_CREATOR)
uSwidEntity(Richard Hughes,hughsie.com->DISTRIBUTOR,LICENSOR,MAINTAINER,SOFTWARE_CREATOR)])
```

https://github.com/hughsie/python-uswid

https://github.com/veraison/swid

LVFS SBoM with SWID & SPDX export



Red Hat

We have to do this **right now**

FEDERAL REGISTER

The Daily Journal of the United States Government

Notice

Software Bill of Materials Elements and Considerations

A Notice by the National Telecommunications and Information Administration on 06/02/2021

Red Hat

Call to action

What we should do:

- Talk about coSWID and uSWID

- Realize that hosting something immutable for 20 years is hard.

Red Hat