The next BPF documents

Dave Thaler

Next steps after ISA

- ~~[PS] the BPF instruction set architecture (**ISA**) that defines the instructions and low-level virtual machine for BPF programs,~~
- [I] **verifier expectations** and building blocks for allowing safe execution of untrusted BPF programs,
- [PS] the **BPF Type Format (BTF)** that defines debug information and introspection capabilities for BPF programs,
- [I] one or more documents that recommend conventions and guidelines for **producing portable BPF program binaries**,
- [PS] cross-platform **map types** allowing native data structure access from BPF programs,
- [PS] cross-platform **helper functions**, e.g., for manipulation of maps,
- [PS] cross-platform BPF **program types** that define the higher level execution environment for BPF programs, and
- [I] an **architecture and framework** document.

verifier expectations

- Intended to describe what a verifier is expected to ensure, e.g.:
  - No undefined instructions or registers
  - Adherence to API contracts (helpers, kfuncs, ctx, etc.)
  - Safe memory pointer dereferencing
  - Termination
- Existing documents to help inform what to put in standardization:
  - verifier.rst currently in tree but not under standardization
    - Contains details about the Linux kernel verifier
  - Academic papers like PREVAIL paper in PLDI ’19
  - Various emails in the archive such as recent thread:
    - [BPF Security] what security properties does verifier guarantee?

BTF

- btf.rst currently in tree but not under standardization
- First step is format transformations to create a file under standardization
  - IETF packet format for structures
  - Tables instead of “#defines”
  - Omit Linux kernel specific info/terms

Platform-specific ABI (psABI)

- The only other file in Documentation/bpf/standardization today
- Proposed scope:
  - Topics for any psABI definition to answer
  - Today’s Linux kernel answers as an example of how to answer
- Maybe also:
  - A specific psABI definition can have an identifier usable with tools (e.g., compilers)
  - Each runtime can then document which psABI it uses (by identifier)
    - IANA registry for IDs and pointers to associated documentation?
  - But we also don’t want to encourage lots of psABIs

ELF profile for BPF

- draft-thaler-bpf-elf is one proposal to document
  - Contains info from elf.rst
  - Contains func/line info from btf.rst (but likely should be in BTF spec once one exists in standardization)
  - Contains some TODOs that also depend on BTF spec (.maps section contents, .BTF* section contents)
  - Needs BTF spec done first or concurrently
- Some debate happened on whether to do in IETF or SystemV or…
  - ELF itself is a SystemV spec
  - Intent was that it is a BPF-specific profile so could be in IETF
    - But defines “e_machine must be set to EM_BPF (247)” in the ELF header
  - Discussion punted until after ISA is done to focus on that one
  - Does “one or more documents that recommend conventions and guidelines for producing portable BPF program binaries,” include psABI and ELF, or only psAPI, or …?

“[Bpf] Registration of e_machine 247 for BPF in gABI”

- Discussion on list back in May
  - In 2016, Richard Henderson registered EM_BPF=247
    1. https://inbox.sourceware.org/libc-alpha/mvmh9c1zglp.fsf@hawking.suse.de/T/
    2. https://groups.google.com/g/generic-abi/c/cmq1LFFpWqU/m/plTYhw8kBwAJ
  - Registry is at https://www.sco.com/developers/gabi/latest/ch4.eheader.html#machine but is no longer maintained so was never updated once registered
  - Propose referencing all three links above with explanation.
    - I’d propose the registry and #2 as normative and perhaps #1 as informative

SCO Registry stopped being updated at 243

EM_VISIUM	221	Controls and Data Services VISIUMcore processor
EM_FT32	222	FTDI Chip FT32 high performance 32-bit RISC architecture
EM_MOXIE	223	Moxie processor family
EM_AMDGPU	224	AMD GPU architecture
	225 - 242	
EM_RISCV	243	RISC-V

https://groups.google.com/g/generic-abi/c/cmq1LFFpWqU/m/plTYhw8kBwAJ

Recent e_machine assignments from Xinuos (registry@sco.com)

157 views

Subscribe

Francois H. Theron

Oct 23, 2017, 9:56:32 AM

to Generic System V Application Binary Interface

When I received my EM_NFP assignment, I also received a list of recent assignment which I may post here for visibility. The following are therefore official e_machine EM_* assignments.

EM_LANAI	244	Lanai processor
EM_CEVA	245	CEVA Processor Architecture Family
EM_CEVA_X2	246	CEVA X2 Processor Family
EM_BPF	247	Linux BPF – in-kernel virtual machine
EM_GRAPHCORE_IPU	248	Graphcore Intelligent Processing Unit
EM_IMG1	249	Imagination Technologies
EM_NFP	250	Netronome Flow Processor (NFP)

Francois

Informative reference too?

```
* [PATCH v2 0/2] Updates for EM_* values
@ 2016-06-20 23:04 Richard Henderson
  2016-06-20 23:04 ` [PATCH v2 2/2] elf.h: Add declarations for BPF Richard Henderso
                   ` (2 more replies)
  0 siblings, 3 replies; 7+ messages in thread
From: Richard Henderson @ 2016-06-20 23:04 UTC (permalink / raw)
  To: libc-alpha

Since my posting two weeks ago, I've gotten an official
number assigned for EM_BPF.  But I thought I'd re-sync
with the draft gabi as well.


r~
```

...

```
diff --git a/elf/elf.h b/elf/elf.h
index 7fc09b9..b6112d9 100644
--- a/elf/elf.h
+++ b/elf/elf.h
@@ -359,7 +359,9 @@ typedef struct
                                /* reserved 225-242 */
 #define EM_RISCV       243     /* RISC-V */

-#define EM_NUM         244
+#define EM_BPF         247     /* Linux BPF -- in-kernel virtual machine */
+
+#define EM_NUM         248
```