eBPF on Windows

Dave Thaler <dthaler@microsoft.com>
# (e)BPF Runtime Platforms

<table>
<thead>
<tr>
<th>Platform →</th>
<th>Linux</th>
<th>MacOSX</th>
<th>Android</th>
<th>FreeBSD</th>
<th>Windows</th>
<th>TockOS (embedded)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Kernel</td>
<td>User</td>
<td>Kernel</td>
<td>User</td>
<td>Kernel</td>
<td>User</td>
</tr>
<tr>
<td>Linux</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2017</td>
</tr>
<tr>
<td>uBPF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rbpf</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2017</td>
</tr>
<tr>
<td>Generic eBPF</td>
<td>2017</td>
<td>2017</td>
<td>2017</td>
<td></td>
<td>2017</td>
<td>2017</td>
</tr>
<tr>
<td>eBPF for Windows</td>
<td></td>
<td></td>
<td></td>
<td>2017</td>
<td></td>
<td>2021</td>
</tr>
<tr>
<td>Tock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2021</td>
</tr>
</tbody>
</table>
eBPF runs in many contexts

- Multiple operating systems
- Kernel mode or user mode
- Main processor, co-processor, SmartNIC, etc.
- Host or Guest VM
- Inside or outside containers
- Native (JIT-compiled) code or interpreted

All using common toolchains and APIs
eBPF program source

Compiler toolchains (clang, bcc, etc.)

eBPF program bytecode

User

Kernel

App, e.g., bpftool
Libbpf, gobpf, etc.

eBPF program bytecode

Verifier

JIT compiler (opt.)

eBPF program native code

eBPF-enlightened component
e.g., TCP/IP stack

eBPF program native code

eBPF execution context

Libbpf, gobpf, etc.
Goal: Cross-plat BPF program source compat

• Pre-BTF ELF format + func/line info from BTF
• PREVAIL verifier: loops, helper functions, etc.
• Program types: 4 (XDP, CGROUP_SOCKET_ADDR, SOCK_OPS, ...)
• Map types: 13 (hash, array, queue, stack, LPM, ringbuf, ...)
• Helpers: 20 (maps, tail call, ringbuf, printk, csum_diff, pid, ...)

• BPF ISA conformance test suite: same tests for both Linux and Windows
  • Atomics support not done yet for PREVAIL/uBPF

• E.g.: Cilium layer-4 load balancer port uses >95% of code unmodified
Questions?