Confidential Computing
Guest Image Deployment

Jianxiong Gao <jxgao@google.com>
Marc Orr <marcorr@google.com>
Available and upcoming CVM features

- **AMD**
  - [supported] SEV
  - [supported] SEV-ES
    - [supported] Live Migration (host-to-host)
  - [supported] SEV-SNP
    - [upcoming] Lazy accept
    - [upcoming] HW-rooted measured boot
    - [upcoming] Confidential IO
    - [upcoming] Restricted Interrupt Injection

- **Intel**
  - [supported] TDX
    - [upcoming] Lazy accept
    - [upcoming] HW-rooted measured boot
    - [upcoming] Confidential IO

- ...
Features are not necessarily additive

- Google cloud’s experience w/ SEV (so far)
  - Generation 0: SEV
  - Generation 1: SEV w/ lazy pinning
- What’s next? (illustrative only; not a roadmap)
  - Generation 2: SEV + lazy pinning + live migration
  - Generation 3: SEV-SNP
  - Generation 4: SEV-SNP + lazy accept
  - Generation 5: SEV-SNP + lazy accept + live migration

=> Different architectures will get sub-features at drastically different rates

=> Features are NOT strictly additive
What could go wrong?

- **Live Migration**
  - Control plane configures guest to live migrate
  - Customer uses VM for a while. Everything is working great.
  - After days or weeks, control plane tries to migrate CVM => Oops!

- **Lazy Accept**
  - Guest FW thinks guest kernel supports lazy accept
  - Guest FW accepts minimal amount of memory (e.g., 4GB)
  - Guest kernel cannot see unaccepted member => Oops!

- **HW-rooted measured boot**
  - Should fail early on!

=> Best: Features “just work”

=> 2nd best: VM dies early on
What can we do?

- Get feature in from day zero
- Guest queries itself for its features
- Feature negotiation
- Image annotation
Get feature in from day zero

- Avoids “roll out” issues from the getgo
- Can be a good solution when viable
  - Can also complicate launches by making them less incremental
  - Generally, adding features incrementally is easier and less risky
- Not always viable
Guest queries itself for its features

**Guest FW:**

```c
struct vm_config = GetVmConfig();
struct fw_features = GetFwFeatures();
if (vm_config.feature_present && !fw_features.feature_present)
    self_terminate();
```

**Guest Kernel:**

```c
struct vm_config = GetVmConfig();
struct kernel_features = GetKernelFeatures();
if (vm_config.feature_present && !kernel_features.feature_present)
    self_terminate();
```
Feature Negotiation

Guest FW:

```c
struct vm_config = GetVmConfig();
struct kernel_features = GetKernelFeatures();
if (vm_config.feature_present && !kernel_features.feature_present)
    self_terminate();
```

Discussed upstream a bit in the context of lazy accept:

- [PATCHv7 00/14] mm, x86/cc: Implement support for unaccepted memory
  - [xuliog@kernel.org](mailto:xuliog@kernel.org)
  - [https://lore.kernel.org/linux-mm/CAMkAt6osbEGBFrkn=y1=x4mDHC1aL40BwaW0NdGHF8qmWd7ktA@mail.gmail.com](https://lore.kernel.org/linux-mm/CAMkAt6osbEGBFrkn=y1=x4mDHC1aL40BwaW0NdGHF8qmWd7ktA@mail.gmail.com)
- UEFI bug titled "GetMemoryMapEx"
  - [https://bugzilla.tianocore.org/show_bug.cgi?id=3987](https://bugzilla.tianocore.org/show_bug.cgi?id=3987)
## Image annotation: Feature Matrix

- **Tagging**
  - Communicate with customer what each image supports
  - Tagging images with `--tdx=live_migration,lazy_pvalidate,upm` to indicate what each image supports
  - Is there a common/standard way?

<table>
<thead>
<tr>
<th></th>
<th>Live Migration</th>
<th>Lazy-accept</th>
<th>etc</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEV-SNP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDX</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Image annotation: Version Info

Example annotation:

OS: Ubuntu Version: 22.04.03 Kernel: 5.17.11-1ubuntu20

Control plane can gate enabling features with logic like:

def HasLazyAccept(guest_os_name, 
    guest_os_image_version, 
    guest_os_kernel_version):
    # return True if guest_os_kernelI_version >= min kernel version
    # where lazy accept appeared in guest OS. Otherwise, return False.

Cons:

● Custom images
● What if customer downgrades their kernel?
Thanks!