DRM2
Let's fix the DRM authentication policy and buffer sharing

Kristian Høgsberg, Martin Peres, Timothée Ravier & Daniel Vetter

X.org foundation

September 19 – 21, 2012

Summary

Current DRM authentication scheme

DRM's security model

- The DRM master has all the rights;
- A DRM client can ask the DRM master to authenticate him;
- Once authenticated, a client can basically do everything it wants.

Who can be the DRM master?

- Needs to have the CAP_SYS_ADMIN (root);
- One master at the time;
- The first one to request it gets it;
- The DRM master rights can be released.

Current DRM authentication scheme

Who actually needs to be the master?

- XServer(s);
- Anything else?

The X Server and DRM_MASTER

- Limit modesetting calls to the currently active X-Server;
- Isolate applications between X-Servers?

Current DRM authentication scheme

X-Servers and VT-switches

- X-servers are located on different VT;
- Switching user sessions is done by doing a VT-switch;
- Before VT-switching, the X-Server must release the DRM_MASTER rights;
- When entering the VT, the new X-Server must acquire the DRM_MASTER rights;
- If it fails, the new X-Server cannot authenticate new clients.

Buffer-sharing with GEM

- Clients need to be authenticated/associated with a MASTER;
- A GEM buffer is shared by calling the GEM flink IOCTL;
- This buffer is then shared between all the "minors" authenticated by the MASTER (rw).

Summary

DRM's authentication problems

VT-switching problems

- There is a potential synchronisation problem when switching;
- A malicious root application may try to acquire the DRM_MASTER rights in a loop and steal them from a legitimate X-Server when a VT-switch occur.

Confidentiality/Integrity problems

Applications within a X-Server can access others' shared buffers (GEM flink).

Non-graphical applications (GPGPU, video encoders)

Non-graphical applications cannot ask the X-Server to authenticate them in order to access the GPU.

Summary

Fixing Buffer Sharing

Fixing GEM buffer sharing

- Split MASTER into MASTER and GEM_MASTER;
- Allow multiple GEM_MASTERs;
- Only root users can became GEM_MASTERs;
- Allow sharing only from minors to masters;
- It doesn't break the userspace!
- $\rightarrow$ mitigates GEM sharing's security problem;
- $\rightarrow$ encourages the use of DMA-Buf for new applications!

Dropping some privileges requirements

Allow non-authenticated GPU clients

- Non-GEM-flink users may not pose security problems;
- They should be able to access the GPU without a MASTER (x-server);
- Let the driver/hw actually isolate GPU users;
- Isolation can be done using GPU VMs/pushbuf validation;
- $\rightarrow$ allow GPGPU/video encoding without a MASTER.

Problems

- Some drivers/devices may not be able to isolate GPU users;
- What should be done?
- Export a DRM attribute (provides_client_isolation)?
- let udev change the permissions to only allow root users?

Fixing MASTER switching

Is VT really needed nowadays?

- KMSCon can become a system compositor;
- It can be responsible for forwarding events to the right compositor;
- It can allow compositors to change the modesetting (only the current one);
- It can provide terminals and deprecate VTs.

What should be done with the MASTER mode?

- If using KMSCon as a system compositor;
- KMSCon would acquire and never release the MASTER attribute;
- We can leave MASTER as is!

Further secure DMA-Buf

DMA-Buf

- Allows sharing buffers with only the needed clients;
- But we cannot specify the sharing rights;
- We could use LSM to allow (pwrite/pread/mmap/unmmap);
- We can use SELinux to do access control on DMA-Buf;
- That will complete buffer sharing security.

Thank you for listening!
Questions or other propositions?