

NET Token

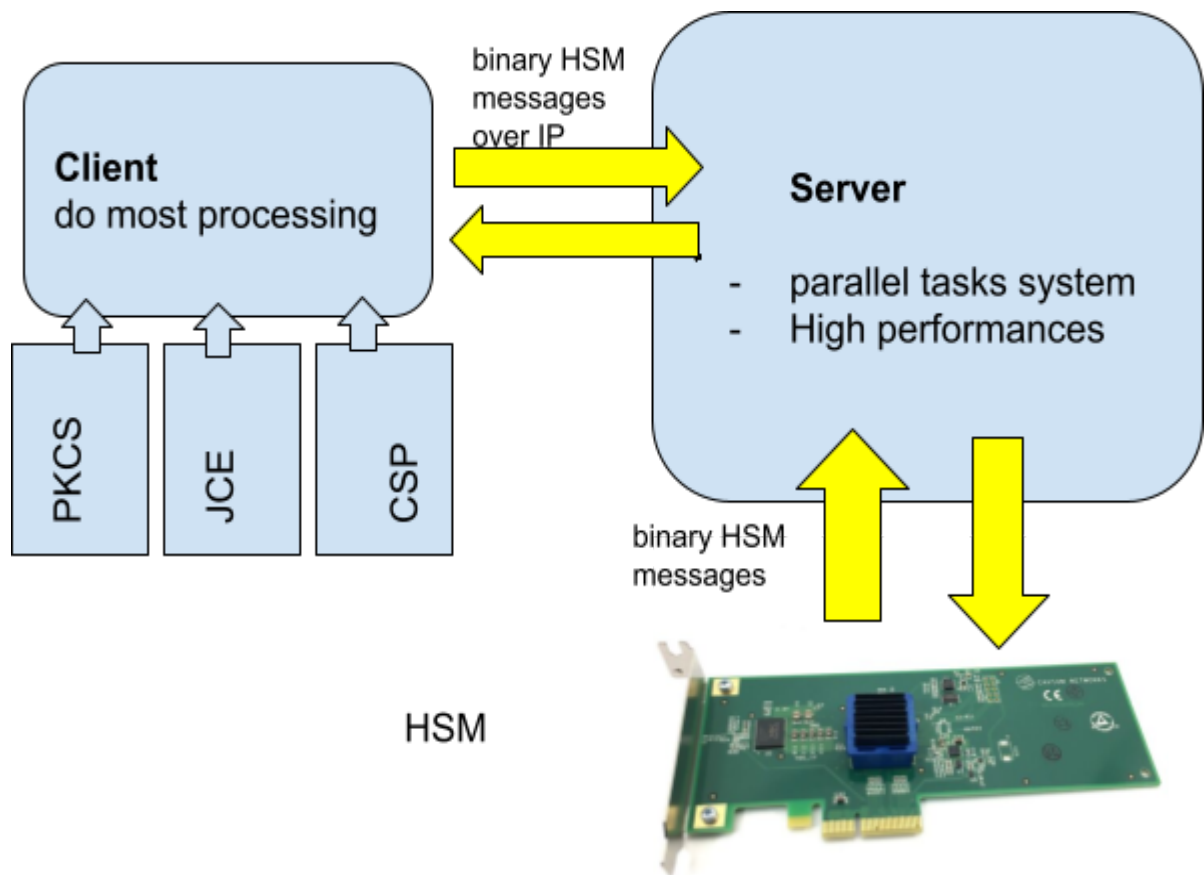
Developed by SCD

Goal: providing PKCS, JCE and CSP access to network-based HSMs

- Entirely developed in plain C++
- Tested according to PKCS#11 standards
- Performances are similar to the target HSM
- Parallel calls managed by a dedicated system
- Client and server version for centos, RHEL and windows 64 bits
- All PKCS features functions available via the web remotely including certificates management



```
CommandProcessor.cpp | talker.cpp | Value.cpp | utests.cpp | TextSocket.cpp | ProtoCommand.h
talker (Global Scope) | get_attribute_value(Talker::HsmContext & c
461     static void
462     set_attribute_value(Talker::HsmContext& context, JsonBox::Value& request, JsonBox::Value& respo
463     {
464         std::cerr << "set_attribute_value" << std::endl;
465
466         JsonBox::Value& args = request["args"];
467         CK_SESSION_HANDLE hSession = args["session_handle"].getInt();
468         CK_OBJECT_HANDLE hObject = args["object_handle"].getInt();
469
470         std::vector<Marshalling::AttributeWrap> attrs = Marshalling::json_to_template(args["templat
471         std::cerr << "attributes here:" << NetCard::dump_attributes((CK_ATTRIBUTE_PTR)&attrs[0], (C
472         SmartCards::EToken::tok_chk(context.getFunctionList()->C_SetAttributeValue(hSession, hObjec
473         response["args"]["result"].setString("OK");
474     }
475
476
477     static void
478     get_attribute_value(Talker::HsmContext& context, JsonBox::Value& request, JsonBox::Value& respo
479     {
480
481     try
482     {
```



Net Token was a system allowing remote access to an enterprise-grade HSM from a standard PC using a special driver. Locally a PKCS#11 DLL containing all the PKC#11 entry points was present on the PC and under the hood, it was doing remote calls to a PKCS#11 proxy server, handling calls to the HSM. With such a system, access to an HSM could be shared by thousands of customers. A plug-in was developed, cert++, allowing certificates to be also securely remotely stored.

Additionally we developed a CSP and JCE driver, wrapping the PKCS#11 calls.

[Recording #1.mp4 - Google Drive](#)

[Recording #2.mp4 - Google Drive](#)

For the work, we developed from scratch a **custom PKC#11 driver**.