OpenEapSmartcard.NET

A .NET EAP smartcard that fully controls accesses to LAN, WLAN and VPN resources

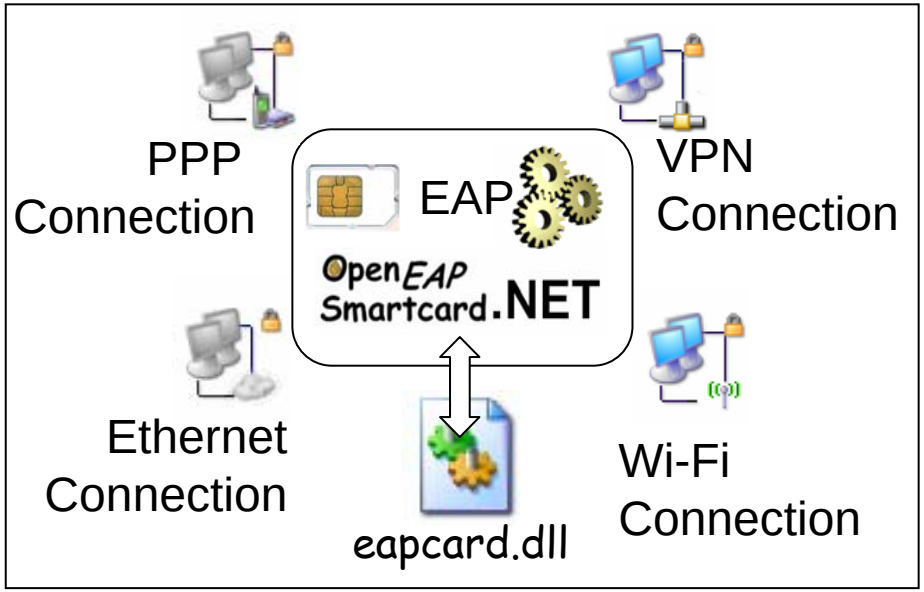
Pascal.Urien@enst.fr

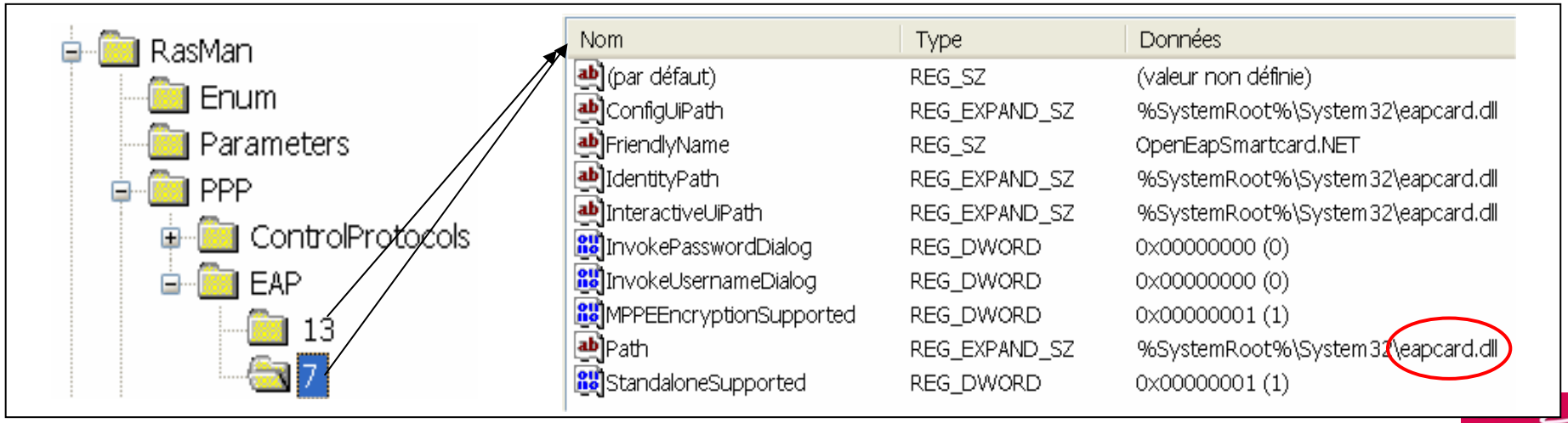
TELECOM PARIS
école nationale supérieure des télécommunications

What is OpenEapSmartcard.NET ?

A new concept of smartcards that enhances the WEB security privacy and mobility for

- ✓ PPP
- ✓ Ethernet
- ✓ Wi-Fi
- ✓ VPN
- ✓ And soon WiMax

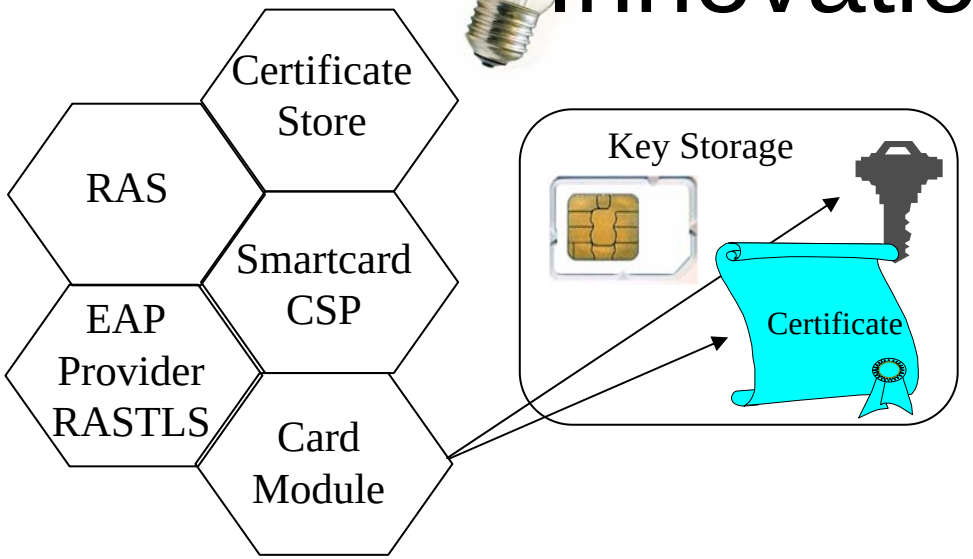




Classical (VISTA) Architecture

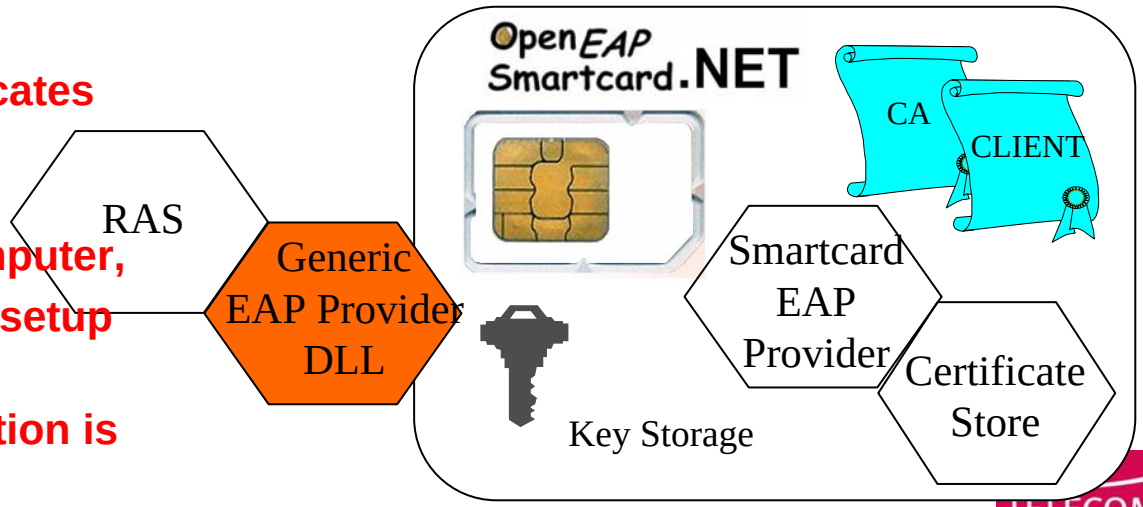
- Classical solution works with multiple components, e.g. Certificate Store, RAS, Smartcard CSP, Card Module, EAP provider
- Certificates need previous set up.
- Sometimes user is prompted, in order to reject or accept a certificate.

Innovation



OpenEapSmartcard.NET

- ✓ **Embedded and standalone EAP provider (e.g. EAP-TLS application),**
- ✓ **Dual factors authentication, PINcode + Smartcard,**
- ✓ **More security, embedded certificates checking, no user prompt,**
- ✓ **Enhanced privacy, no personal information is stored on the computer,**
- ✓ **Enhanced mobility, no previous setup is required,**
- ✓ **Remote (standalone) administration is possible via TLS messages.**



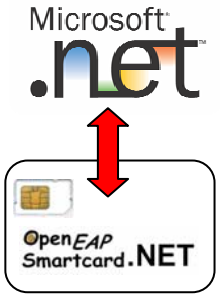
OpenEapSmartcard.NET Services

- Two classes of services

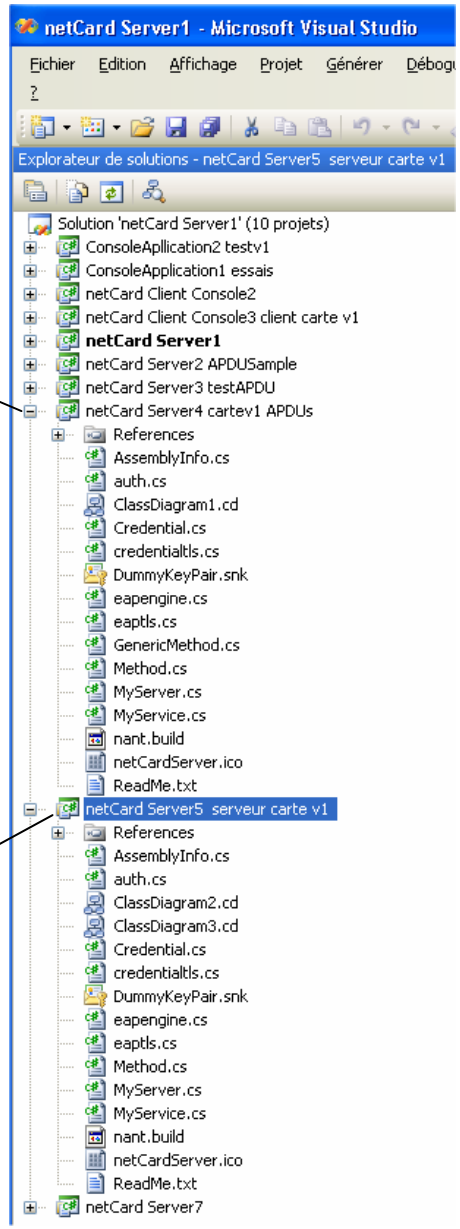
DLL ✓ Legacy services.

- EAPCARD.DLL is a system library that was previously written for classical ISO 7816 smartcards. In order to modify only the smartcard side, OpenEapSmartcard.NET implements a classical (legacy) interface, thanks to ***APDU attributes***, compatible with these constraints. This is showed in demo 2.

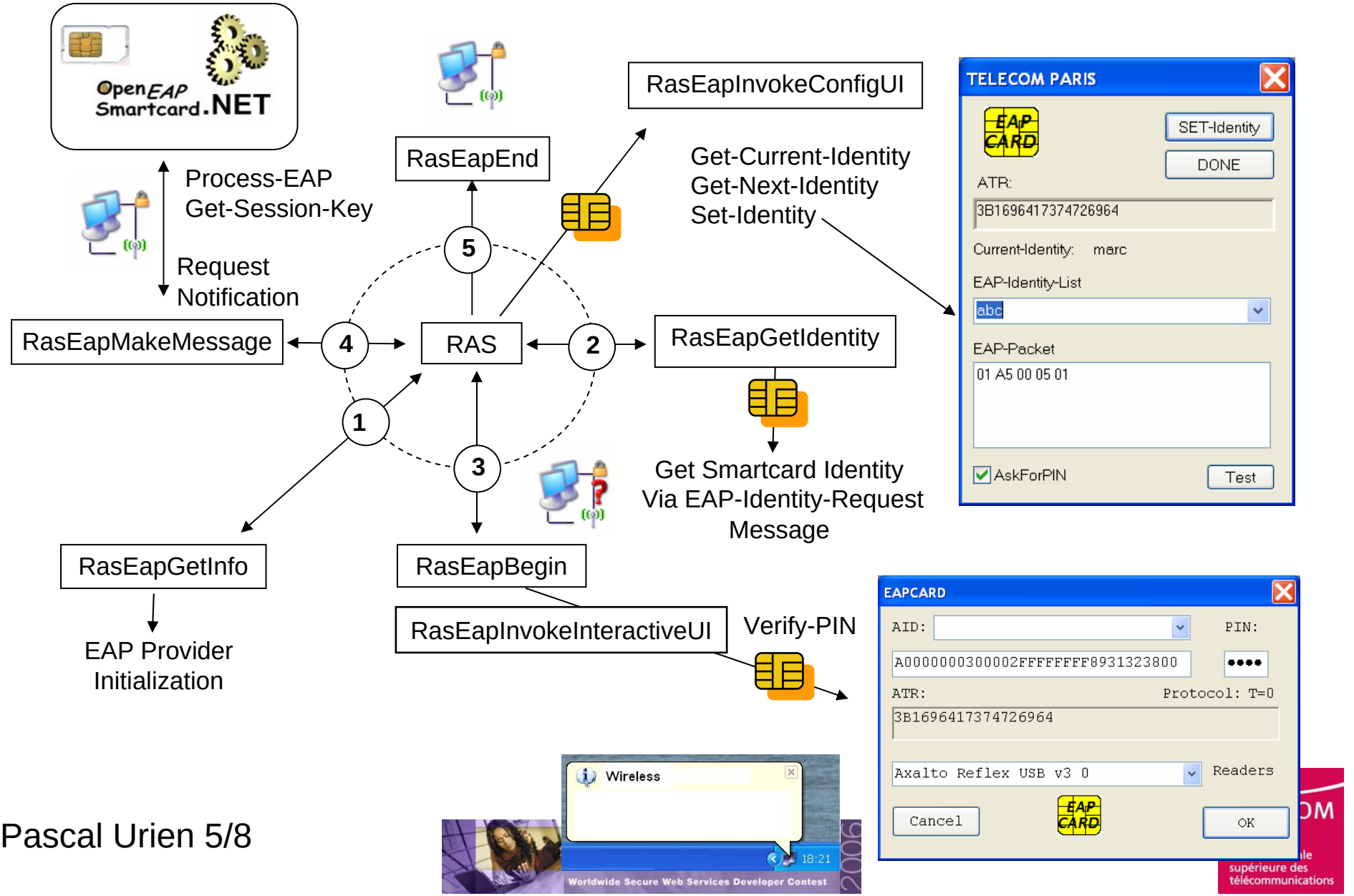
✓ .NET services.



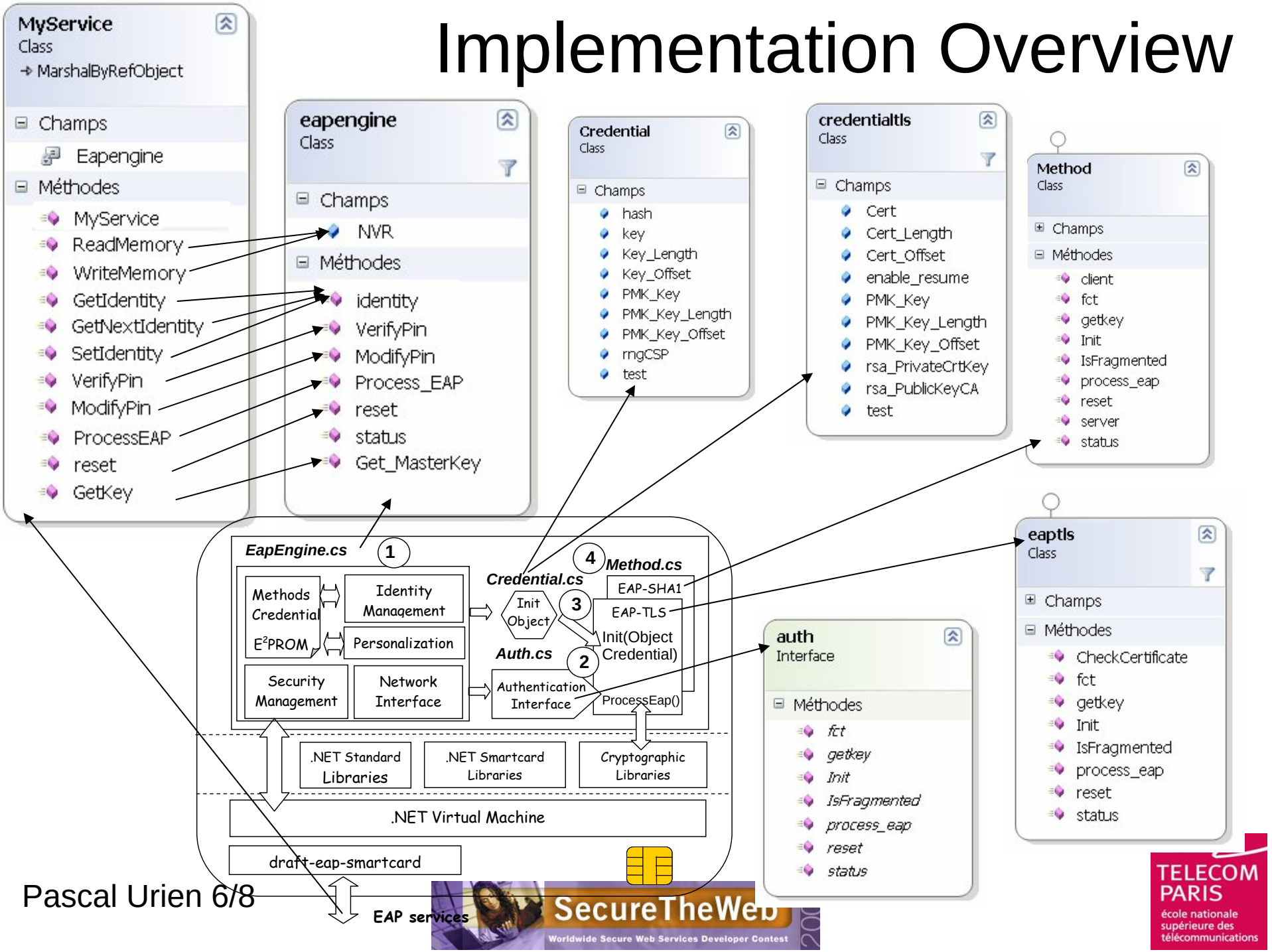
- OpenEapSmartcard.NET offers highly secure and mobile **.NET services** for EAP client and server entities. But obviously there are not yet customers for these innovative facilities, that could be used both on terminal and server (such as RADIUS) side. This is illustrated in demo 1.



Integration in Windows Computers



Implementation Overview



```
done = service.VerifyPin(s2b("0000"));
done = service.WriteMemory(-2, a2b("80"));
Console.WriteLine("WRITE (Offset=-2, value=x80)");

bin = service.ReadMemory(-3, 3);
Console.WriteLine("READ(Offset=-3, Length=3): " + b2s(bin));
for (i = 0; i < 7; i++)
{ id = service.GetNextIdentity();
  Console.WriteLine("GETNEXT: " + b2a(id));}

id = service.SetIdentity(s2b("marc"));
Console.WriteLine("SET: " + b2a(id)) ;

id = service.GetIdentity()          ;
Console.WriteLine("GET: " + b2a(id));
```

Microsoft .net

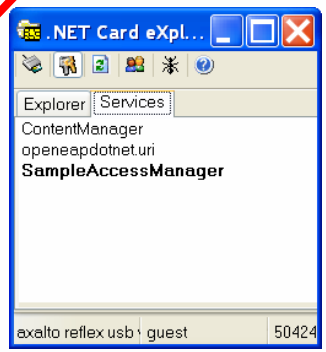
Console3.exe

Server5.exe

OpenEAP Smartcard.NET

```
file:///D:/Documents and Settings/urien/Mes docume
WRITE (Offset=-2, value=x80)
READ(Offset=-3, Length=3): ff8054
GETNEXT: a
GETNEXT: test
GETNEXT: c
GETNEXT: marc
GETNEXT: abc
GETNEXT: aaa
GETNEXT: a
SET: marc
GET: marc
REQ : 01a5000501
RESP: 02a50009016d617263
REQ : 01a6001a071483d972d101f40973dec8e32068b1de581641ea76
RESP: 02a6001a0714f23eedcfcf2eca7d5390435769d625a35624612e
MSK_Client f98d3ad3b6de88f21679523837fba230825c19220de75d5e01b3c116931593206214c
187e73fe159cff5c821de0bf47b4f7f3f4a07c9dd606b35aecd42c4b48d
REQ: 03a60004
MSK_Server f98d3ad3b6de88f21679523837fba230825c19220de75d5e01b3c116931593206214c
187e73fe159cff5c821de0bf47b4f7f3f4a07c9dd606b35aecd42c4b48d
SET: abcd
REQ : 011400060d20
RESP: 021400500d8000000046160301004101000003d03013faa2b6a08bdd285b43d1f3bc9715fc9
f85fc453fe58f3a9e07ff397cd6539220000160004000500a0009006400620003000600130012006
30100
```

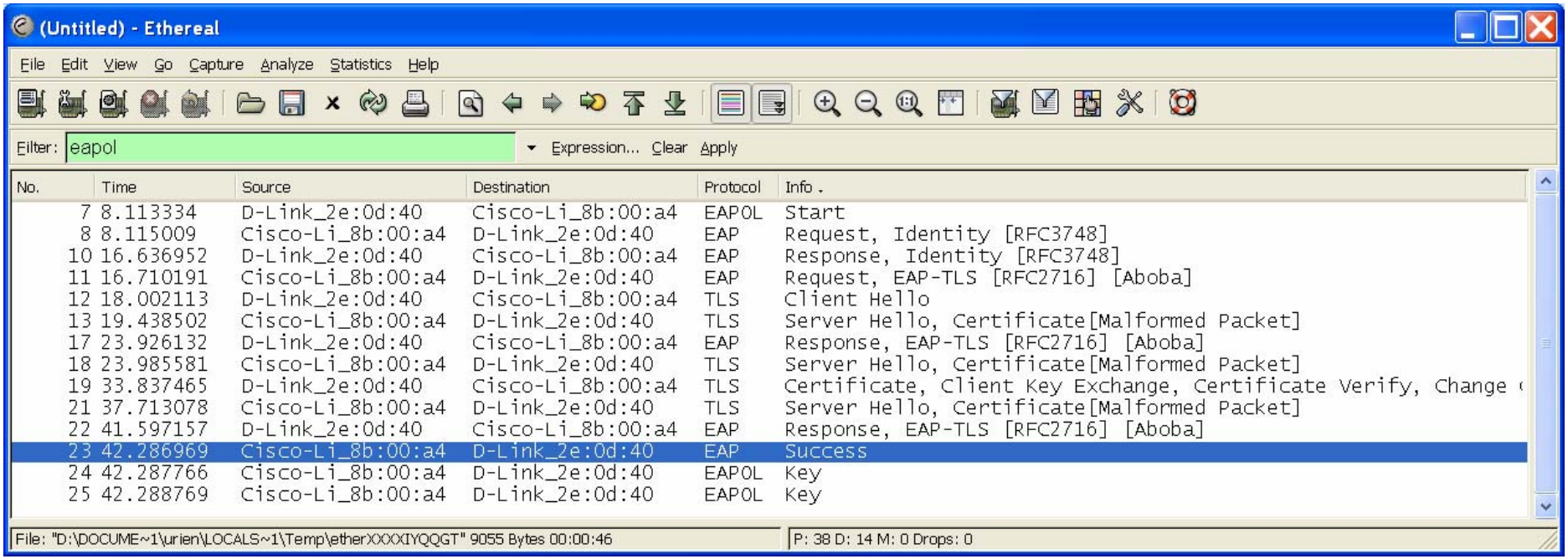
Standalone dialog between EAP Client and Server entities

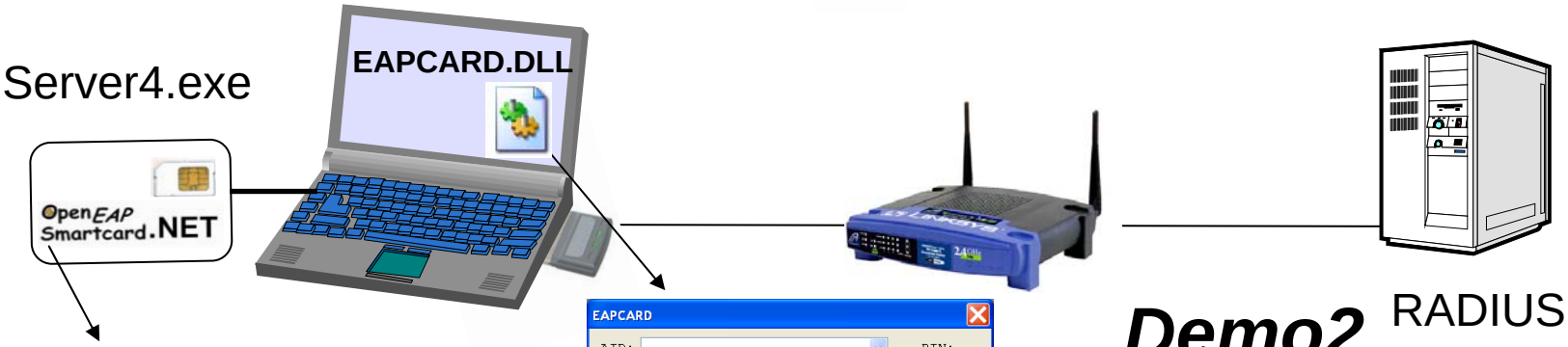


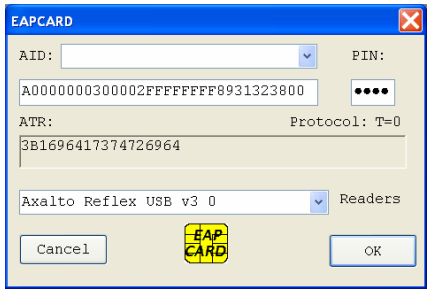

```
//=======================================
//            Test EAP-SHA1
//=======================================
eapreq = a2b("01 A5 0005 01");
Console.WriteLine("REQ : " + b2s(eapreq));
eapresp = service.ProcessEAP(false, eapreq);
Console.WriteLine("RESP: " + b2s(eapresp));

eapreq = service.ProcessEAP(false, eapresp);
Console.WriteLine("REQ : " + b2s(eapreq));

eapresp = service.ProcessEAP(false, eapreq);
Console.WriteLine("RESP: " + b2s(eapresp));
mkey = service.GetKey();
Console.WriteLine("MSK_Client " + b2s(mkey));

eapreq = service.ProcessEAP(false, eapresp);
Console.WriteLine("REQ: " + b2s(eapreq));
mkey =  service.GetKey();
Console.WriteLine("MSK_Server " + b2s(mkey));

////////////////////////////////////////////
//              Test eap-tls
////////////////////////////////////////////
id = service.SetIdentity(s2b("abc"));
Console.WriteLine("SET: " + b2a(id));

eapreq = a2b(pkt1);
Console.WriteLine("REQ : " + b2s(eapreq));
eapresp = service.ProcessEAP(false, eapreq);
Console.WriteLine("RESP: " + b2s(eapresp)) ;
```

Demo 1 .NET Services Client & Server

TELECOM PARIS
école nationale supérieure des télécommunications







RADIUS

Standalone EAP-TLS

- ✓**High security**
- ✓**High privacy**
- ✓**High mobility**

Demo2

Wi-Fi, EAP-TLS Authentication

Pascal Urien 8/8

TELECOM PARIS
école nationale supérieure des télécommunications