

IpMorph :

« vers la mystification de la prise d'empreinte de pile »



Présentation à l'OSSIRB
17 février 2009

<http://dev.ipmorph.org>



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Sommaire

- **Contexte**
- **Historique**
- **Etymologie**
- **Techniques de détection**
- **Etat de l'art de la mystification**
- **Utilisation d'IpMorph**
- **IpMorph != Packet Purgatory / Morph**
- **Socle logiciel**
- **Architecture générale**
- **Translation ARP**
- **Mystification de la congestion**
- **Focus : Calcul ISN**
- **Focus : Interfaces réseaux « read/write »**
- **Personality Manager**
- **Démonstration**
- **Etat d'avancement**
- **Perspectives**
- **Bibliographie**



IpMorph : « vers la mystification de la prise d'empreinte de pile »

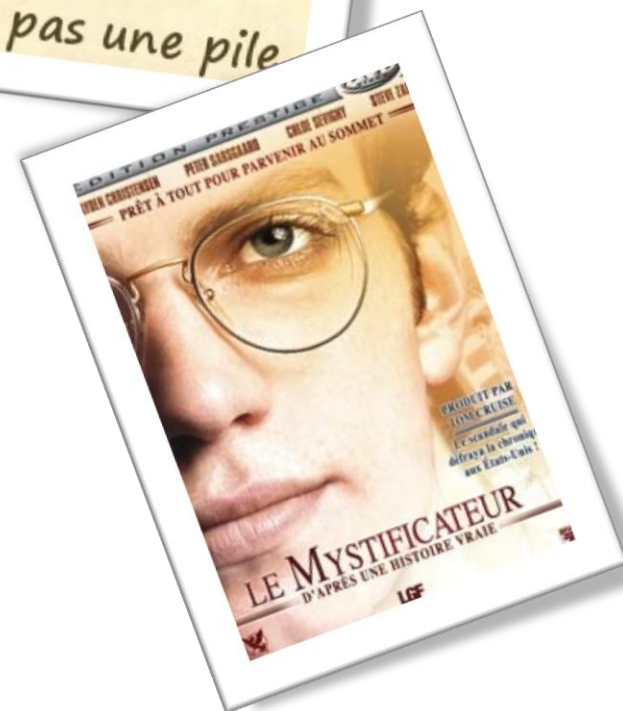
Contexte

Théorème :

« Vivons heureux, vivons caché »

Corolaire :

« Si une machine peut falsifier son identité au yeux des outils de reconnaissance et usurper celle d'un système moins intéressant pour la menace, celle ci minimise l'attrait de l'attaquant et perturbe la pertinence des attaques ciblées à sa nature apparente.»

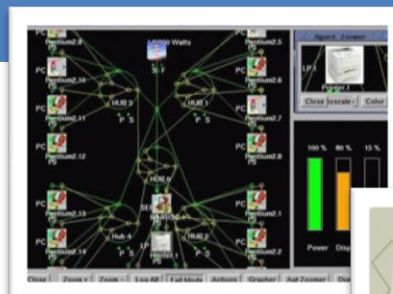




IpMorph : « vers la mystification de la prise d'empreinte de pile »

Historique

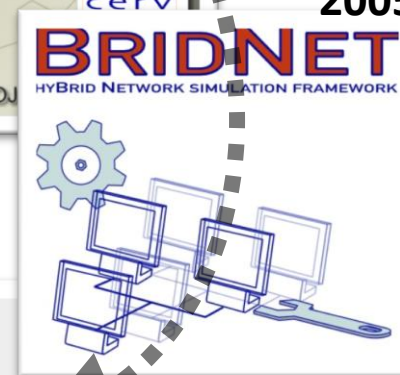
- **GFOX-0 & GFOX-1**
 - Simulation multi-agents de la confidentialité, de l'intégrité et de la disponibilité d'un système d'information
- **EREBOR**
 - Vers l'hybride « réel/virtuel »
- **BridNet**
 - Première pile « UserLand TCP/IP »
- **Hynesim**
 - Plateforme de virtualisation de système d'information



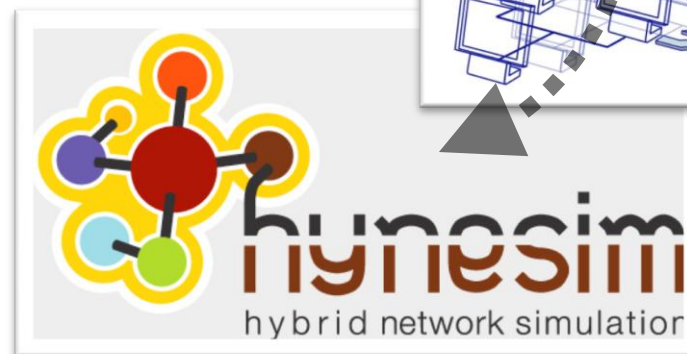
1999



2002



2005



2008-?



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Etymologie

– IpMorph

- Du néo-grec numérique [μορφή - θημωνιά](#), *morphéstakis*, « [forme de pile](#) ».

– Suffixe

- -morph [/mɔʁf/](#)

1. En relation avec la forme d'une pile IP, qui a la forme d'une pile IP.

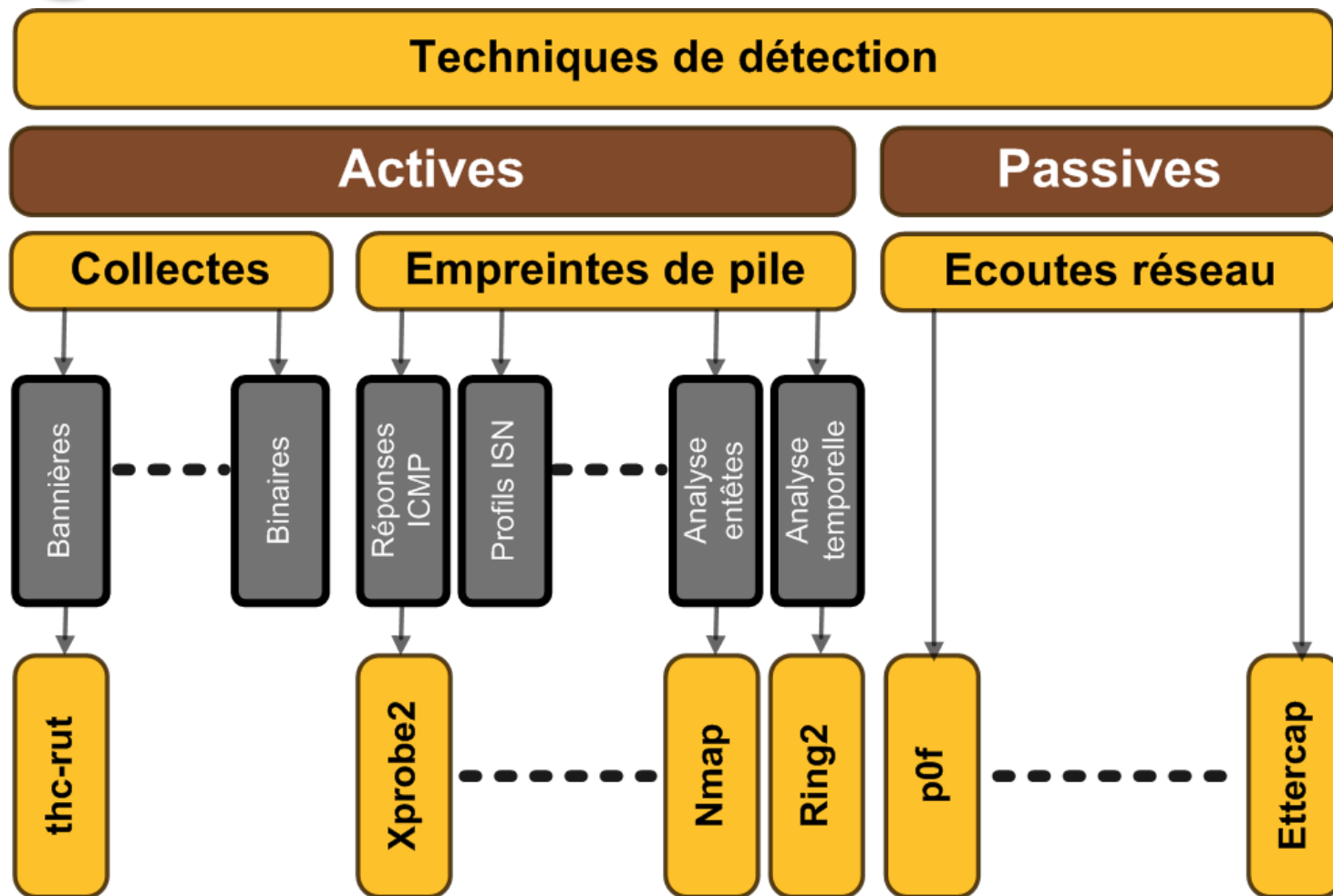
– Apparentés étymologiques

1. FpFucker
2. OS fingerprint frustrating



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Techniques de détection [11]





IpMorph : « vers la mystification de la prise d'empreinte de pile »

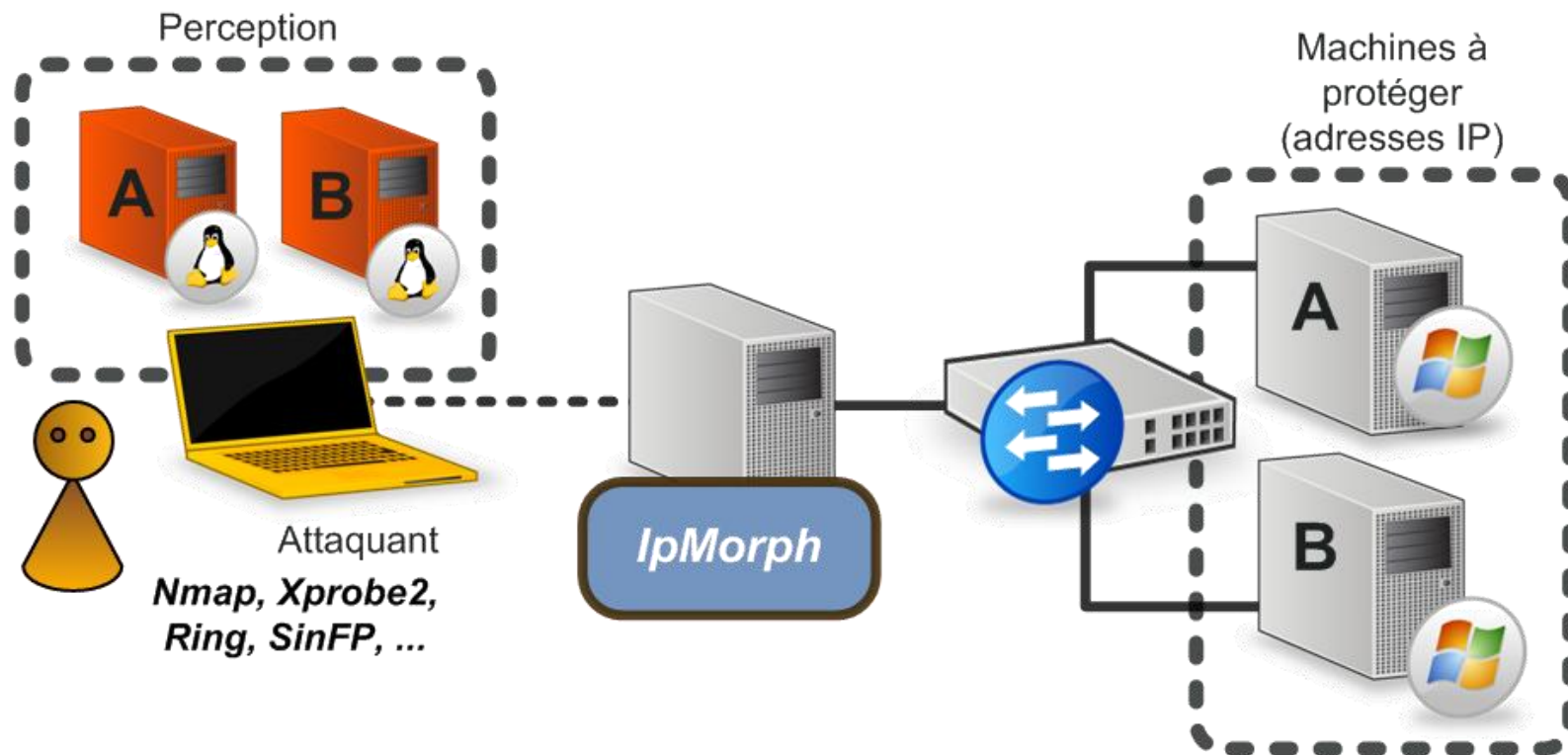
Etat de l'art de la mystification [7]

- **Filtrage**
 - Stealth patch : **Unmaintained as of 2002, GNU/Linux kernel 2.2-2.4 [14]**
 - Blackhole : **FreeBSD, kernel options [16]**
 - IPlog : **Unmaintained as of 2001, *BSD [17]**
 - Packet filter : **OpenBSD [18]**
- **Configuration et modification de pile TCP/IP ("host based")**
 - Ip Personality [19]
 - Fingerprint Fucker [12][13]
 - Fingerprint scrubber [1]
 - OSfuscate [8]
- **Substitution de pile TCP/IP ("proxy behaviour")**
 - Honeyd [9]
 - Packet purgatory / Morph [10]



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Utilisation d'IpMorph 1/2

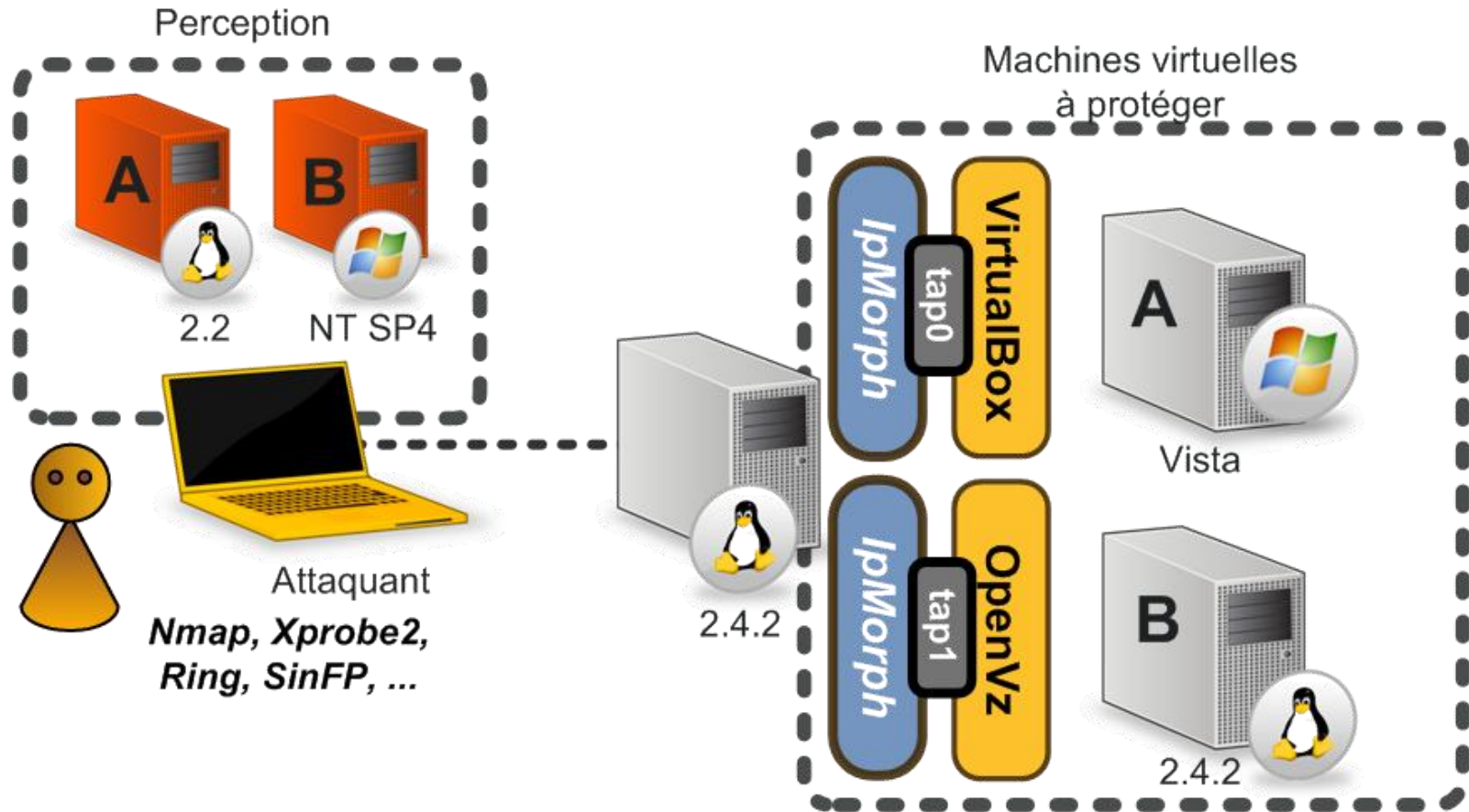


IpMorph en coupure de machines réelles



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Utilisation d'IpMorph 2/2

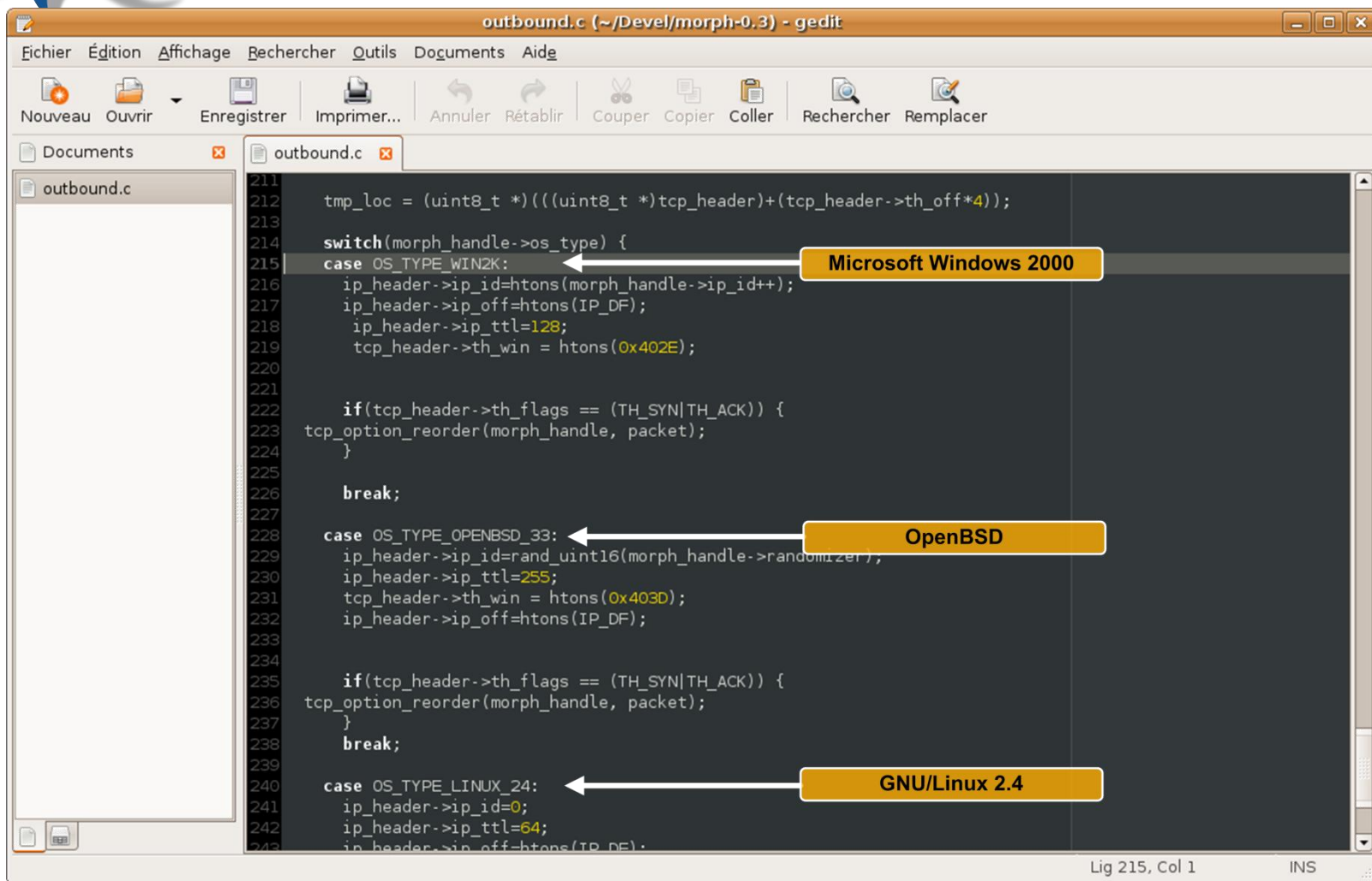


IpMorph en relai protection de machines virtuelles



IpMorph : « vers la mystification de la prise d'empreinte de pile »

IpMorph != Packet Purgatory / Morph [20]



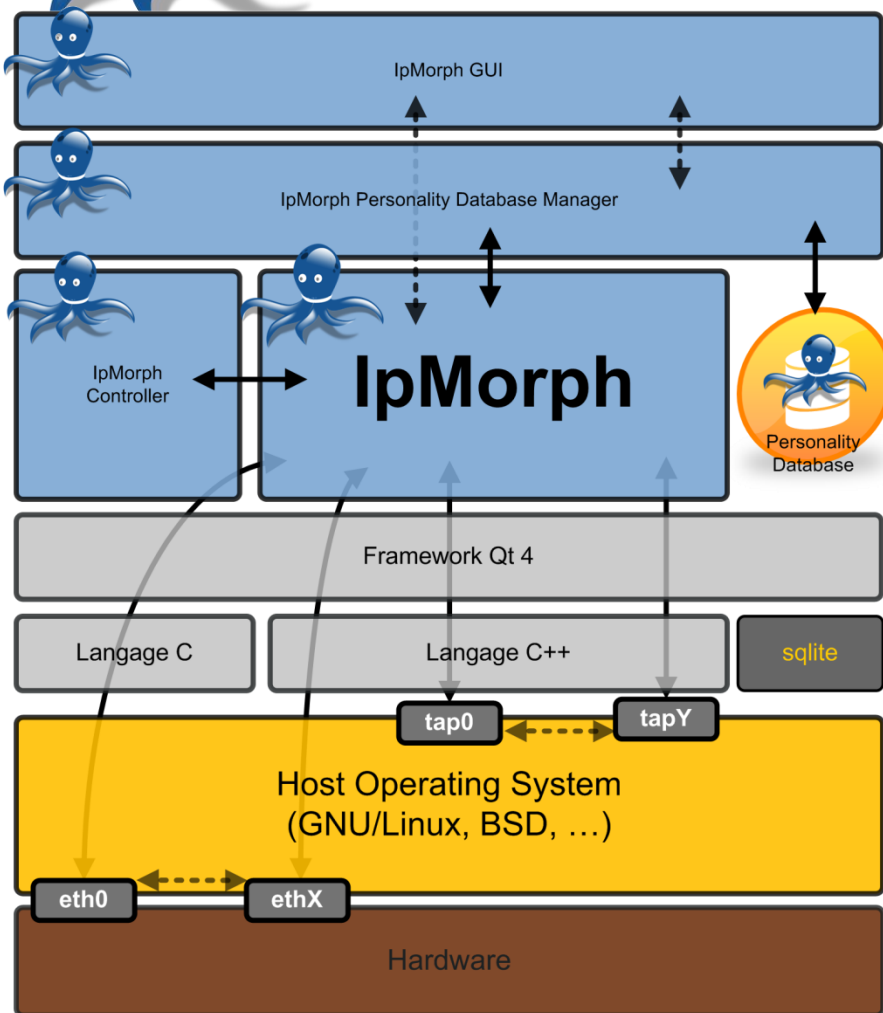
```
outbound.c (~/Devel/morph-0.3) - gedit
Fichier  Édition  Affichage  Rechercher  Outils  Documents  Aide
Nouveau  Ouvrir  Enregistrer  Imprimer...  Annuler  Rétablir  Couper  Copier  Coller  Rechercher  Remplacer
Documents
outbound.c
outbound.c
211 tmp_loc = (uint8_t *)(((uint8_t *)tcp_header)+(tcp_header->th_off*4));
212
213
214 switch(morph_handle->os_type) {
215 case OS_TYPE_WIN2K: ← Microsoft Windows 2000
216     ip_header->ip_id=htons(morph_handle->ip_id++);
217     ip_header->ip_off=htons(IP_DF);
218     ip_header->ip_ttl=128;
219     tcp_header->th_win = htons(0x402E);
220
221
222     if(tcp_header->th_flags == (TH_SYN|TH_ACK)) {
223 tcp_option_reorder(morph_handle, packet);
224     }
225
226     break;
227
228 case OS_TYPE_OPENBSD_33: ← OpenBSD
229     ip_header->ip_id=rand_uint16(morph_handle->randomizer);
230     ip_header->ip_ttl=255;
231     tcp_header->th_win = htons(0x403D);
232     ip_header->ip_off=htons(IP_DF);
233
234
235     if(tcp_header->th_flags == (TH_SYN|TH_ACK)) {
236 tcp_option_reorder(morph_handle, packet);
237     }
238     break;
239
240 case OS_TYPE_LINUX_24: ← GNU/Linux 2.4
241     ip_header->ip_id=0;
242     ip_header->ip_ttl=64;
243     ip_header->ip_off=htons(IP_DF);
```

Lig 215, Col 1 INS



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Socle logiciel

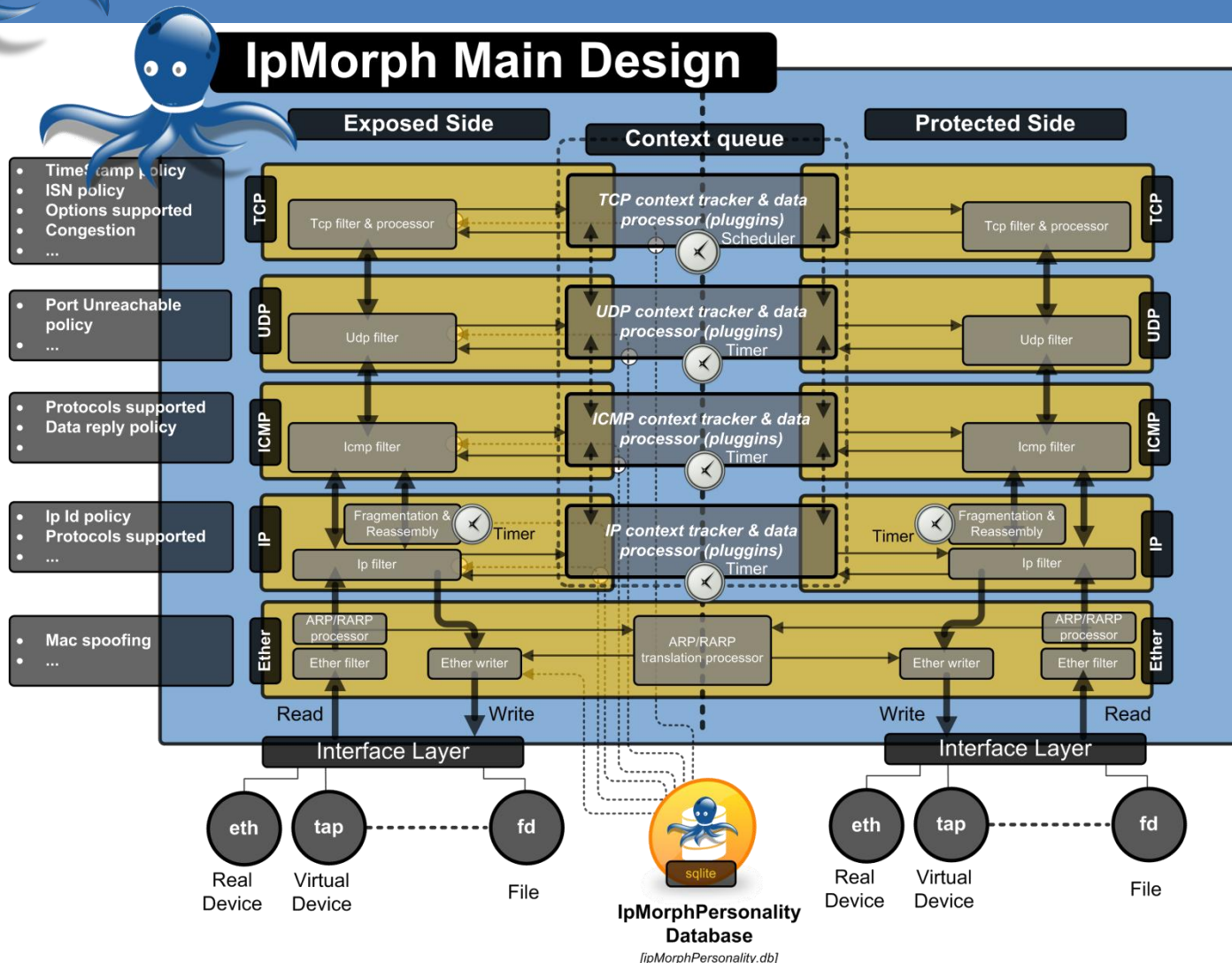


- **Langage C++**
- **Application « UserLand »**
- **Framework Qt4**
- **Éléments constituant :**
 - IpMorph (Core)
 - IpMorph Controller
 - IpMorph Personality Manager
 - IpMorph Personality Database
 - IpView (IpMorph GUI)
- **Portabilité :**
 - GNU/Linux
 - *BSD, Mac OS
- **License GPLv3**



IpMorph : « vers la mystification de la prise d'empreinte de pile »

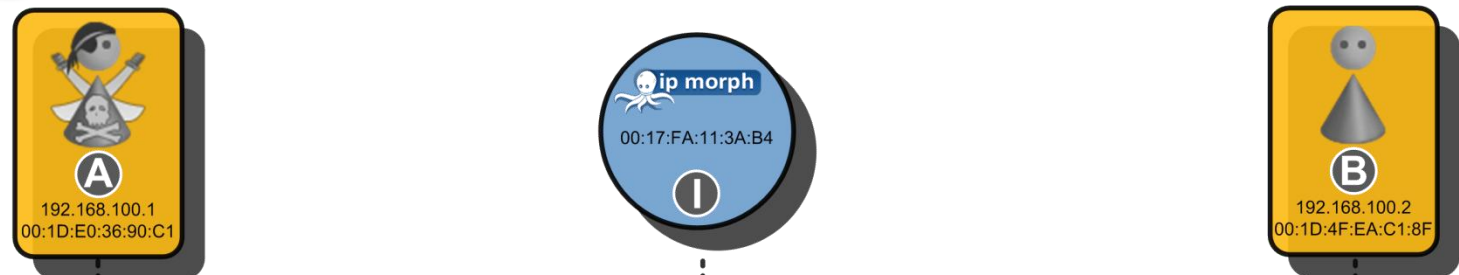
Architecture générale



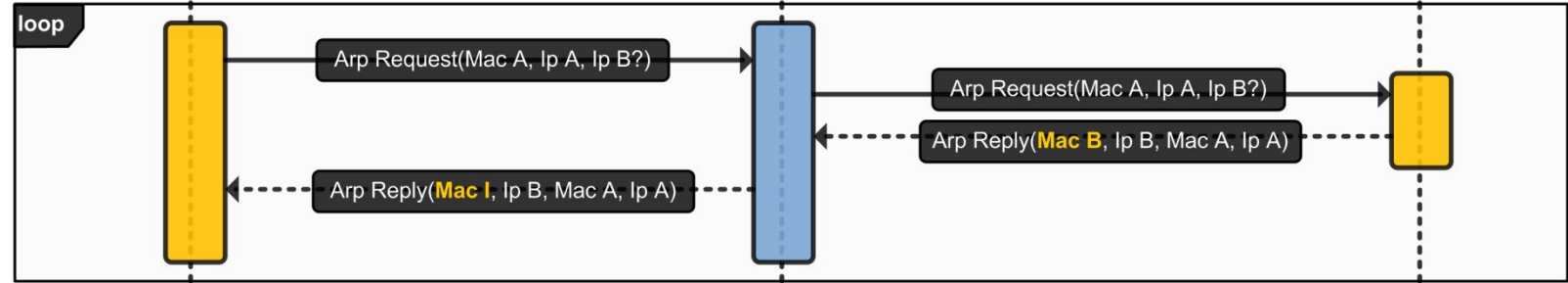


IpMorph : « vers la mystification de la prise d'empreinte de pile »

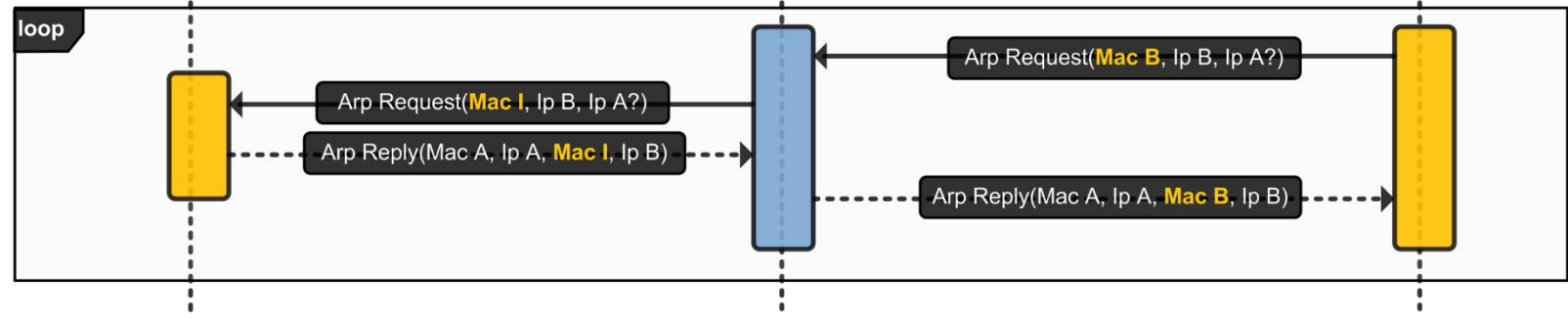
Translation ARP



[translation de résolution en entrée]



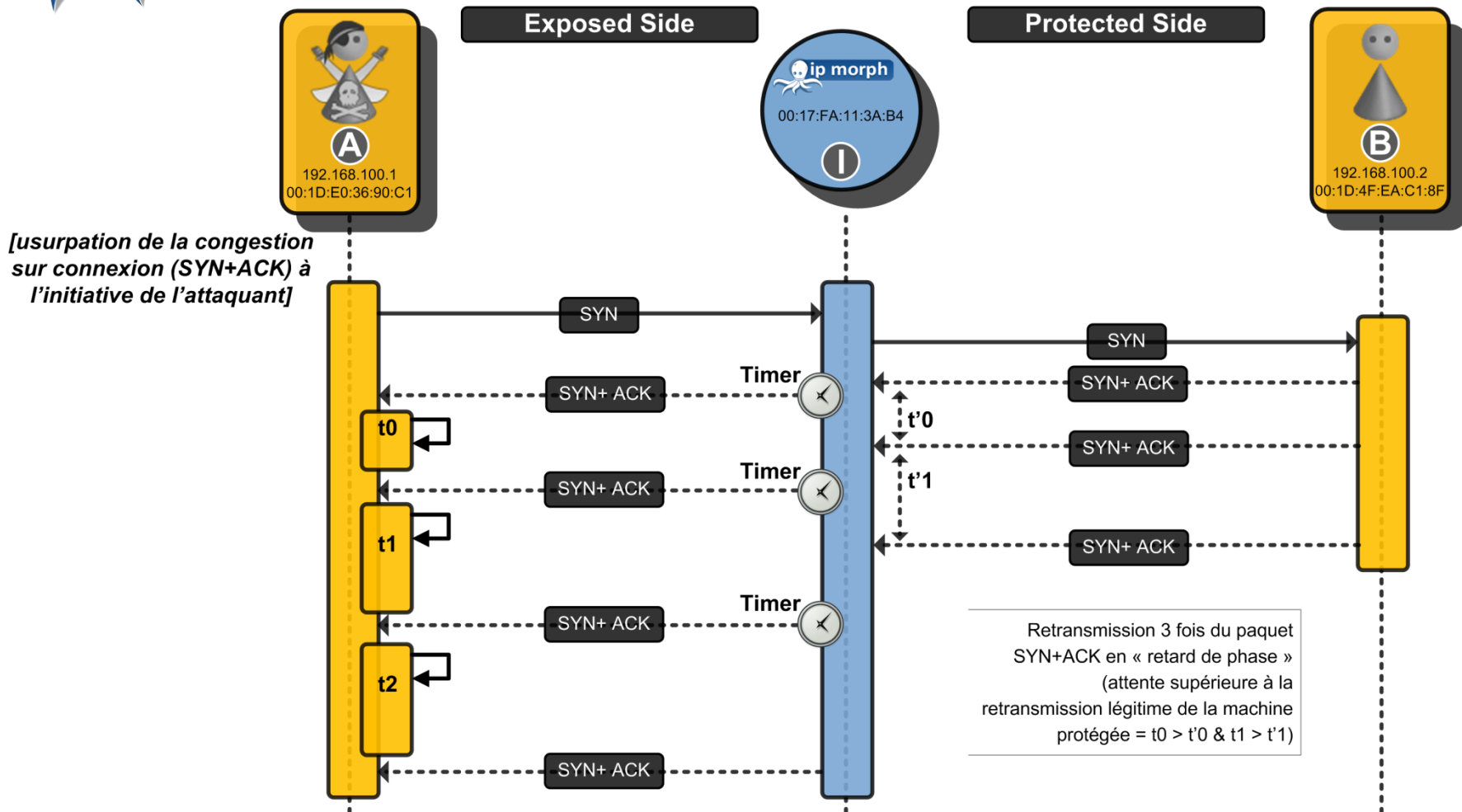
[translation de résolution en sortie]





IpMorph : « vers la mystification de la prise d'empreinte de pile »

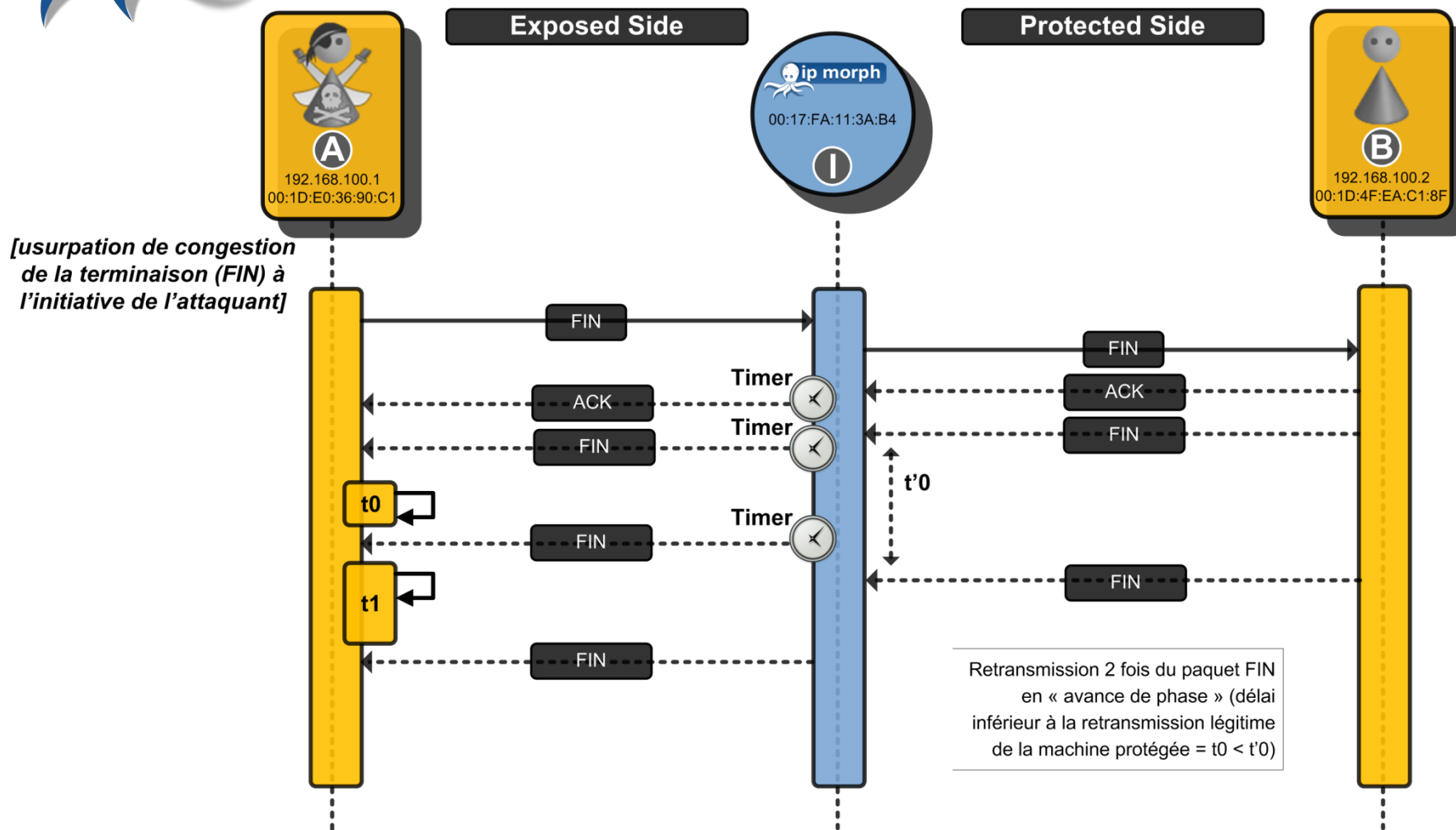
Mystification de la congestion - SYN+ACK





IpMorph : « vers la mystification de la prise d'empreinte de pile »

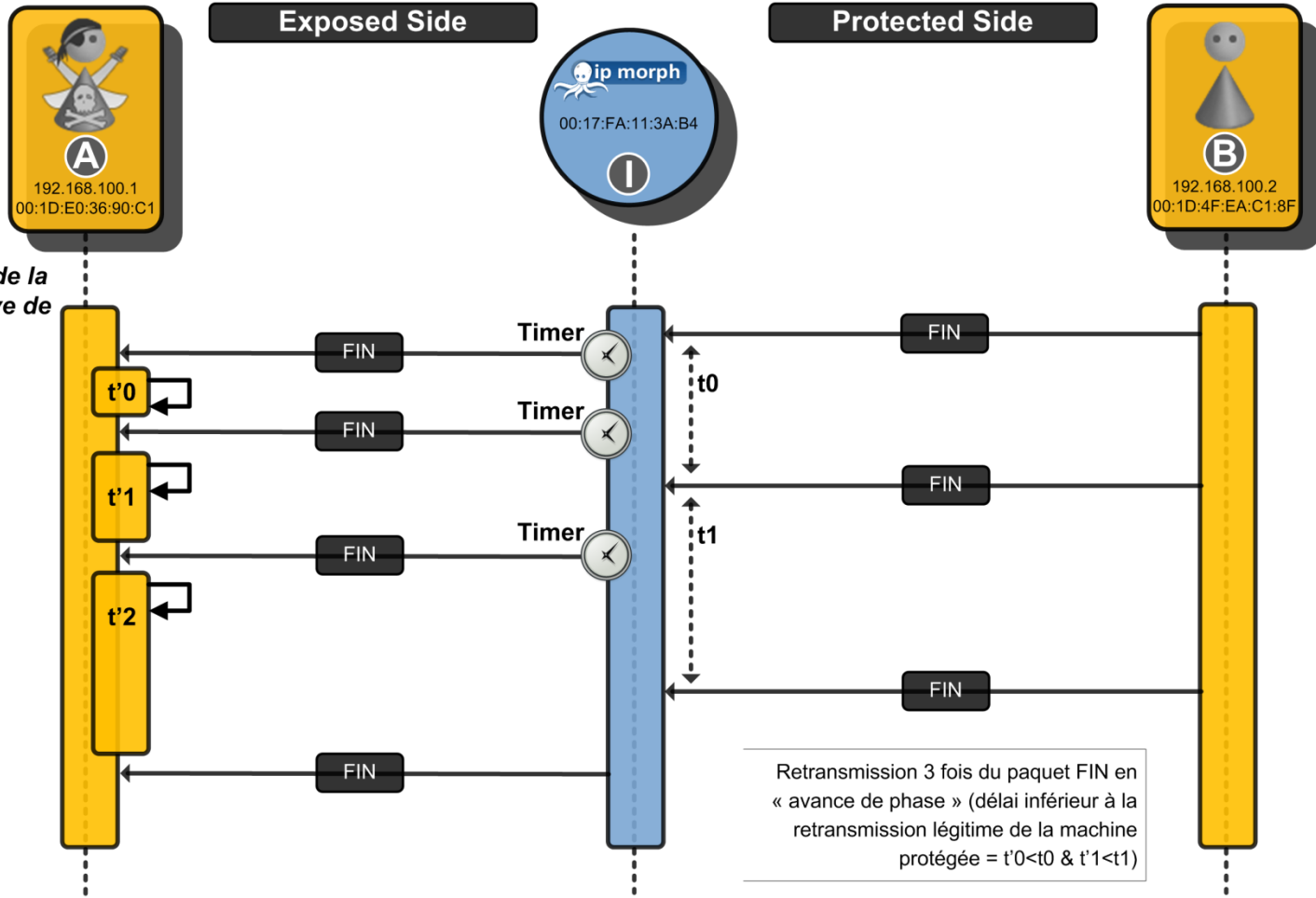
Mystification de la congestion - FIN+ACK « client »





IpMorph : « vers la mystification de la prise d'empreinte de pile »

Mystification de la congestion - FIN+ACK « serveur »



[usurpation de congestion de la terminaison (FIN) à l'initiative de la machine protégée]



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Focus : Calcul ISN

```

ipMorphTcpContext.cpp (~) - gedit
Fichier Édition Affichage Rechercher Outils Documents Aide
Nouveau Ouvrir Enregistrer Imprimer... Annuler Rétablir Couper Copier Coller Rechercher Remplacer

ipMorphTcpContext.cpp
1209 if(_NOS.getTcpIsnMode() == NmapOS::ISN_TR) // True Random
1210 {
1211     _isn = rand();
1212 }
1213 else if(_NOS.getTcpIsnMode() == NmapOS::ISN_RI) // Random Increment
1214 {
1215     // méthode de Box-Muller
1216     // r et phi aléatoires entre 0 et 1 : (0 1]
1217     // calcul des nombres de la loi normale :
1218     // z0 = cos(2*PI*phi) * sqrt(-2*ln(r))
1219     // z1 = sin(2*PI*phi) * sqrt(-2*ln(r))
1220     // et on obtient nos 2 nombres avec le bon écart-type autour de la moyenne :
1221     // n0 = moyenne + écart-type * z0
1222     // n1 = moyenne + écart-type * z1
1223     // note : en C, ln() correspond à la fonction "log()" de math.h
1224
1225     unsigned long ecartType = _NOS.getTcpIsnParm();
1226     unsigned long moyenne = 2*ecartType;
1227
1228     srand(_getTime() + _isn); // réinitialisation de la génération des nombres aléatoires
1229     double r = ((double)1*(1+rand()/(RAND_MAX+1.0)));
1230
1231     srand(_getTime() + (int)(_isn * (1+r))); // réinitialisation de la génération des nombres aléatoires
1232     double phi = ((double)1*(1+rand()/(RAND_MAX+1.0)));
1233
1234     double z0 = cos(2*M_PI*phi) * sqrt(-2*Log(r));
1235     double n0 = moyenne + ecartType * z0;
1236
1237     _isn = (Long)n0;
1238 }
1239 else if(_NOS.getTcpIsnMode() == NmapOS::ISN_TD) // Time Dependant
1240 {
1241     unsigned int gcd = _NOS.getTcpIsnParm();
1242     if(gcd!=0)
1243     {
1244         _isn+=gcd;
1245     }
1246     else
1247     {
1248         unsigned int avg = 10;
1249         _isn+=avg+lrnt((double)avg*rand()/(RAND_MAX+1.0));
1250     }
1251 }
1252 else if(_NOS.getTcpIsnMode() == NmapOS::ISN_64K) // 64K
1253 {
1254     _isn+=64000;
1255 }
1256 else if( _NOS.getTcpIsnMode() == NmapOS::ISN_i800) // i800
  
```

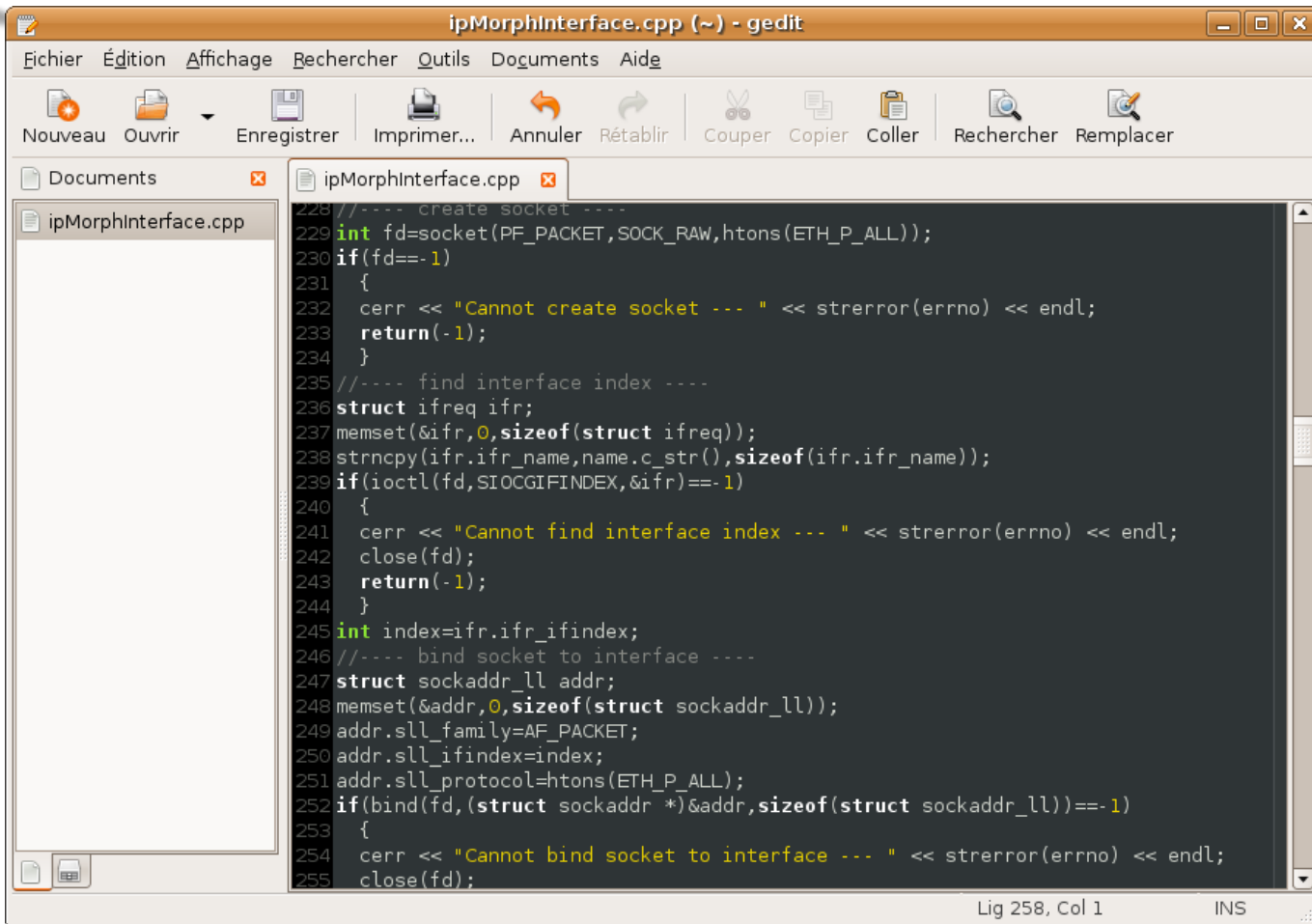
Lig 1235, Col 1 INS

Génération de l'ISN pour l'ancienne version de Nmap (<4.11)



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Focus : Interfaces réseaux « read/write »



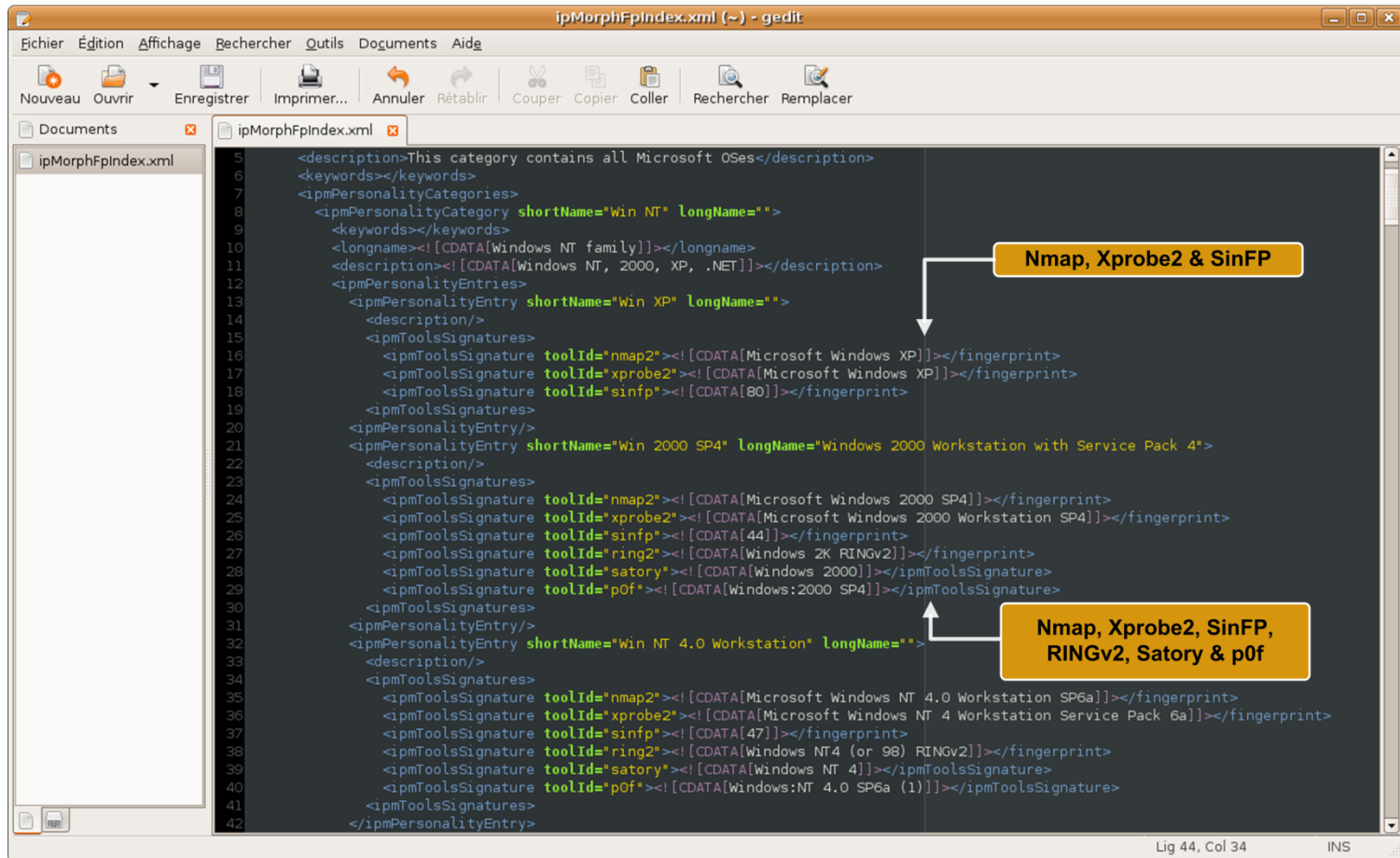
```
ipMorphInterface.cpp (~) - gedit
Fichier Édition Affichage Rechercher Outils Documents Aide
Nouveau Ouvrir Enregistrer Imprimer... Annuler Rétablir Couper Copier Coller Rechercher Remplacer
Documents
ipMorphInterface.cpp
ipMorphInterface.cpp
228 //---- create socket ----
229 int fd=socket(PF_PACKET,SOCK_RAW,htons(ETH_P_ALL));
230 if(fd==-1)
231 {
232     cerr << "Cannot create socket --- " << strerror(errno) << endl;
233     return(-1);
234 }
235 //---- find interface index ----
236 struct ifreq ifr;
237 memset(&ifr,0,sizeof(struct ifreq));
238 strncpy(ifr.ifr_name,name.c_str(),sizeof(ifr.ifr_name));
239 if(ioctl(fd,SIOCGIFINDEX,&ifr)==-1)
240 {
241     cerr << "Cannot find interface index --- " << strerror(errno) << endl;
242     close(fd);
243     return(-1);
244 }
245 int index=ifr.ifr_ifindex;
246 //---- bind socket to interface ----
247 struct sockaddr_ll addr;
248 memset(&addr,0,sizeof(struct sockaddr_ll));
249 addr.sll_family=AF_PACKET;
250 addr.sll_ifindex=index;
251 addr.sll_protocol=htons(ETH_P_ALL);
252 if(bind(fd,(struct sockaddr *)&addr,sizeof(struct sockaddr_ll))== -1)
253 {
254     cerr << "Cannot bind socket to interface --- " << strerror(errno) << endl;
255     close(fd);
```

Ouvertures des interfaces (eth, tap) pour capture et injection



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Personality Manager : Structure de l'index



```

5 <description>This category contains all Microsoft OSes</description>
6 <keywords></keywords>
7 <ipmPersonalityCategories>
8   <ipmPersonalityCategory shortName="Win NT" longName="">
9     <keywords></keywords>
10    <longname><![CDATA[Windows NT family]]></longname>
11    <description><![CDATA[Windows NT, 2000, XP, .NET]]></description>
12    <ipmPersonalityEntries>
13      <ipmPersonalityEntry shortName="win XP" longName="">
14        <description/>
15        <ipmToolsSignatures>
16          <ipmToolsSignature toolId="nmap2"><![CDATA[Microsoft Windows XP]]></fingerprint>
17          <ipmToolsSignature toolId="xprobe2"><![CDATA[Microsoft Windows XP]]></fingerprint>
18          <ipmToolsSignature toolId="sinfp"><![CDATA[80]]></fingerprint>
19        </ipmToolsSignatures>
20      </ipmPersonalityEntry>
21      <ipmPersonalityEntry shortName="win 2000 SP4" longName="Windows 2000 Workstation with Service Pack 4">
22        <description/>
23        <ipmToolsSignatures>
24          <ipmToolsSignature toolId="nmap2"><![CDATA[Microsoft Windows 2000 SP4]]></fingerprint>
25          <ipmToolsSignature toolId="xprobe2"><![CDATA[Microsoft Windows 2000 Workstation SP4]]></fingerprint>
26          <ipmToolsSignature toolId="sinfp"><![CDATA[44]]></fingerprint>
27          <ipmToolsSignature toolId="ring2"><![CDATA[Windows 2K RINGv2]]></fingerprint>
28          <ipmToolsSignature toolId="satory"><![CDATA[Windows 2000]]></ipmToolsSignature>
29          <ipmToolsSignature toolId="p0f"><![CDATA[Windows:2000 SP4]]></ipmToolsSignature>
30        </ipmToolsSignatures>
31      </ipmPersonalityEntry>
32      <ipmPersonalityEntry shortName="win NT 4.0 Workstation" longName="">
33        <description/>
34        <ipmToolsSignatures>
35          <ipmToolsSignature toolId="nmap2"><![CDATA[Microsoft Windows NT 4.0 Workstation SP6a]]></fingerprint>
36          <ipmToolsSignature toolId="xprobe2"><![CDATA[Microsoft Windows NT 4 Workstation Service Pack 6a]]></fingerprint>
37          <ipmToolsSignature toolId="sinfp"><![CDATA[47]]></fingerprint>
38          <ipmToolsSignature toolId="ring2"><![CDATA[Windows NT4 (or 98) RINGv2]]></fingerprint>
39          <ipmToolsSignature toolId="satory"><![CDATA[Windows NT 4]]></ipmToolsSignature>
40          <ipmToolsSignature toolId="p0f"><![CDATA[Windows:NT 4.0 SP6a (1)]]></ipmToolsSignature>
41        </ipmToolsSignatures>
42      </ipmPersonalityEntry>

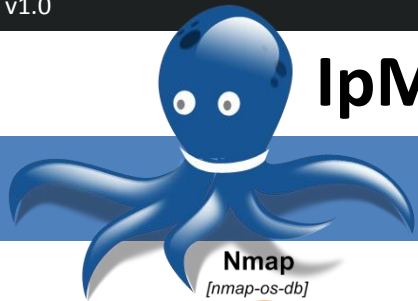
```

Nmap, Xprobe2 & SinFP

Nmap, Xprobe2, SinFP, RINGv2, Satory & p0f

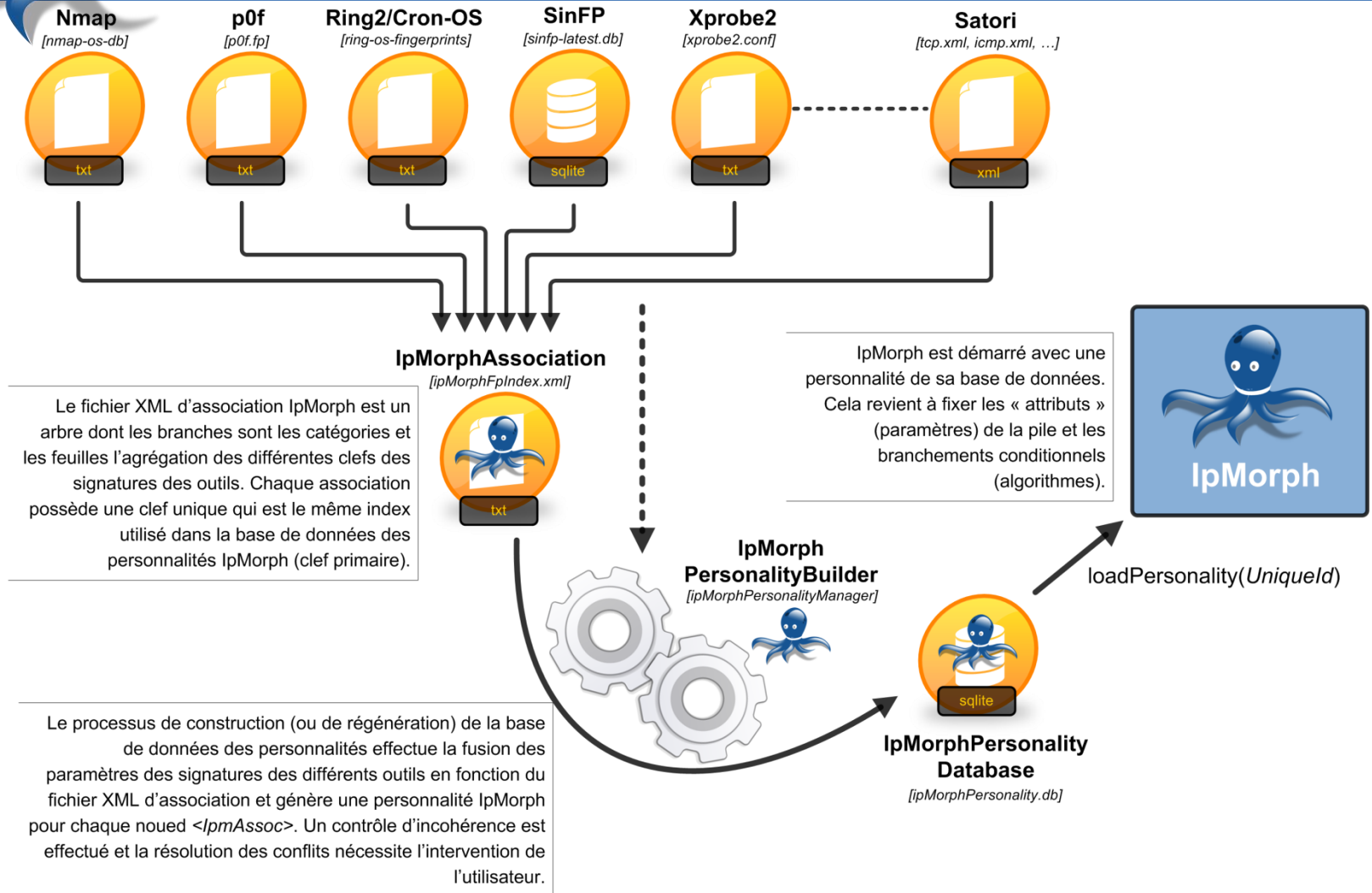
Lig 44, Col 34 INS

Formalisation XML et association des signatures des outils d'OSFP



IpMorph : « vers la mystification de la prise d'empreinte de pile »

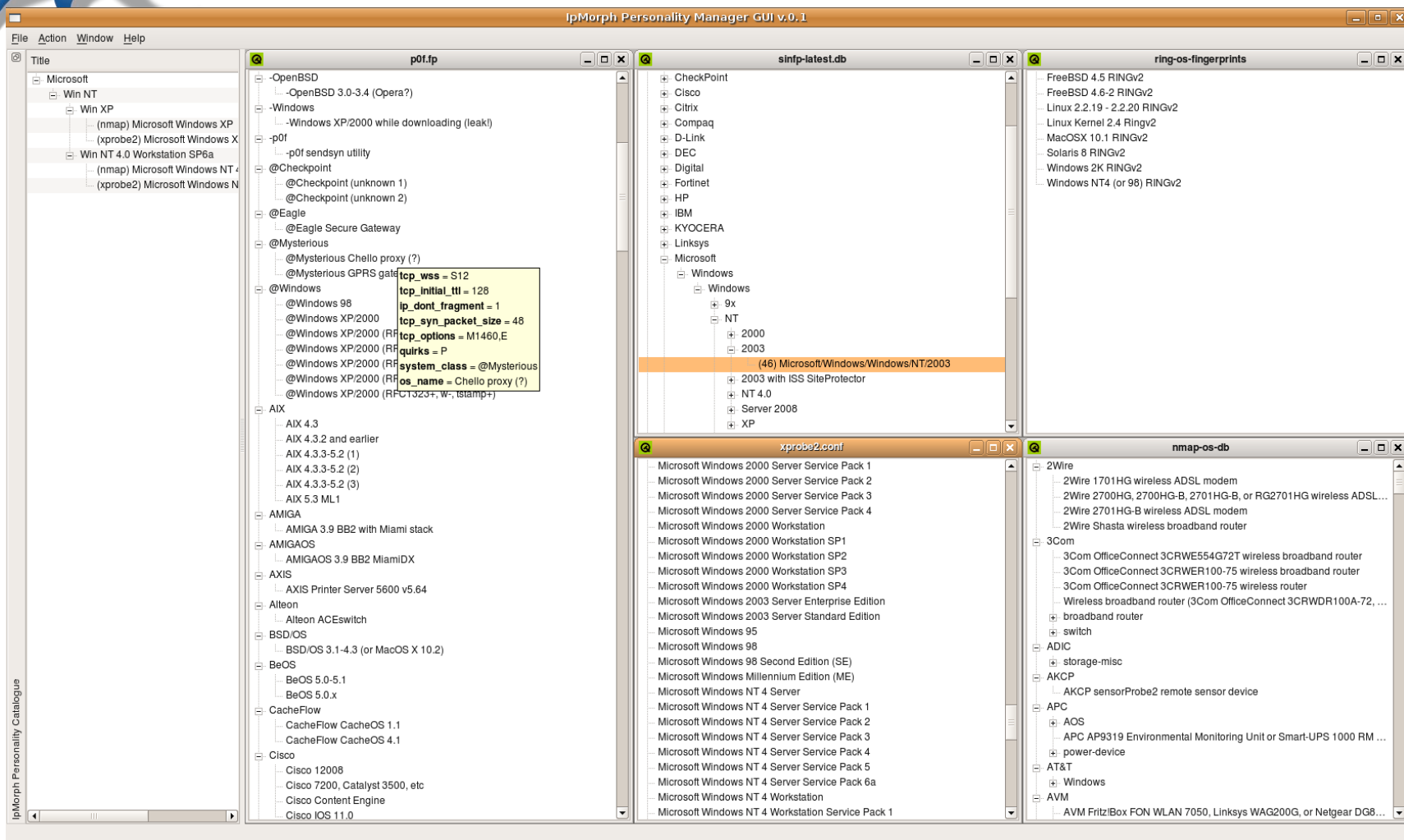
Personality Manager : Présentation





IpMorph : « vers la mystification de la prise d'empreinte de pile »

Personality Manager : Démonstration



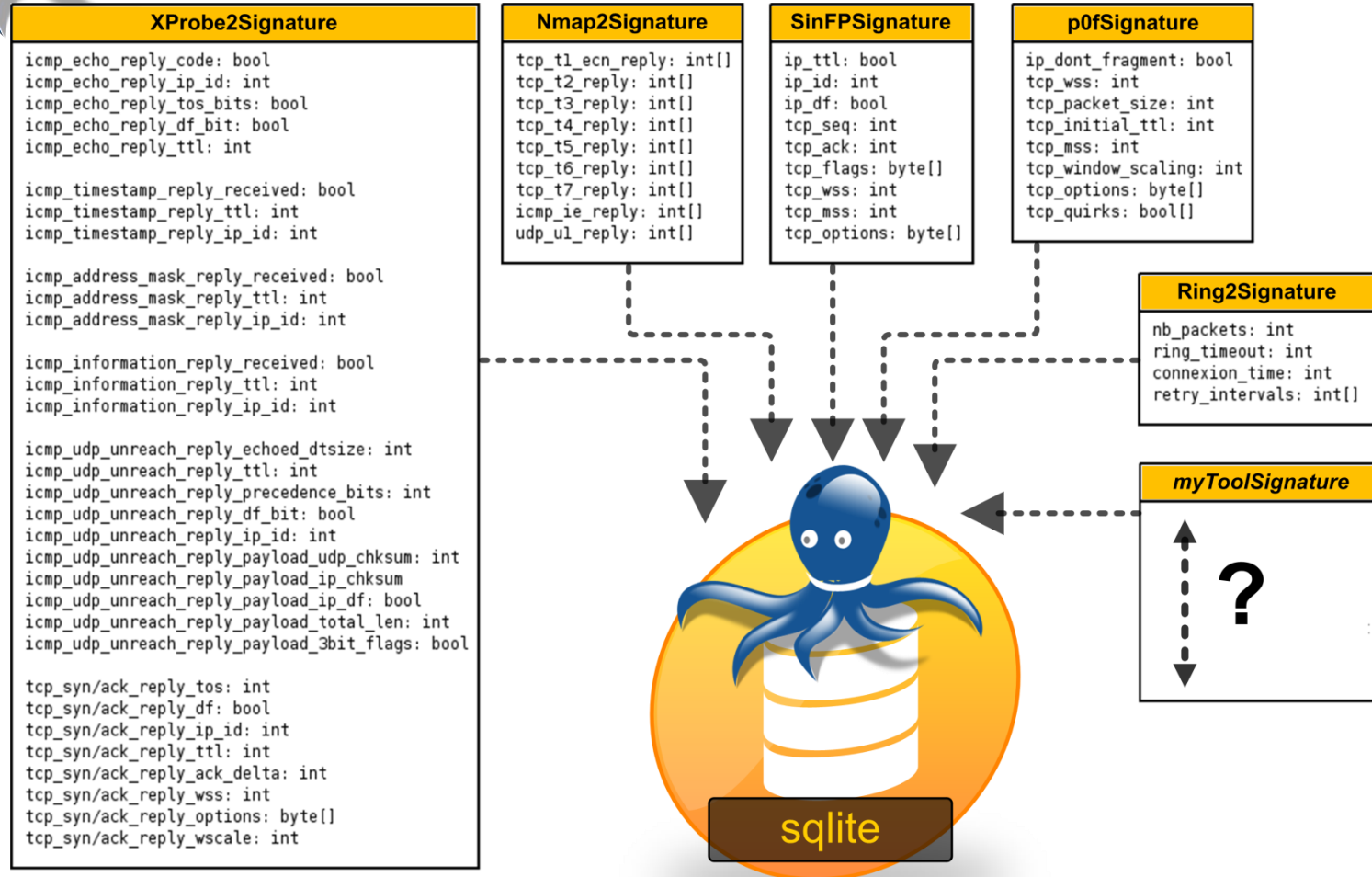
The screenshot displays the IpMorph Personality Manager GUI v.0.1. The interface is divided into several panes:

- Left Pane (Title):** A tree view showing the hierarchy of personalities. The selected path is: Microsoft > Win NT > Win XP > (nmap) Microsoft Windows XP (xprobe2) Microsoft Windows X.
- Top Middle Pane (p0f.fp):** A list of personalities with their associated fingerprints. A yellow highlight is on the entry for '@Windows XP/2000 (Rf)'. The highlighted entry shows:
 - tcp_wss = S12
 - tcp_initial_ttl = 128
 - ip_dont_fragment = 1
 - tcp_syn_packet_size = 48
 - tcp_options = M1460,E
 - quirks = P
 - system_class = @Mysterious
 - os_name = Chello proxy (?)
- Top Right Pane (sinfp-latest.db):** A tree view of fingerprints. The selected entry is '(46) Microsoft/Windows/WindowsNT/2003'.
- Bottom Middle Pane (xprobe2.conf):** A list of personalities with their associated fingerprints. The selected entry is 'Microsoft Windows 2000 Server Service Pack 1'.
- Bottom Right Pane (nmap-os-db):** A tree view of fingerprints. The selected entry is '2Wire 2700HG, 2700HG-B, 2701HG-B, or RG2701HG wireless ADSL...'.



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Personality Manager : Database Overview



IpMorphPersonalityDatabase

[ipMorphPersonality.db]



IpMorph : « vers la mystification de la prise d'empreinte de pile »

IpMorph : Illustration pratique

DEMONSTRATION



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Etat d'avancement

- **Conception = 70%**
 - Etat de l'art bibliographique = 85%
 - Etat de l'art technique mystification = 70%
 - Analyse des outils d'OSFP = 70%
 - *Nmap* = 80%
 - *Xprobe2* = 95%
 - *SinFP* = 60%
 - *Satori* = 45%
 - *Ring2* = 80%
 - Architecture générale = 80%
 - Spécification & UML/Software design = 50%
- **Implémentation = 10%**
 - Proof of concept « trash programming » = 85%
 - Développement des composants = 5% (50% pour Personality Manager)
 - *Tests unitaires, tests fonctionnels, documentation* = 5%



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Perspectives

- **Juin 2009 – SSTIC 2009**
 - Beta release 0.5 en download
 - Site web, Mailing list, Bugtracking
 - Subversion / sources
- **Décembre 2009**
 - Version 1.0 en download
 - Documentation, UserGuide, DevGuide
 - Distribution .deb, .rpm, ...
 - Intégration de quelques scrubbers applicatifs (DNS, SMB, DHCP, ...).



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Bibliographie 1/2

1. **Smart, M., Malan, G.R., Jahanian, F.:** **Defeating TCP/IP Stack Fingerprinting**, 9th USENIX Security Symp.
2. **Smith, C., Grundl, P.:** **Know Your Enemy: Passive Fingerprinting**, <http://old.honeynet.org/papers/finger/>, (2002)
3. **Fyodor :** **Remote OS detection via TCP/IP Stack FingerPrinting**, Phrack 1998, <http://www.insecure.org/nmap/nmap-fingerprinting-article.txt>, (1998)
4. **Spangler, R.:** **Analysis of Remote Active Operating System Fingerprinting Tools**, <http://www.packetwatch.net/documents/papers/osdetection.pdf>
5. **Veysset, F., Courtay, O., Heen, O.:** **New Tool And Technique For Remote Operating System Fingerprinting**, http://www.hackerz.ir/e-books/remote_os_detection.pdf, (2002)
6. **Auffret, P. :** **SinFP, Unification de la prise d'empreinte passive et active des systèmes d'exploitation**, http://actes.sstic.org/SSTIC08/SinFP_Unification_Prise_Empreinte_Active_Passive_Systemes_Exploitation/, SSTIC 08, (2008)
7. **Berrueta, D.B.:** **A practical approach for defeating Nmap OS-Fingerprinting**, <http://nmap.org/misc/defeat-nmap-osdetect.html> (2003)
8. **Crenshaw, A. :** **OSfuscate: Change your Windows OS TCP/IP Fingerprint to confuse P0f, NetworkMiner, Ettercap, Nmap and other OS detection tools**, <http://www.irongeek.com/i.php?page=security/osfuscate-change-your-windows-os-tcp-ip-fingerprint-to-confuse-p0f-networkminer-ettercap-nmap-and-other-os-detection-tools>, (2008)
9. **Provos, N. :** **Honeyd: A Virtual Honeypot Daemon**, <http://www.citi.umich.edu/u/provos/papers/honeyd-eabstract.pdf> , (2003)
10. **Wang, K. :** **Frustrating OS Fingerprinting with Morph**, <http://www.synacklabs.net/projects/morph/Wang-Morph-TheFifthHOPE.pdf>, The Fifth HOPE, (2004)
11. **Veysset, F., Courtay, O., Heen, O.:** **Détection des systèmes d'exploitation avec RINGv2**, Actes SSTIC, (2003)
12. **Fusys, CyRaX :** **Fingerprint Fucker**, <http://www.s0ftpj.org/tools/fingfuck.tgz>
13. **Reed, D. :** **Fingerprint Fucker**, <http://packetstormsecurity.org/UNIX/misc/bsdspf.tar.gz>
14. **Trifero, S., Callaway, D. :** **Linux Kernel Stealth Patch**, <http://www.innu.org/~sean/>



IpMorph : « vers la mystification de la prise d'empreinte de pile »

Bibliographie 2/2

15. **Aubert, S.** : **Antimap**, <http://www.hsc.fr/ressources/breves/antimap.html>
16. **FreeBSD, Blackhole**, <http://www.gsp.com/cgi-bin/man.cgi?section=4&topic=blackhole>
17. **McCabe, R.** : **Iplog**, <http://ojnk.sourceforge.net/stuff/iplog.readme>
18. **OpenBSD** : **PF « le filtrage de paquet »**, <http://www.openbsd.org/faq/pf/fr/filter.html>
19. **Roulland, G., Saffroy, J-M.** : **IP Personality**, <http://ippersonality.sourceforge.net/>
20. **Wang, K.** : **Frustrating OS Fingerprinting with Morph**, DEFCON 12, <http://www.synacklabs.net/projects/morph/Wang-Morph-DEFCON12.pdf> (2004)

Merci de votre attention.

Interrogations ? Critiques !

Suggestions...



Nous contacter :
contact@ipmorph.org
Beta Release 2009-06

<http://dev.ipmorph.org>