

# Des correctifs de sécurité à la mise à jour

*Audit, déploiement distribué et injection à chaud*

Nicolas Lorient\*, Marc Ségura-Devillechaise, Jean-Marc  
Menaud  
EMN Obasco/INRIA

DECOR – 28 octobre 2004

# Tendance des nouveaux vers

- 2003, 80% des attaques informatiques subies le sont à cause d'un manque de suivi des mises à jour. [DEV]
- SASSER
  - développement du virus sur la base du bulletin d'alerte publié par Microsoft
  - virus déployé 15 jours **après** la mise à disposition du correctif

# Problématique de la mise à jour

- Suivi des alertes de sécurité [CERT]
  - Lecture des alertes
  - 5500 par an
  - 5 minutes par alerte
  - Total : **13 semaines** de travail
- Administration d'un parc informatique
  - Supposons :
    - 100 machines/administrateur (EMN : 150)
    - affectées par 1% des alertes (55 alertes/an)
  - 1 heure par mise à jour
  - Total lecture + mise à jour : **157 semaines** de travail

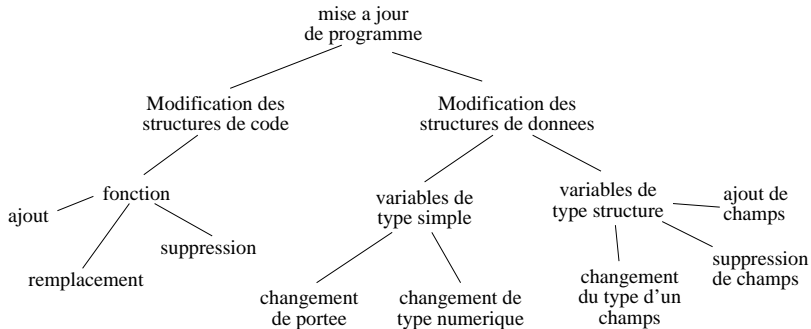
# Contribution

- Un atelier de déploiement de mises à jour de sécurité pour logiciels C, open-source :
  - déploiement semi-automatique et immédiat sur l'ensemble d'un parc informatique
  - sans arrêt ou perte des sessions/travaux en cours
  - intégration dans le processus standard de mise à jour
  - facilitation de l'audit et de la validation des correctifs
- deux outils : Minerve & Arachne

# Minerve

- Transformation automatique d'un correctif source en un correctif dynamique :
  - stratégie de transformation : granularité fonction
  - vérification de l'applicabilité dynamique du correctif
    - ex :  $s = s + 1$ ;  $\rightarrow s = s + 2$ ;
    - résultat dépendant de l'état du programme au moment de la mise à jour.
- étude de l'ensemble des modifications possibles

# Étude



# Minerve

- facilitation de l'audit : langage d'aspect décrivant les modifications

```
@@ -256,6 +256,8 @@  
+ if (nresp > 100)  
+ fatal("nresp too big %u", nresp);
```

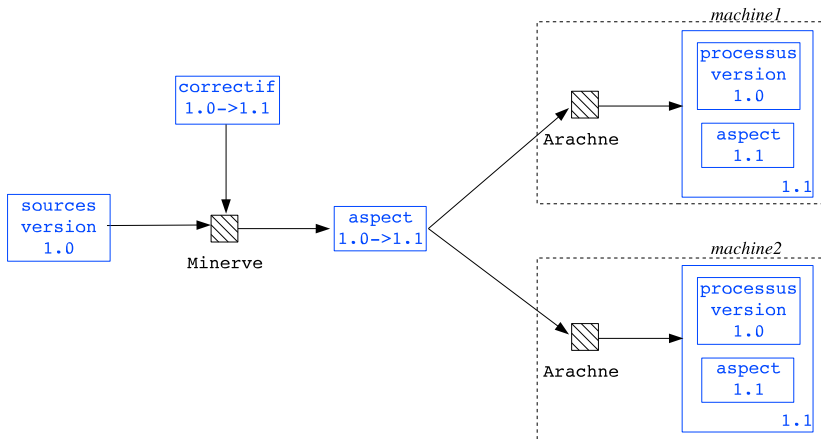
---

```
ReplaceFunctionCall [:  
    void userauth_info_response(int type, u_int32_t seq,  
        void* ctxt) [:  
        userauth_info_response_new(type, seq, ctxt);}  
:]  
:]
```

# Arachne

- Tisseur d'aspects pour programme C
  - réécriture du binaire à la volée
  - tissage fiable
- Injecte les aspects dans les applications
  - atomiquement
  - sans interrompre la continuité de service
  - rollback également atomique

# Atelier de déploiement



# Expérimentations

- Tous les failles de sécurité répertoriés en CERT Advisories depuis 2002 sur les logiciels open-source
  - Tous applicables dynamiquement
  - Durée d'une mise à jour  $250\mu s$  (en excluant les temps de diffusion sur le réseau)

# Limitations

- Un coup de chance ?
  - jeu d'essai trop simple ?
  - combinaisons de modifications
  - analyses de code complexes
    - langage C
    - cas général, incomplètes



# Conclusion

- atelier semi-automatique s'insérant dans le processus standard de mise à jour
- pas d'interruption la continuité de service
- diminution du temps d'audit des correctifs
- Travaux futurs :
  - Monitoring distribué et mise à jour statique
  - Affinement des analyses
  - Détecter des patterns de bogues