



Cybersecurity Institute

Univ. Grenoble Alpes

Vulnérabilités Logicielles

Pourquoi ? Comment ?

Laurent Mounier

VERIMAG - PACSS

21 Février 2017





Qu'est-ce qu'une vulnérabilité dans un programme ?

Comment cela fonctionne t-il ?

Pourquoi y en a t-il autant ?

Comment s'en prémunir ?

Quel matériel faut-il pour exécuter un programme ?

Plateforme d'exécution :

ordinateur, téléphone, "objet connecté", automate industriel, voiture, etc.

→ Fournit les **ressources** suivantes :

- une "unité de traitement et de calcul" (processeur, CPU)
- de la mémoire interne
- des interfaces de communications vers/depuis l'extérieur
 - clavier, caméra, micro, capteurs, etc
 - écran, haut-parleur, actionneurs,
 - réseau (Internet, ou autre)
 - mémoires "externe" (disques durs, clés USB, etc.)
 - ...

Ressources partagées entre plusieurs applications/utilisateurs ...

La mémoire interne :

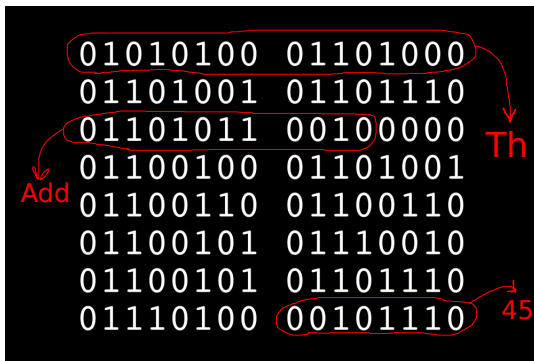
- contient les **instructions** et les **données** du programme
- mode de codage unique : la base 2
un contenu mémoire \Rightarrow **plusieurs interprétations possibles !**
Exemple : un entier codé base 2
 \hookrightarrow une instruction, un entier décimal, du texte , des pixels, ...

La mémoire interne :

- contient les **instructions** et les **données** du programme
- mode de codage unique : la base 2
un contenu mémoire \Rightarrow **plusieurs interprétations possibles !**

Exemple : un entier codé base 2

\hookrightarrow une instruction, un entier décimal, du texte , des pixels, ...

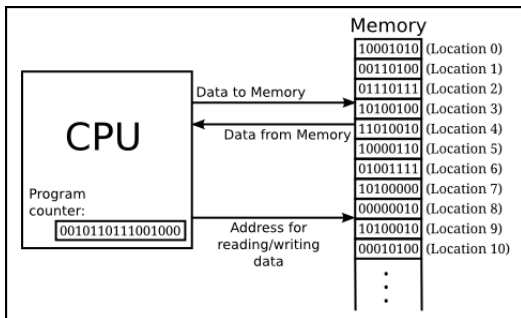


Le processeur/CPU

- 1 lit une instruction en mémoire (ex : additionner 2 entiers)
- 2 exécute cette instruction (avec lecture/écriture en mémoire)
- 3 choisit l'instruction suivante à exécuter ...
...en fonction du contenu courant de la mémoire !

Le processeur/CPU

- 1 lit une instruction en mémoire (ex : additionner 2 entiers)
- 2 exécute cette instruction (avec lecture/écriture en mémoire)
- 3 choisit l'instruction suivante à exécuter ...
... en fonction du contenu courant de la mémoire !



Sécurité = confidentialité, intégrité, disponibilité, . . .

attaques directes sur la **mémoire** = lire et/ou modifier une donnée
(ex : *write-what-where condition*)
→ déni de service, fuite d'information, exécution arbitraire, etc

L'attaquant, ses objectifs, ses moyens . . .

Sécurité = confidentialité, intégrité, disponibilité, . . .

attaques directes sur la **mémoire** = lire et/ou modifier une donnée
(ex : *write-what-where condition*)

→ déni de service, fuite d'information, exécution arbitraire, etc

Différents modèles d'attaquant

- utilisateur "ordinaire" . . . mais malintentionné !
→ utilise les entrées/sorties standards du pgm

L'attaquant, ses objectifs, ses moyens . . .

Sécurité = confidentialité, intégrité, disponibilité, . . .

attaques directes sur la **mémoire** = lire et/ou modifier une donnée
(ex : *write-what-where condition*)

→ déni de service, fuite d'information, exécution arbitraire, etc

Différents modèles d'attaquant

- utilisateur "ordinaire" . . . mais malintentionné !
→ utilise les entrées/sorties standards du pgm
- utilisateur plus "avancé" :
 - observer des **canaux cachés** :
consommation, temps d'exécution, mémoires partagées
 - perturber l'exécution par **injection de fautes**
laser, champ magnétique, etc.

→ l'attaquant peut donc aussi être la **plateforme d'exécution** . . .

Tout part d'un bug !



Tout part d'un **bug** !



... avec pour **conséquences** :

- des exécutions **non prévues** par le programmeur
- des accès **incorrects** à la mémoire

⇒ fuite/altération de données ; exécution arbitraire de code

Exemple classique de vulnérabilité logicielle

Tout part d'un **bug** !



... avec pour **conséquences** :

- des exécutions **non prévues** par le programmeur
- des accès **incorrects** à la mémoire

⇒ fuite/altération de données ; exécution arbitraire de code

Exemple

- 1 lire une entrée x au clavier (ex : age de l'utilisateur)
- 2 allouer un zone mémoire M de taille x
- 3 utiliser M pour écrire des données utilisateur

Bug possible : ne pas vérifier la **pertinence** de x (ex : $1 \leq x \leq 150$)

Si $x < 0$? si x très grand ?

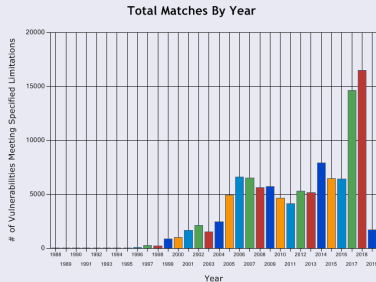
→ accès à M peut permettre une attaque sur la mémoire ...

L'importance du phénomène . . . sur le plan quantitatif

Veracode "State of Software Security Report 2018"

- More than **85%** of all applications have at least one **vulnerability** in them; more than **13%** have at least one **critical** severity flaw
- flaw persistence analysis:
75% after 21 days ; 50% after 121 days, **25% after 472 days**

NIST statistics on software vulnerabilities



L'importance du phénomène . . . sur le plan qualitatif

The screenshot shows the CERT-FR website with a dark blue header. The main content is divided into two sections: 'ALERTE DE SÉCURITÉ' and 'AVIS DE SÉCURITÉ'. Each section contains a list of security incidents with columns for date, ID, description, and status.

ALERTE DE SÉCURITÉ

Date	ID	Description	Status
20 janvier 2019	CSRT19-010-A2-001	Attaque de phishing ciblant l'entreprise Microsoft Exchange et l'adresse Outlook	Alerte en cours
11 janvier 2019	CSRT19-010-A2-002	Empoisonnement de l'infrastructure	Alerte en cours
22 janvier 2019	CSRT19-010-A2-003	Attaque de phishing ciblant l'entreprise SAP	Alerte en cours
26 décembre 2018	CSRT19-010-A2-014	Vulnérabilité dans Microsoft Internet Explorer	Closé le 16/10/2019
26 novembre 2018	CSRT19-010-A2-013	Vulnérabilité dans Oracle Java	Closé le 16/10/2019

100 vulnérabilités

AVIS DE SÉCURITÉ

10 vulnérabilités

Date	ID	Description	Status
17 janvier 2019	CSRT19-010-AF-001	Mitigation vulnérabilité dans Oracle Java	Closé
11 janvier 2019	CSRT19-010-AF-002	Mitigation vulnérabilité dans le protocole Ssh de Oracle	Closé
12 janvier 2019	CSRT19-010-AF-003	Mitigation dans Oracle Java JRE	Closé
11 janvier 2019	CSRT19-010-AF-004	Mitigation vulnérabilité dans Microsoft Exchange	Closé
22 décembre 2018	CSRT19-010-AF-005	Mitigation vulnérabilité dans l'entreprise Microsoft	Closé
11 janvier 2019	CSRT19-010-AF-006	Mitigation vulnérabilité dans Microsoft .NET	Closé

The screenshot shows the Adobe Security Bulletin page for CVE-2019-0054. It includes a table with columns for Bulletin ID, Date Published, and Priority. Below the table is a 'Summary' section and an 'Affected Versions' table.

Adobe Security Bulletin

Bulletin ID	Date Published	Priority
MSB19-02	February 1, 2019	1

Summary

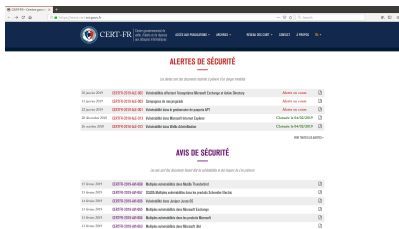
Security updates available for Adobe Acrobat and Reader (AF5219-01)

Adobe has released security updates for Adobe Acrobat and Reader for Windows and Mac OS. These updates address a critical vulnerability. Security updates must be installed as early as possible in the context of the current use.

Affected Versions

Product	Platform	Affected Versions	Advisors
Adobe DC	Continuous	2019.005.00001 and earlier versions	Windows, macOS, iOS
Adobe Reader DC	Continuous	2019.005.00001 and earlier versions	Windows, macOS, iOS
Adobe PDF	Classic, 2019	201.005.00001 and earlier versions	Windows, macOS, iOS
Adobe Reader (PDF)	Classic, 2019	201.005.00001 and earlier versions	Windows, macOS, iOS
Adobe DC	Classic, 2019	201.005.00001 and earlier versions	Windows, macOS, iOS
Adobe Reader (PDF)	Classic, 2019	201.005.00001 and earlier versions	Windows, macOS, iOS

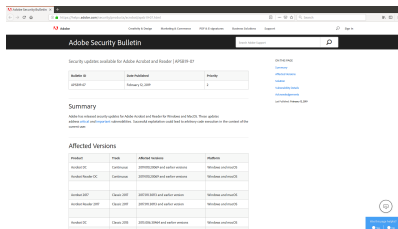
L'importance du phénomène . . . sur le plan qualitatif



The screenshot shows the CERT-FR website with two main sections: 'ALERTE DE SÉCURITÉ' and 'AVIS DE SÉCURITÉ'. Each section contains a list of security incidents with columns for date, ID, description, and status.

Date	ID	Description	Status
20 janvier 2019	CS219-0104-01-01	Attaques d'injection SQL sur Microsoft Exchange et Office 365	Alerte en cours
11 janvier 2019	CS219-0104-01-02	Attaques de phishing	Alerte en cours
22 janvier 2019	CS219-0104-01-03	Attaques de phishing et phishing de fraudeurs DPT	Alerte en cours
20 décembre 2018	CS219-0104-01-04	Attaques de phishing et fraudeurs de fraudeurs	Clôturé le 04/02/2019
20 octobre 2018	CS219-0104-01-05	Attaques de phishing	Clôturé le 04/02/2019

Date	ID	Description	Status
17 février 2019	CS219-0104-01-06	Attaques de phishing sur Microsoft Exchange	Clôturé
11 février 2019	CS219-0104-01-07	Attaques de phishing sur Microsoft Exchange	Clôturé
14 février 2019	CS219-0104-01-08	Attaques de phishing sur Microsoft Exchange	Clôturé
22 février 2019	CS219-0104-01-09	Attaques de phishing sur Microsoft Exchange	Clôturé
14 février 2019	CS219-0104-01-10	Attaques de phishing sur Microsoft Exchange	Clôturé



The screenshot shows the Adobe Security Bulletin page for CVE-2019-0104. It includes a table with columns for Bulletin ID, Date Published, and Priority. Below the table is a summary of the vulnerability and a table of affected versions.

Bulletin ID	Date Published	Priority
MSB19-01	February 12, 2019	1

Product	Track	Affected Versions	Advisors
Adobe DC	Enterprise	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro
Adobe Reader DC	Enterprise	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro
Adobe PDF	Client	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro
Adobe Reader 2019	Client	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro
Adobe DC	Client	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro
Adobe Reader 2019	Client	2019.008.000 and earlier versions	Microsoft, Cisco, Trend Micro

Deux catégories d'attaques

- attaquer un logiciel "de sécurité" :
chiffrement, pare-feu réseau, authentification, etc
- attaquer un logiciel "général", **non critique** :
navigateur, lecteur PDF, éditeur de texte, etc.
↳ élévation de privilèges sur la plateforme :
installer un malware, récupérer un fichier d'adresses, ...
⇒ **ratio surface d'attaque / gain bien plus élevé !**

Pourquoi une telle situation ?

Persistance de **bugs** dans le programmes

- limites théoriques : analyse de code est **indécidable**
- programmation = conception d'un système **discret**
- poids du code existant, complexité des applications récentes

Sécurité logicielle = préoccupation récente !

- peu de prise de conscience des développeurs (time to market) ?
- faiblesses de certains **langages de programmation**
expressivité vs contrôle vs performance
- pas (encore ?) de chaîne de développement sécurisé "standard"



Collaboration internationale

- Recensement des vulnérabilités : CERT, CVE et CWE
- Recherche “éthique” de vulnérabilité
responsible disclosure vs bug bounties

Contre-mesures → protection du logiciel

- langages de programmation, règles de codage
- compilateur, outils d'analyse de code
- plateforme d'exécution
système d'exploitation et/ou matériel “durci”

+ **certification** dans certains domaines (ex : cartes à puces)

Analyse de code pour la sécurité

- analyse de code “bas niveau”
- modèle d'attaquant
- passage à l'échelle, qualité des verdicts

exploitabilité des vulnérabilités

- quels bugs sont dangereux ?
- quantifier le coût d'une attaque ?

robustesse à l'injection de fautes

- aide au développeur : où durcir le code ?
- aide à l'évaluateur : quelles attaques possibles en multi-faute ?

Quelques projets en cours



IRT NanoElec (CLAPS, NanoTrust)

ANR Sacade (Sécurité des Systèmes Industriels)

SecureIoT-2

Collaboration CEA-List



Personnes impliquées dans Vérimag

Saddek Bensalem, Etienne Boespflug, Sylvain Boulmé, Pierre Corbineau, Cristian Ene, Claire Maiza, David Monniaux, Laurent Mounier, Marie-Laure Potet, Jean-Louis Roch, Valentin Touzeau