BRIANNE Adrien
DUBOST Yann

04/01/2021

Compte Rendu
INFO 910 - CRYPTOLOGIE
Encryptions des trames du jeu Escape From Tarkov

1) Table des matières.

2) Introduction.

2.1) La sécurité informatique.

Aujourd'hui, de plus en plus de choses sont informatisées, automatisées... On parle d'informatisation. Si cela présente beaucoup d'avantages, ça a également des contrecoups.

L'informatique est une science vaste, abstraite et qui touche à tout, qui est applicable dans énormément de domaines. Il est naturel que nous voulions des avantages de l'informatique dans autant de domaines que possible. Seulement, les ordinateurs par leur nature complexe, en couches superposées allant de l'interface jusqu'à une exécution sur le processeur fonctionnant en binaire, créés par l'Homme, ne sont pas totalement infaillibles.

Seulement aujourd'hui nos ordinateurs touchent à notre économie, des informations cruciales et/ou sensibles, aux informations personnelles de chacun… Il existe des personnes dont les intentions sont malicieuses, malveillantes. Des personnes qui usent des failles de ces systèmes pour en tirer des avantages quelconque avec des conséquences pouvant être plus ou moins importantes.

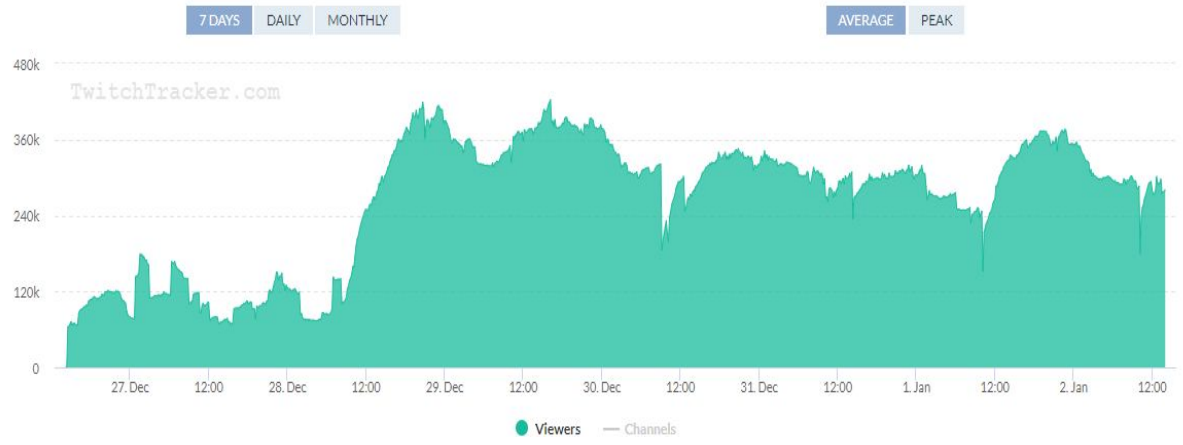
C'est là qu'intervient la notion de cybersécurité. C'est un ensemble de pratiques, de lois, d'outils, de concepts, de dispositifs, etc, ayant pour but de rendre les ordinateurs aussi peu faillibles que possible, que leur fonctionnement ne puisse être détourné à des fins autres que leur programmation initiale, ni exploités autrement que voulu lors de leur programmation. Il en va de l'économie, de la sécurité des territoires, de la sécurité des informations des entreprises, de la santé économique d'organismes, des droits individuels ou d'entités, de la sécurité des informations personnelles… Parmi ces pratiques, nous retrouvons la cryptologie, qui est une pratique visant à crypter des messages/données afin d'en assurer l'intégrité ou la confidentialité.

Cet exposé a pour but de présenter un pan de ce combat qu'est la sécurité informatique, et plus précisément parler de cryptologie.

2.2) Contexte de la présentation puis son plan.

Nous avons donc décidé de parler d'un jeu vidéo nommé Escape From Tarkov développé par Battlestate Games. C'est un jeu dont le développement est très avancé, si bien qu'il est déjà beaucoup joué (pic de 200 000 joueurs simultanés atteint en juin 2020), mais qui est toujours en phase Beta.

Voici un graphique du nombre de personnes regardant des chaînes twitch au cours des 7 derniers jours, il en atteste que ce jeu est très populaire dans le monde (source : https://twitchtracker.com/games/491931).



Ce jeu a été, et est toujours, sujet à des logiciels de triche. C'est là qu'intervient la cybersécurité, les développeurs de ce jeu ont usé de moyens pour sécuriser leur jeu et empêcher autant que possible la triche. Parmi ces moyens, ils ont crypté les trames de communication entre le jeu sur la machine utilisateur et le serveur. Battlestate Games ne peuvent pas se permettre de laisser Escape From Tarkov sans le sécuriser, un jeu avec trop de tricheurs est un jeu qui cesse vite d'être joué, si cela venait à arriver leurs pertes financières seraient terribles.

Jusqu'à maintenant, la sécurité informatique a toujours été de mettre en place des mesures pour sécuriser et rendre inexploitables des failles. Puis les hackers trouvent un moyen de contourner ces mesures, et le cycle recommence (dans la majeure partie des cas jusqu'à aujourd'hui).

Le combat contre les logiciels de triche de ce jeu est un cas intéressant. C'est pourquoi nous avons choisi de vous en parler comme sujet de cryptologie.

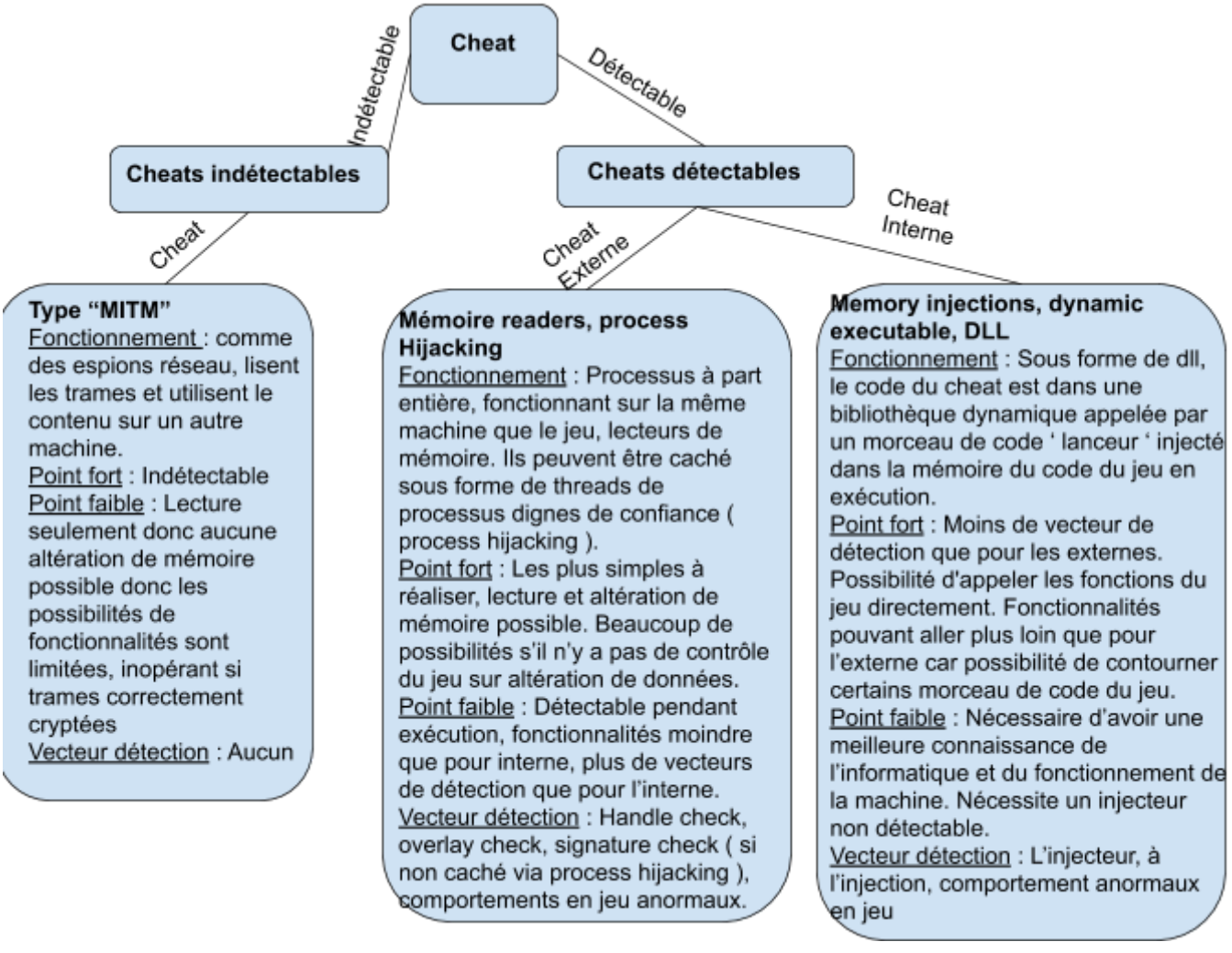
Pour traiter aussi bien le sujet de ce combat que possible, expliquer les concepts techniques et montrer pourquoi il s'inscrit comme un sujet intéressant pour parler de cryptologie nous introduirons dans un premier temps les ' cheats ' (ou logiciels de triche) et en présenterons quelques grandes familles ainsi que le type de cheat dont nous parlons plus précisément par la suite. Nous continuerons en expliquant pourquoi tricher sur des jeux, ici escape from tarkov, n'est pas anodin et comment les développeurs de ce jeu ont lutté au fil du temps contre les logiciels malveillants. Nous en viendrons alors au cœur du sujet en parlant des mesures qu'ils ont mis en place pour sécuriser leur jeu et alors nous verrons pourquoi cela s'inscrit comme sujet de cryptologie puis nous présenterons d'un point de vue technique ces méthodes mises en place. Enfin nous parlons de l'état actuel de ce combat entre hacker et développeur qui n'est pas terminé et nous verrons en quoi la cryptographie a été efficace ici. Alors nous conclurons sur cet exposé et pourquoi c'est un cas intéressant.

3) Les logiciels de triche, ou “cheats”.

3.1) Présentons quelques familles de cheat.

Comme énoncé précédemment, nous allons commencer par présenter les différentes grandes familles de logiciels de triche sur les jeux vidéos en expliquant de façon superficielle leurs fonctionnements, points forts, faiblesses et éventuels vecteurs de détection par des logiciels conçus pour les créer.

Le graphe qui suit a été créé par nos soins. Il ne représente rien d’officiel, il permet juste de cartographier et comprendre plus facilement ce dont nous parlons là.



Il existe un lien réel entre le nombre de joueurs utilisant un même logiciel de triche et le risque de détection. En effet, certains logiciels (externes, internes) sont très bien

dissimulés et non détectés. Si beaucoup de joueurs utilisent, alors de plus en plus de joueurs utilisant le même logiciel sont découvert, alors le logiciel de triche devient, par sa signature ou encore sa façon de contourner les vecteurs de détection ou d'être injecté, le dénominateur commun de ces joueurs et peut-être détecté de cette manière.

De plus, il faut bien garder à l'esprit pour la suite que indétectable et non détecté sont deux choses très différentes.

3.2) Quel type de cheat nous intéresse ici.

Maintenant que nous avons présenté les différents grands types de logiciels de triche, parlons de celui qui nous intéresse pour la suite.

Le type de logiciel de triche qui nous intéressera par la suite est celui qui est indétectable : l'espion réseau (MITM). Ce type de logiciel fonctionne sur un autre ordinateur que là où est exécuté le jeu et lit les trames puis affiche les informations sur une interface exploitable par le joueur qui triche.

En réalité, Escape From Tarkov doit lutter contre tous les types de cheats et même des bots (dont je n'ai pas parlé ici) mais nous nous intéressons qu'à celui qui va traiter de cryptographie. C'est donc un logiciel qui tourne sur une autre machine que le jeu, qui lit les trames et les utilise. Indétectable, on y voit la logique d'attaque MITM.

A présent, parlons du lien réel de ce cheat avec Escape From Tarkov.

4) Escape From Tarkov contre les cheats.

4.1) Les raisons des tricheurs et intérêts des partis.

Tout d'abord, il faut savoir que sur tous les jeux il y a toujours eu des logiciels de triches divers et variés embarquant des fonctionnalités diverses et variés directement inhérentes au type de jeu, aux contrôles serveurs plus ou moins permissifs pour le cheat, le l'anti cheat en lui même et à l'intérêt à tricher sur ce jeu.

Le point le plus important pour nous ici est l'intérêt à tricher sur ce jeu, l'impact de ces logiciels pour le jeu.

Escape From Tarkov est un FPS - Survie - Hardcore - Looter, cela signifie que le jeu est dur. Les développeurs du jeu le revendiquent dur et ne souhaitent pas toucher la communauté de joueurs ' casuals ', les joueurs aimant un jeu simple et facile à jouer. C'est un jeu très punitif, la mort dans le jeu entraîne la perte de tout l'équipement apporté dans la partie et la mort peut arriver très vite de n'importe où. L'équipement étant dur à avoir…

C'est un jeu de bonne qualité assez unique en son genre quant à son réalisme. Il attire donc beaucoup de monde, même des joueurs casuals malgré tout.

C'est là que les triches entrent en jeu, d'un jeu aussi dur peut naître une frustration ou une envie de quand même réussir malgré la difficulté du jeu, certains vont donc tricher pour y arriver.

Il existe une 2 ème vraie raison de tricher (hors l'envie de juste faire n'importe quoi, voler, passer à travers les murs, se téléporter …) c'est l'argent. En effet, si le jeu est aussi dur alors l'équipement a une vraie valeur. Des joueurs sont prêts à acheter cet équipement pour de l'argent réel. Il existe alors une demande. L'offre ne tarde pas à être créée par appât du gain. Pour être plus productif il faut être plus efficace, donc la triche est la voie suivie dans ce cas. Alors ils revendent l'équipement récupéré.

Pour indication : des joueurs utilisant des logiciels de triche puissants (type interne - externe) témoignent avoir gagné 2 500 à 3 000 euros en 1 mois à jouer. Cela peut monter plus haut encore si l'utilisation est extrême et uniquement orientée pour gagner de l'argent. Là nous parlons dans le cas d'Escape From Tarkov, pour certains jeux à certaines périodes cela peut-être plus.

Des joueurs préfèrent cependant des logiciels de triche plus sûrs, donc ils s'orientent vers des cheats ' radar ' fonctionnant sur le concept de MITM attack. Parmi ces raisons nous retrouvons l'attachement à la progression du compte de jeu car un tricheur détecté perd instantanément son compte de façon irrémédiable. Les bannissements de compte sont suivis d'un bannissement de la machine par les identifiant des comportements, les HWID, cela contribue à dissuader certains joueurs.

Il existe aussi des joueurs qui streament leur jeu sur twitch et vivent de cela, pour montrer du contenu de qualité ils peuvent tricher discrètement (sans affichage au stream du logiciel de triche). Un streamer peut gagner plusieurs dizaines de milliers d'euros par mois si il est très connu. Il serait donc désastreux pour lui que sa triche se remarque trop.

Aussi, pour les créateurs des logiciels l'intérêt est direct : vendre leur produit et donc gagner de l'argent massivement. A titre indicatif, un logiciel de triche ' privé ' est loué à ~20 personnes à 300 euros/mois. Un logiciel public peut avoir une communauté de 100 - 400 personnes avec un prix allant de 40 à 80 euros/mois. Voire même plus. Nous parlons toujours dans le cas de Escape From Tarkov.

Les développeurs du jeu en questions ne sont pas restés sans agir, nous allons voir cela par la suite. En effet, trop de tricheur sur leur jeu ferait fuir les joueurs, il faut donc y remédier autant que possible. De plus, bien que cela leur soit inavouable : un compte banni est souvent un compte racheté …

4.2) Historique de Escape From Tarkov contre les cheats.

Gardons à l'esprit ce qui a été dit précédemment. La notion d'offre et de demande a joué un rôle important ici.

Quand le jeu est sorti en Beta ouverte (jouable par tout le monde) il n'était pas très connu. Il n'y avait donc que peu de personnes voulant jouer à haut niveau dans ce jeu. Il n'avaient alors des mesures anti triche qu'assez faibles et aucune encryption des trames. Puis au fil du temps le jeu s'est fait connaître. C'est en début d'année 2020 qu'ils feront une opération commerciale qui leur vaudra la venue d'un nombre énorme de joueurs, si bien que les serveurs ne tenaient pas la charge. Quelques mois après, les logiciels de triches ont commencé à pulluler pour les raisons financières précédemment dites. Pour y remédier, Battlestate Games fera appel à BattleEye, une société de service spécialisée dans les logiciels anti-triche, ils fournissent à énormément de jeu une sécurité très correcte.

A ce stade, le combat entre les tricheurs et les équipes du jeu a commencé, chacun défendant ses intérêts. Les logiciels de types MITM (radar) devenant alors un vrai fléau pour le jeu, étant indétectable, ils ont séduit beaucoup de joueurs.

Alors les développeurs du jeu ont dû encrypter les trames pour mettre fin à ce type de triche pour limiter le nombre de tricheurs ou les ramener sur des triches détectables. Pour ce faire, une première encryption de leur trames entre le jeu client et le serveur a vu le jour en Mai 2020. Cette encryption a été crackée en 2 jours environ. Puis une seconde est apparue assez vite après, celle-ci étant alors incrackable par une attaque MITM et rendant toute triche basée UNIQUEMENT sur ce genre d'attaque purement impossible.

Nous verrons par la suite que ce combat n'a pas pris fin ici, mais la seconde encryption a mis fin aux triches purement basée sur le concept de MITM.

Pour mieux comprendre tout ce qui s'est passé, pourquoi la 1ere encryption a pu être brisée mais pas la seconde, nous allons rentrer dans le cœur du sujet des encryptions de façon technique.

5) Implémentations des mesures d'encryption.

5.1) Première implémentation ayant tenu 2 jours.

Pour la sécurisation de leur jeu, Battlestate Games a fait appel à la société d'anti-cheat BattlEye qui est utilisée par différents jeux connus (H1Z1, Fortnite, DayZ…) mais l'encryption qui avait été mis en place pour le jeu à été mis à mal seulement deux jours après son implémentation.

Pour pouvoir casser cette protection, des équipes de hackers ont fait des analyses sur des comportements du jeu et de l'anti-cheat. La procédure de reverse-engineering a été la suivante.

5.1.1) Analyse de Escape from Tarkov.

La première analyse à faire se base sur le jeu. Celui-ci est codé sous Unity qui utilise le langage C#, ce qui indique que le code du jeu peut être lu via des outils comme ILDasm ou dnSpy. Pour cette partie, le choix de dnSpy a été fait.
Une fois le jeu téléchargé, ils ont cherché le fichier AssemblyCSharp.dll qui est un fichier contenant du code du jeu, puis l'ont ouvert dans dnSpy. Une fois ouvert, la recherche du mot clé "encryption" a été faite, ce qui à mené à ce point :

```
channelCombined.bool_2 = encryptionEnabled;
channelCombined.bool_3 = decryptionEnabled;
channelCombined.bool_2 = false;
logger.LogInfo("{0}:{1}, {2}:{3}", new object[]
{
    "_encryptionEnabled",
    channelCombined.bool_2,
    "_decryptionEnabled",
    channelCombined.bool_3
});
```

Ce morceau de code se trouvait dans une classe nommée "ChannelCombined". En regardant de plus près les arguments de la classe, ils ont déduit qu'il s'agissait de la classe de mise en réseau.
On remarque que bool_2 correspond à un indicateur pour l'encryption, une recherche sur l'utilisation de ce booléen a montré un deuxième emplacement où il est appelé.

```
// Token: 0x06004C86 RID: 19590 RVA: 0x0024889C File Offset: 0x00246A9C
private ArraySegment<byte> method_5(in ArraySegment<byte> segment)
{
    if (this.bool_2)
    {
        int count = ChannelCombined.byte_1.Length;
        ArraySegment<byte> arraySegment = segment;
        byte[] array = arraySegment.Array;
        arraySegment = segment;
        int offset = arraySegment.Offset;
        arraySegment = segment;
        BEClient.EncryptPacket(array, offset, arraySegment.Count, ChannelCombined.byte_1, 0, ref count);
        return new ArraySegment<byte>(ChannelCombined.byte_1, 0, count);
    }
    return segment;
}
```

On retrouve ici un appel à “BEClient.EncryptPacket”. Grâce à cette information, on peut trouver la classe BEClient et la décortiquer pour avoir plus d’informations sur l’encryption.

On peut notamment trouver la méthode “DecryptServerPacket” qui sera appelée dans la fonction pfnDecryptPacketServer dans le fichier BECliet_x64.dll. Cette dernière fonction va décrypter les données dans un buffer et indiquer la taille de ce buffer dans un pointeur fourni par celui appelant la fonction.

PfnDecryptPacketServer n’est pas fourni par BattlEye mais est appelé par BattlEye lors de l’initialisation faite par le jeu

L’analyse des fichiers du jeu à permis aux hackers de déduire que Escape from Tarkov fait appel à BattlEye pour faire les encryptions. La prochaine étape va donc être de faire une analyse de BattleEye.

5.1.2) Analyse de BattlEye.

Grâce à la déduction faite lors de l’analyse du jeu, les hackers savent maintenant qu’ils doivent analyser BattlEye. Mais cette fois-ci, il ne s’agit pas de code C# mais de code assembleur. Donc la tâche est plus compliquée. En effet, faire de l'ingénierie-inverse de code natif est beaucoup plus compliqué.

D’autant plus que la société BattlEye utilise VMProtect pour protéger le code. VMProtect va virtualiser et modifier certains segments dans la mémoire.
Pour pouvoir inverser du binaire qui a été protégé par cet obfuscateur, il faut le déployer. Pour cela, il a fallu le charger sur un processeur local puis appliquer sa mémoire sur le disque à l’aide de l’outil Scylla. Une fois ces actions réalisées, il faut aller dans la classe DecryptServerPacket qui permet d’obtenir la fonction suivante :

```
push    198FEBB5h
call    sub_3AD7F0
```

C’est ce qui appelle une “vmentry”, qui va pousser une “vmkey” sur la stack puis va appeler “vminit” qui est le gestionnaire de la machine virtuelle.

Cette analyse de BattlEye permet de trouver un problème.
En effet les instructions de la fonction ne sont compréhensible que par le programme à cause de sa “virtualisation” par VMProtect.
Cependant, un membre du groupe a réalisé un outil qui a permis de casser cette protection.

5.1.3) Découverte de l’algorithme.

Cet outil a produit un fichier qui a réduit la fonction de 12195 instructions à 256 ce qui simplifie énormément l’analyse. De plus, certaines routines de VMProtects ont pu être détectées puis ignorées.

L’encryption à donc été détectée dans le code et les hackers ont su qu’elle commençait à cet endroit :

```
Entry point VIP:       0x20e7bf
Stack pointer:         -0x98
Already visited?:      N
-----------------------
0000: [PSEUDO]     +0x0     movq     t230         &&base
0001: [PSEUDO]     +0x0     addq     t230         0x4f8ac
0002: [PSEUDO]     +0x0     lddd     t231:32      t230         0x0
0003: [PSEUDO]     +0x0     tneb     t228:1       t231:32      0x1b
0004: [PSEUDO]     +0x0     movq     t232         &&base
0005: [PSEUDO]     +0x0     subq     t232         0x180000000
0006: [PSEUDO]     +0x0     strq     $sp          -0x98        t232
0007: [PSEUDO]     -0x98    jsq      t228:1       0x20e445     0x20e52b
| | | | Entry point VIP:       0x20e445
| | | | Stack pointer:         0x0
| | | | Already visited?:      N
| | | | -----------------------
| | | | 0000: [PSEUDO]     +0x0     jmpq     0x1a0a4a
| | | | | | | | Entry point VIP:       0x1a0a4a
| | | | | | | | Stack pointer:         0x0
| | | | | | | | Already visited?:      N
| | | | | | | | -----------------------
| | | | | | | | 0000: [PSEUDO]     +0x0     movq     t265         sr12
| | | | | | | | 0001: [PSEUDO]     +0x0     andq     t265         -0x100
| | | | | | | | 0002: [PSEUDO]     +0x0     strq     $sp          0x58         0x96715933
| | | | | | | | 0003: [PSEUDO]     +0x0     strq     $sp          0x68         t265
| | | | | | | | 0004: [PSEUDO]     +0x0     jmpq     0x1196fd
```

L'observation de la fonction de cryptage permet de voir ce qui est fait. On découvre donc que des informations sont copiées sur le début de la stack à une certaine adresse avant d'aller à une autre adresse où on peut observer une pause dans l'exécution virtuelle avant de la reprendre. Cette pause indique que la fonction appelée à ce moment-là n'est pas virtuelle. Cette fonction va donc être traduite en pseudo-code puis analysée par les hackers.

```
void __fastcall sub_3DCCB7(char *packet_data, __int64 packet_size, unsigned int xor_key)
{
  signed int i; // er9

  for ( i = 7; i < (unsigned __int64)(packet_size - 3); ++i )
  {
    *(_DWORD *)&packet_data[i] ^= xor_key;
    if ( (xor_key >> (i % 32)) & 1 )
      xor_key *= ~xor_key;
    if ( i < (int)(packet_size - (*((_BYTE *)&xor_key + i % 4) & 3) - 4) )
      i += *((_BYTE *)&xor_key + i % 3) & 3;
  }
}
```

Cette fonction décrypte les paquets en bloc de 4 octets en commençant par le 8ème avec un XOR.
La manière dont sont cryptées les données est maintenant connu des hackers.

En continuant les recherches, ils ont réussi à trouver un deuxième endroit où cette fonction est appelée mais avec une clé différente. Une analyse rapide permet de déterminer qu'un des deux appels est en fait un leurre. L'autre appel permet donc de trouver l'ordre des arguments, leur type et leur utilité. Il ne reste aux hackers plus qu'à mettre en place une attaque de type Man In The Middle qui va modifier les données qui sont transmises.

Cette première protection qui n'as donc pas tenu longtemps aura quand même servi de tests de stress pour les serveur pour voir comment ils réagissent lors de connexions simultané massive.

5.2) Encryption finale : RSA - AES.

Suite à cet échec de sécurisation, BattlEye va utiliser des algorithmes puissants pour le jeu. La sécurité actuelle du jeu repose donc sur les algorithmes RSA et AES.

5.2.1) RSA.

Le chiffrement RSA est un algorithme de cryptographie asymétrique décrit en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman. C'est l'une des méthodes de chiffrement la plus utilisée pour la sécurité.

Contrairement à un chiffrement symétrique, RSA fonctionne à l'aide de deux clés, une publique et une privée. Les deux se complètent ce qui signifie qu'un message chiffré par l'une ne peut pas être déchiffré sans l'autre.
La clé privée ne doit pas être divulguée, contrairement à la clé publique qui doit être disponible. On ne peut pas calculer la clé privée à partir de la publique il n'y a donc pas de risque en la laissant à la disponibilité de tous.

Le fonctionnement est plutôt simple, une personne A, Alice, souhaite communiquer de manière confidentielle avec une personne B, Bob.
Bob va dans un premier temps devoir créer le duo de clés, pour ce faire, il devra suivre les étapes suivantes :

- Choisir deux nombres premiers distinct p et q
- Calculer le produit de ces nombres n = pq, appelé le module de chiffrement
- Calculer $\varphi(n) = (p - 1)(q - 1)$
- Choisir un entier naturel e qui doit être premier avec $\varphi(n)$, c'est l'exposant de chiffrement
- Calculer l'entier d qui est l'inverse de e modulo $\varphi(n)$, c'est l'exposant de déchiffrement

Une fois ces étapes réalisées, on a comme clé publique (n,e) et comme clé privée (n,d).
Ces clés ne sont pas à calculer pour chaque échange puisqu'elles sont réutilisables.
On ne les re-calcule que si la clé privée est compromise ou pour un renouvellement de sécurité au bout d'un certain temps.

Pour pouvoir échanger avec Bob, Alice chiffrera donc son message avec la clé publique de Bob, pour cela elle élèvera le message à la puissance 'e' puis en fera le modulo 'n'. Le message sera chiffré à partir de ce moment-là.
Pour que Bob puisse comprendre le message, il lui suffira d'élever ce qu'il à reçu à la puissance 'd' puis d'en faire le modulo 'n'.

Bob aura finalement l'information qu' Alice voulait lui transmettre et personne ne connaissant pas la clé privée n'aura pu traduire ce message.

Cette méthode de chiffrement est très puissante ce qui en fait une des méthodes les plus utilisées

5.2.2) AES.

L'advanced Encryption Standard (AES) est un algorithme de chiffrement symétrique ayant remporté en octobre 2000 le concours AES visant à concevoir un algorithme d'encryption par bloc pouvant remplacer l'algorithme DES qui devenait de plus en plus vulnérable.

AES est actuellement l'algorithme de chiffrement le plus utilisé et le plus sécurisé. Cet algorithme a plusieurs avantages, il est très rapide, peut chiffrer et déchiffrer une grande quantité de données et ne demande pas beaucoup de ressources.

Cet algorithme prend comme données d'entrée un bloc de 128 bits(16 octets) et un clé qui fait soit 128, 192 ou 256 bits.
Le fonctionnement est le suivant :

- Les 16 octets en entrée sont permutés selon une table définie au préalable
- On place des octets dans une matrice 4x4
- On fait subir une rotation à chacune des lignes, l'incrément de rotation dépend du numéro de la ligne
- On applique ensuite une transformation linéaire à la matrice, cela consiste en la multiplication binaire de chaque membre de la matrice par des éléments issus d'une matrice auxiliaire.
- Calcul d'une clé intermédiaire, consiste à faire un XOR entre l'état actuel et la clé du tour

Ces étapes constituent un tour et doivent être répétées un certain nombre de fois selon la taille de la clé d'entrée, 10 tours pour une clé 128 bits, 12 pour une clé 192 bits et 14 tours pour une clé 256 bits.

Pour déchiffrer un texte qui à été chiffré par AES, il suffit de faire ces tours dans le sens inverse.

Cet algorithme est symétrique, les deux personnes souhaitant communiquer doivent donc connaître la clé de chiffrement.

AES est l'algorithme le plus sécurisé actuellement et plus la taille de la clé est grande, plus le nombre de traductions possible est grand.

On estime que pour cracker une clé AES de 128 bits avec un ordinateur il faudrait plus de temps que l'âge présumé de l'univers.

5.2.3) Utilisation.

Le jeu utilise donc maintenant ces deux algorithmes.
RSA est utilisé pour communiquer les clés AES en toute sécurité puis le chiffrement symétrique de AES est utilisé pour la majorité des communications. Ce dernier étant plus sûr et permettant une rapidité de traitement supérieure à la vitesse de RSA, ce qui est essentiel pour préserver de bonnes performances et donc une bonne expérience de jeu pour les joueurs. Pour preuve de leur inquiétude pour les performances, lors de l'implémentation le déploiement de ces concepts a été fait pas à pas. Au début, seulement 50% des parties étaient cryptées et uniquement les informations relatives aux joueurs, pas celles liées aux équipements ou à l'univers dynamique.

Cette utilisation rend les attaques Man In The Middle sans lecture de la mémoire du processus impossible. Le cryptage utilisé est donc efficace et utile.

C'est la première fois que BattlEye mettait en place ce type de protection, il y a donc ici aussi fallu faire des tests de stress sur les serveurs. On sait que les premier semaines où cette solution a été mise en place il y avait des jours de test et des jours sans protection.

6) L’après encryption et état global.

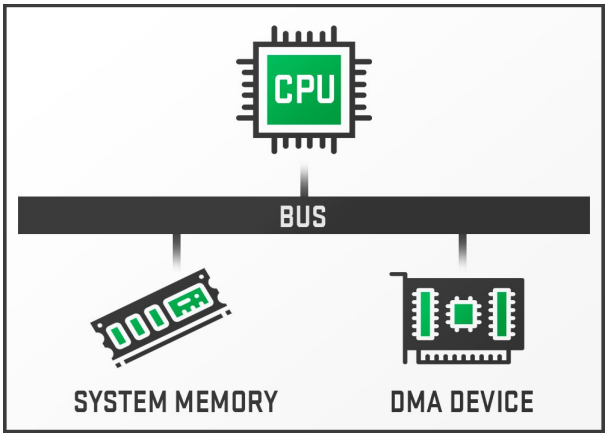
6.1) Cheat MITM après implémentation RSA - AES.

Aujourd’hui, leur méthode de cryptage rend toute triche indétectable totalement impossible. Cela ne signifie pas qu’il n’y a plus de tricheur, il y en a qui sont non détectés (mais détectables) et qui mettront plus ou moins de temps à être détectés selon les méthodes utilisées pour camoufler et le nombre de joueurs utilisant le logiciel. La triche type ‘ radar ‘ est un type de triche plutôt soft (fonctionnalités qui laissent le joueur tout de même jouer sa partie en lui offrant des avantages) et plaît à beaucoup de joueurs. Mais une triche légère sous les mêmes vecteurs de détection que des triches lourdes n’en vaut pas vraiment la peine pour tous...

Être indétectable reste une nécessité pour certains tricheurs, alors certains codeurs de logiciels de triche se sont penchés dessus pour réagir à l'encryption mise en place par Battlestate et BattlEye. Un moyen a été trouvé pour limiter autant que possible les risques de détection, sans atteindre l’impossibilité totale de détection car il faut d'une façon ou d’une autre récupérer la clé pour pouvoir décrypter à nouveau les trames. Il faut donc aller la chercher dans la mémoire de la machine, et donc risquer de se faire détecter. Nous allons parler de ce moyen.

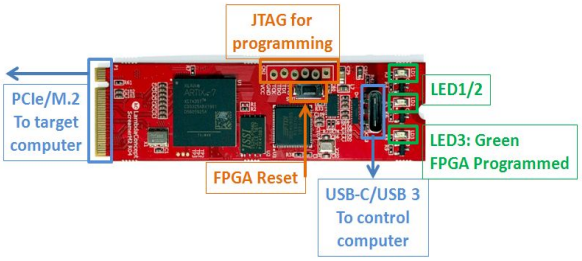
6.2) Mise en place d’une attaque DMA avec PCILeech

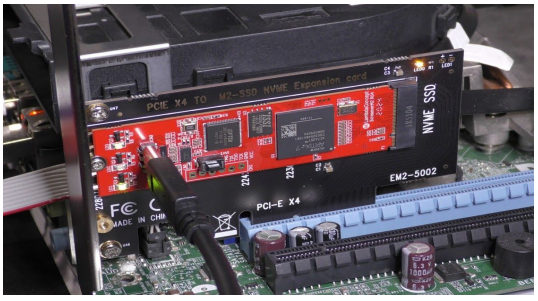
A ce stade, le seul moyen est donc de récupérer la clé en mémoire. La façon qui a été trouvée passe par une méthode DMA (direct memory access en anglais). Nous n’en expliquerons pas le fonctionnement en détail ici. Mais ce procédé permet d’accéder à toute la mémoire.



Pour ce faire, les tricheurs ont besoin d’acheter un composant, appelé screamer, qu’ils mettent directement sur leur carte mère, alors ce composant est capable de lire la

mémoire à très bas niveau, au niveau matériel. Ce qui rend la détection par un programme de haut niveau très compliqué, étudier le comportement du composant pour le logiciel d'anti-triche est une chose délicate. La clé est alors récupérée via le screamer grâce à un firmware spécifique chargé dans le dit screamer. Ensuite elle est envoyée directement à l'ordinateur où est utilisée la clé pour décrypter les trames et fonctionner comme avant. Pour envoyer la clé, c'est un câble connecté physiquement au screamer et à l'autre ordinateur, il n'y a donc pas de vecteur de détection de ce côté.





Lien du composant en question :
https://shop.lambdaconcept.com/home/44-screamer-m2-usb-c-r04-jtag-serial-pack.html

Le JTAGSerial (composant bleu) est utilisé uniquement pour flasher le screamer : charger le firmware fourni par le développeur du logiciel de triche.

Cette méthode a fait ses preuves pendant quelques semaines avant que les équipes BattlEye ne trouvent un moyen d'isoler le composant et de bannir ses utilisateurs. Le seul moyen de l'isoler, qui a été trouvé jusque-là par les équipes de BattlEye, est par le comportement du joueur lié à l'apparition du HWID (Hardware ID) et la signature du firmware (la signature du programme embarqué par le screamer).

Les concepteurs des logiciels de triche utilisant ce composant ont alors réagi en embarquant un spoofer pour le HWID et en vendant à chaque fois un code firmware, à charger dans le screamer, ayant une signature différente.
Actuellement, les équipes de développement de l'anti-triche n'ont pas encore sû répondre à cela efficacement, et ce depuis des mois. Ils ne peuvent toujours pas bannir massivement les tricheurs, mais au cas par cas seulement. Ce qui n'est pas efficace et réduit énormément les risques de bannissement pour l'utilisation de cette triche.

Après la récupération de la clé, les trames sont décryptées et tout fonctionne à nouveau comme indiqué précédemment. L'encryption à été efficace et concrètement non brisée, c'est la clé qui a été récupérée.

6.3) Bilan actuel.

Actuellement, le jeu continue son combat contre toutes les formes de triches. Du côté des logiciels à fonctionnement interne/externe il n'y a pas tellement de nouveautés. C'est toujours trouver des façon de contourner, des bypass, des processus à voler (process hijacking) côté créateurs de logiciels de triche et corriger les failles, ajouter des vecteurs de détection, réagir du côté des éditeurs du jeu et des concepteurs de l'anti-triche qui travaillent en concordance.

Là où le combat a été intéressant c'est pour la triche reposant sur la lecture des trames et toutes les actions et réactions qui ont suivi pour éviter d'être détecter ou détecter. La seconde encryption est toujours en place et de façon totale et permanente contrairement au premiers jours où c'était une partie sur deux pour tester la charge des serveurs.

7) Conclusion.

Nous pouvons très clairement voir ici l'efficacité des méthodes de cryptage actuelles. Une fois une des méthodes de cryptage efficace implémentées, les créateurs des logiciels de triche n'ont eu d'autre choix que de trouver la clé ou de se rabattre sur d'autre formes de triches pour continuer à défendre leurs intérêts. Le cryptage a donc été un obstacle incontournable dans le sens où la méthode n'a pu être brisée. Il a fallu aller récupérer la clé par des moyens ou d'autres.

Ce cas-là ne traite pas seulement de la cryptographie, il illustre également, à notre sens, ce qu'est la cybersécurité. Ce jeu du chat de la souris où la souris est le hacker ne voulant pas se faire attraper, trouve des méthodes pour contourner des failles de sécurité que les concepteurs des systèmes ou personnes travaillant dans la cybersécurité vont colmater ou même essayer d'anticiper, de tester leurs systèmes etc…

C'est un combat sans fin, à chaque fois l'attaque vient d'un nouvel endroit malgré toute les mesures prises, et le cas de ce jeu illustre parfaitement cela.

C'est un sujet qui nous intéresse réellement, nous espérons que cela vous a également intéressé et qu'il traite correctement de la cryptographie, en plus d'être un cas réel montrant les intérêts financiers de chacun, les risques/pertes de chaque parti (hacker - entreprise) et ce que représente dans la réalité le fait de sécuriser un système par des mesures qui sont sans cesse contournées par des moyens plus innovants les uns que les autres. Le second cryptage en lui-même n'a pu être brisé, il a été contourné. Il ne peut cependant pas être contourné dans tous les cas !