

Examen 2026

Reverse / Système Windows



*The SOC raised an alert on an executable named **chesscode.exe**.
The observed command line was:*

None

`chesscode.exe 1GSCjQ0u`

It seems this executable is doing system checks by calling the [systeminfo](#) command.

The SOC captured the executable and provided you a PCAP capture of the network communication from this executable.

You have to list all commands done by the executable and maybe find a hidden flag. Good luck!

Questions

1. Quel est le serveur public utilisé comme Command and Control aka C&C ?
2. Pourquoi les développeurs de malware utilisent-ils souvent des API publiques comme C&C ?
3. Cette API est utilisée pour partager des parties d'échecs. Quel est le format de la donnée ?
4. En analysant le PCAP, qu'est ce qui vous semble anormal ?
5. En analysant les exports de l'exécutable nous pouvons identifier que le développeur utilise **tree-sitter-pgn**, un parser de fichier **pgn**.
tree-sitter-pgn est disponible ici : <https://github.com/rolandwalker/tree-sitter-pgn>. (attention, **le développeur a utilisé la version 1.1.0**)

Extrayez de la capture pcap le fichier pgn afin de le parser en vous aidant de tree-sitter-pgn dans sa version node.js.

Vous devez obtenir quelque chose comme:

None

```
(series_of_games [0, 0] - [19, 7]
  game: (game [0, 0] - [19, 7]
    header: (header [0, 0] - [15, 0]
      tagpair: (tagpair [0, 0] - [0, 21]
        tagpair_delimiter: (tagpair_delimiter_open [0, 0] - [0, 1])
        tagpair_key: (tagpair_key [0, 1] - [0, 7])
        tagpair_value_delimiter: (double_quote [0, 7] - [0, 8])
        tagpair_value_contents: (tagpair_value_contents [0, 8] - [0, 19])
        tagpair_value_delimiter: (double_quote [0, 19] - [0, 20])
        tagpair_delimiter: (tagpair_delimiter_close [0, 20] - [0, 21]))
      ...
    movetext: (movetext [15, 0] - [19, 7]
      move_number: (move_number [15, 0] - [15, 2])
      san_move: (san_move [15, 2] - [15, 4])
      san_move: (san_move [15, 4] - [15, 7])
      move_number: (move_number [15, 7] - [15, 10])
      san_move: (san_move [15, 10] - [15, 12])
      san_move: (san_move [15, 12] - [15, 17])
      comment: (inline_comment [15, 17] - [15, 1516]
        comment_delimiter: (inline_comment_delimiter_open [15, 17] - [15, 19])
        comment_content: (inline_comment_text [15, 19] - [15, 1515])
```

```
comment_delimiter: (inline_comment_delimiter_close [15, 1515] - [15,
1516]))
    ...
```

6. Dans quelles fonctions retrouvez-vous des références au nom des règles de grammaire (par exemple **tagpair_key**, **comment_content**, **san_move**, **move_number**, ...) ? Dans la suite, renommez cette fonction **node_visitor**.

7. Lister les noms des règles de grammaire qui sont recherchées par cette fonction. Est-ce cohérent avec la remarque faite en question 4 ?

8. IDA semble avoir du mal à déterminer le type du premier paramètre de cette fonction. En analysant le code nous pensons que les trois premiers champs de la structure sont des pointeurs de fonctions.

Créer les types suivants dans la base de données de votre outil de reverse engineering préféré:

```
C/C++
struct Vtable_t
{
    void (__fastcall *fn1)(void *, _OWORD *);
    void (__fastcall *fn2)(void *, _OWORD *);
    void (__fastcall *fn3)(void *, _OWORD *);
};

struct Context_t {
    Vtable_t          vtable;
};
```

- Appliquez le type **Context_t*** au premier paramètre de la fonction identifiée précédemment.
- Identifiez l'appel à cette fonction depuis la fonction main.

Pouvez-vous identifier avec quelle adresse sont initialisées les champs **fn1**, **fn2**, **fn3**? (N'oubliez pas de rappeler le décompilateur pour qu'il prenne en compte les nouveaux types)

C'est un modèle de programmation que l'on rencontre souvent dans le monde objet. Vous pouvez mettre à jour le type **Vtable_t** en indiquant que le premier paramètre de chaque fonction sera toujours du type **Context_t***.

```
C/C++
struct Vtable_t
{
    void (__fastcall *fn1)(Context_t *, _OWORD *);
    void (__fastcall *fn2)(Context_t *, _OWORD *);
    void (__fastcall *fn3)(Context_t *, _OWORD *);
};
```

9. La fonction **fn3** fait référence à trois constantes. Pouvez-vous en déduire l'algorithme?

Ce type d'algorithme se décompose généralement en trois fonctions : **init**, **update**, **finalize**.

Créer les types suivant:

```
C/C++
struct HashAlgoContext_t {
    //algo
    uint64_t size;
    uint32_t buffer[4];
    uint8_t input[64];
    uint8_t digest[16];
}
```

Mettez à jour le type **Context_t**:

```
C/C++
struct Context_t {
    Vtable_t          vtable;
    HashAlgoContext_t hash;
}
```

La fonction **fn3** doit maintenant ressembler à :

```
C/C++
void __fastcall sub_140001000(Context_t *a1)
{
```

```

HashAlgoContext_t *p_hash; // rcx

p_hash = &a1->hash;
p_hash->size = 0;
p_hash->buffer[0] = 0x67452301;
p_hash->buffer[1] = 0xEFCDAB89;
p_hash->buffer[2] = 0x98BADCFE;
p_hash->buffer[3] = 0x10325476;
}

```

10. En analysant le début de la fonction **main** nous observons l'appel à la fonction **memset** sur une taille de 4200 octets à partir du champ **hash** de notre fonction. Ceci indique que notre structure n'est pas complète et doit contenir un buffer de 4096 octet (**104** étant la taille de la structure **HashAlgoContext_t**). Nous allons mettre notre type en conséquence:

```

C/C++
struct Context_t {
    Vtable_t          vtable;
    HashAlgoContext_t hash;
    char              Buffer[4096];
}

```

D'après vous à quoi sert le champ **Buffer** ?

11. En utilisant le debugger de votre choix, lancer l'exécutable chesscode.exe, puis positionner un point d'arrêt juste avant l'appel à la fonction **tree_sitter_pgn** depuis la fonction main. Écrire dans la mémoire ce qui a été extrait du pcap en échappant les caractères spéciaux (\ " \n). Par exemple en utilisant windbg on peut utiliser la commande suivante:

```

None
ea STACKADDRESS "[Event \"Casual game\"]\n[Site \"London ENG\"]\n[Date
\"1851.06.21\"]\n[EventDate \"?\"]\n[Round \"?\"]\n[Result \"1-0\"]\n[White
\"Adolf Anderssen\"]\n[Black \"Lionel Adalbert Bagration Felix
Kieseritzky\"]\n[ECO \"C33\"]\n[WhiteElo \"?\"]\n[BlackElo \"?\"]\n[Source \"La
Régence, v3 n7, July 1851, pp221-222\"]\n[SourceNote \"ends
19.Ke2\"]\n[PlyCount \"45\"]\n\n1.e4 e5 2.f4 exf4

```

```
{u6EBVqDGfPWDYs78S0GEJQv6yyyyEICagHEosPsjY0jMwXX57HvtE/128JrACUyfCvD6fRNxN3Yxen
EYd/G9pWJDFmR2EPSA6x06xLwQ0QSqe2kdm30z+3RphXwvipHZCC9PE/sQ3tsWebMHFn2x8/NkLFpDU
Rva+tWGaLdiu/507CzbUx0ywb3AXFhthpknjGkQ776H0IYnRj40Pst6ZhnXLfncbBYkWhcQPqAsGEmi
9D3WV8GSfac36oCN6L7MD3zi/n8QUaEeJv1BVF6zgjGjq+MQ03YQWq0Rs011I+XhR/sQqU1bCQylxC1
CfHAxqgDGYhxowdc7+m+MkWEWjwQqj0kvPLQtCdt3K+8HYtCqe2kdm30z+3RphXwvipHZFY5BEgH35a
j8/6/sJp4QXzwApdFx4G33j1IvPy1F0ZGqAMZiHGjB1zv6b4yRYRaPTHBPCkpbvD19tk04s1MSaoAx
mIcaMHX0/pvjJFhFo8Eko9JLzy0LXA09yvVB2LQqntPHZtZs/t0aYV8L4qR2TSJ+m5sUo/ZULOaciT7
4e2NN1CdMP7Rg5Gn9qXbGsh7pdq1xateyB8YoSrKhEe7NqXatcWrXsgfGKEqyoRHuzYxo6vjENN2EFq
jkbDpdSPsiSG38Zu7yqQzIKJLTD8mfugi/xcYtjSvr7WVmKSE+8qqAMZiHGjB1zv6b4yRYRaPTHBPCk
ipbvD19tk04s1MSaoAxmIcaMHX0/pvjJFhFo+12rXFq17IHxihKsqER7s2QdshHttUvqaMRwVW51099
qoAxmIcaMHX0/pvjJFhFo9McE8IqKlu8PX22TTizUxJlAdii/dzeTiPeEaPclPbPY02UJ0w/tGDkaf2
pdsayHuUB2KL93N50I94Ro9yU9s9GPfEJy00P2wBzwByyr6aqTwApdFx4G33j1IvPy1F0ZECMJbBLu4
YXKz+SPXGx0tsTHBPCkpbvD19tk04s1MSZQHov3c3k4j3hGj3JT2z3WKbcXfJx1hUlyqa0gfojr13
r1J7gLaFXu7ocD40AnAgIwlsEu7hhcrP5I9cbE62yNN1CdMP7Rg5Gn9qXbGsh7qgDGYhxowdc7+m+Mk
WEWjwQqj0kvPLQtCdt3K+8HYtCqe2kdm30z+3RphXwvipHZGPfEJy00P2wBzwByyr6aqZd65Se4C2hV
7u6HA+NAJwICMJbBLu4YXKz+SPXGx0ts2X/1oaL8rF/Zgl/X5oMGw1xT+F2NbU3FwtJL4coD09oVjKE
SAfflqPz/r+wmnhBfN4bGVNr1nAWtoVd9+ObA8/kyppy+dvd508vhe+Auuk+MaRDvvoc4hidGPjQ+y3
pmGdct+dxsFiRaFxA+oCwYSaL0PdZXwZJ9pzfqgI3ovswxo6vjENN2EFqjkbDpdSPsMaOr4xDthBao
5Gw6XUj7I7GDAyVWJ38CotBSZXSGxNPVxJJ1f6pa+9N1XUatsU3MJUUhD19/o2iiADmsJP/LA==}
3.Bc4 Qh4+ 4.Kf1 b5 5.Bxb5 Nf6 6.Nf3 Qh6\n7.d3 Nh5 8.Nh4 Qg5 9.Nf5 c6 10.g4 Nf6
11.Rg1 cxb5 12.h4 Qg6\n13.h5 Qg5 14.Qf3 Ng8 15.Bxf4 Qf6 16.Nc3 Bc5 17.Nd5 Qxb2
18.Bd6\nBxg1 {sgYkr46C73SDT3KwXr/Mmg==} 19. e5 Qxa1+ 20. Ke2 Na6 21.Nxg7+ Kd8
{7p/cMgz2+zS8h7uLT36V6aTkZk9eLdRyIu+NQx4plthocAtoAf2wRnnKdEE31W2/YMJ8fZKflzmxq
LFPZ0u8nufVNeWk2iioC0oeXC20oVrtzWXIT7ersmiLo/LSjgGeXCETijdkXSZqaBIVAieGrODJPPvj
9Gnt+PODUBLPBKUiTtV6PbM028h6LI9Ir0bjn5LD51IF9Ks1upsvxabeMjStpcE+Boc30gCCPfdN2qv
ZtPYoPuoUFpjP+ULyYBZ} 22.Qf6+ Nxf6\n23.Be7# "
```

Continuer l'exécution. Mettre dans le rapport un screenshot de l'exécution du programme.

12. La fonction **fn1** est associée à la fonction **update** de l'algorithme de hash trouvé en question 9. En vous aidant d'une analyse dynamique (question 11), déterminer quelle est la donnée utilisée comme input à l'algorithme de hash ?

Rappel:

- **dq** **<address>** permet d'afficher le contenu de la mémoire sous forme de quadword
- **da** **<address>** permet d'afficher le contenu de la mémoire sous forme de caractère ASCII

13. En analysant la fonction pointée par **fn2**, nous observons l'appel à la fonction **sub_1400026E0**. Cette fonction fait référence à un tableau à l'adresse **0x1400627F0** (**byte_1400627F0**) présent dans une section d'initialisation globale. Identifier l'algorithme implémenté par cette fonction.

14. En continuant d'analyser cette fonction **fn2**, qui semble vraiment intéressante, identifier le but de la fonction **sub_140001E80** ?

15. Les fonctions **sub_140001650** et **sub_140001640** sont associées à l'algorithme **AES ECB**. En analysant le code quelle donnée est utilisée comme clé pour l'algorithme AES ?

16. En vous aidant du script suivant, ou par toute autre manière qui vous semble adéquate, pouvez-vous décoder le flag ?

```
Python
from tree_sitter import Language, Parser, Node
import tree_sitter_pgn as ts_pgn
from Crypto.Cipher import AES
from base64 import b64decode

PGN_LANGUAGE = Language(ts_pgn.language())
PARSER = Parser(PGN_LANGUAGE)

def main() -> None:
    with open('game.txt', 'rb') as file:
        tree = PARSER.parse(file.read())

    def traverse_tree(node: Node):
        for n in node.children:
            if n.type == "san_move":
                print(n.text.decode("ascii"), end="")
            elif n.type == "inline_comment_text":
                print(n.text.decode("ascii"))
                key = #TODO
                cipher = AES.new(key, AES.MODE_ECB)
                txt = cipher.decrypt(b64decode(n.text))
                print(txt)
            else:
                traverse_tree(n)

    traverse_tree(tree.root_node)
if __name__ == '__main__':
    main()
```

Ce script possède les dépendances suivantes :

None

pycryptodome

tree-sitter-pgn==1.1.0

tree-sitter