Μέθοδοι Συσκότισης Κώδικα

Δημήτριος Υφαντίδης (AEM:3938)
ydimitri@csd.auth.gr

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης,
Τμήμα Πληροφορικής της Σχολής Θετικών Επιστημών

9 Μαΐου 2023

Εισαγωγή

Η Συσκότιση είναι η απόκρυψη του επιδιωκόμενου νοήματος της επικοινωνίας, καθιστώντας το μήνυμα δυσνόητο, συνήθως με συγκεχυμένη και διφορούμενη γλώσσα.

Όπως και η κρυπτογράφηση, συνεισφέρει στην προστασία ενός συνόλου δεδομένων από κακόβουλες οντότητες. Όμως, η συσκότιση και η κρυπτογράφηση είναι δυο έννοιες διαφορετικές που δεν πρέπει να συγχέονται.

Συσκότιση και Κρυπτογράφηση

Η κρυπτογράφηση ασχολείται με την αλλαγή δεδομένων σε άλλα δεδομένα για την πλήρη απόκρυψη του αρχικού μηνύματος. Η συσκότιση είναι η απόκρυψη του σκοπού ή του νοήματος των δεδομένων χωρίς να αλλάζουν τα ίδια τα δεδομένα.

Στόχος της συσκότισης δεν είναι να κάνει αδύνατη την κατανόηση ενός μηνύματος, αλλά να προκαλέσει συμφόρηση σε όποιον το προσπαθήσει.

- **Συσκότιση**:
  "The President has arrived" $\rightarrow$ "The Eagle has landed"
- **Κρυπτογράφηση**:
  "The President has arrived" $\rightarrow$ "buwtixwvfvliaignbvgciu"

Συσκότιση Κώδικα Λογισμικού

Στην ανάπτυξη λογισμικού, η συσκότιση είναι η πράξη της δημιουργίας πηγαίου κώδικα ή κώδικα μηχανής που είναι δύσκολο να κατανοηθεί από ανθρώπους ή υπολογιστές.

Όπως και η απόκρυψη στη φυσική γλώσσα, μπορεί να χρησιμοποιεί περιττές ή/και δυσνότητες (αλλά ισοδύναμες) εκφράσεις για τη σύνθεση δηλώσεων.

Συσκότιση Κώδικα Λογισμικού

Οι προγραμματιστές μπορεί να αποκρύψουν σκόπιμα τον ιδιόκτητο κώδικά τους για να αποκρύψουν τον σκοπό του (ασφάλεια μέσω αδιαφάνειας) ή τη λογική του, κυρίως για να αποτρέψουν τόσο την παραποίηση, όσο και το reverse engineering καθώς και να προστατέψουν τα πνευματικά δικαιώματά τους.

Αυτό μπορεί να γίνει χειροκίνητα ή με τη χρήση ενός αυτοματοποιημένου εργαλείου, με την τελευταία να είναι η προτιμώμενη τεχνική στη βιομηχανία (π.χ. ProGuard, javascript-obfuscator).

Συσκότιση Κώδικα Λογισμικού

Πρώτοι αριθμοί < 1000

```
print(list(filter(None,map(lambda y:y*reduce(lambda x,y:x*y!=0,
map(lambda x,y=y:y%x,range(2,int(pow(y,0.5)+1))),1),range(2,1000)))))
```

Πρώτοι 10 όροι της ακολουθίας Fibonacci

```
print(list(map(lambda x,f=lambda x,f:(f(x-1,f)+f(x-2,f)) if x>1 else 1:
f(x,f), range(10))))
```

Mandelbrot set

```
print((lambda Ru,Ro,Iu,Io,IM,Sx,Sy:reduce(lambda x,y:x+'\n'+y,map(lambda y,
Iu=Iu,Io=Io,Ru=Ru,Ro=Ro,Sy=Sy,L=lambda yc,Iu=Iu,Io=Io,Ru=Ru,Ro=Ro,i=IM,
Sx=Sx,Sy=Sy:reduce(lambda x,y:x+y,map(lambda x,xc=Ru,yc=yc,Ru=Ru,Ro=Ro,
i=i,Sx=Sx,F=lambda xc,yc,x,y,k,f=lambda xc,yc,x,y,k,f:(k<=0)or (x*x+y*y
>=4.0) or 1+f(xc,yc,x*x-y*y+xc,2.0*x*y+yc,k-1,f):f(xc,yc,x,y,k,f):chr(
64+F(Ru+x*(Ro-Ru)/Sx,yc,0,0,i)),range(Sx))):L(Iu+y*(Io-Iu)/Sy),range(Sy
))))(-2.1, 0.7, -1.2, 1.2, 30, 80, 24))
```

source: Python Programming FAQ

Μέθοδοι Συσκότισης Κώδικα

- Αναδιάταξη κώδικα
- Αλλαγή ονομάτων μεταβλητών (identifier renaming)
- Εισαγωγή πλεοναστικού/αχρείαστου κώδικα
- Εισαγωγή μεταπηδήσεων (conditional, unconditional)
- Εκ νέου ανάθεση μεταβλητών (identifier reassigning)
- Κωδικοποίηση συμβολοσειρών
- Παραποίηση της ροής ελέγχου
- Συνδυασμός των παραπάνω τεχνικών

Μέθοδοι Συσκότισης Κώδικα

Αλλαγή ονομάτων μεταβλητών:

Original Source Code Before Rename Obfuscation

```
private void
CalculatePayroll (SpecialList employee-
Group) {
    while (employeeGroup.HasMore()) {
        employee =
employeeGroup.GetNext(true);
        employee.UpdateSalary();
        Distribute Check(employee);
    }
}
```

Reverse-Engineered Source Code After Rename Obfuscation

```
private void a(a b) {
    while (b.a()) {
        a = b.a(true);
        a.a ();
        a.(a);
    }
}
```

source: "What is Code Obfuscation?"

Μέθοδοι Συσκότισης Κώδικα

Παραποίηση ροής του ελέγχου:

Original Source Code Before Control Flow Obfuscation

```
public int CompareTo (Object o) {
    int n = occurrences -
    ((WordOccurrence)o) .occurrences ;
    if (n == 0)  {
       n = String.Compare
  (word, ((WordOccurrence)o) .word) ;
    }
    return (n) ;
}
```

Reverse-Engineered Source Code After Control Flow Obfuscation

```
private virtual int _a(Object A+0) {
    int local0 ;
    int local1 ;
    local 10 = this.a - (c)  A_0.a;
    if (local10  != 0) goto i0 ;
    while  (true)  {
       return local1;
    }
    i1: local10 =
System.String.Compare(this.b,  (c)
A_0.b) ;
   goto  i0;
}
```

source: "What is Code Obfuscation?"

Συσκότιση - Επιστημονική Σκοπιά

- **Συσκότιση μαύρου κουτιού**:
  Κρυπτογραφικό πρωτότυπο που θα επέτρεπε σε ένα πρόγραμμα υπολογιστή να συσκοτιστεί με τέτοιο τρόπο ώστε να είναι αδύνατο να προσδιοριστεί οτιδήποτε γι΄ αυτό εκτός από τη συμπεριφορά εισόδου και εξόδου του.
  Αποδείχθηκε ότι είναι αδύνατη, έστω και θεωρητικά.

Συσκότιση - Επιστημονική Σκοπιά

- **Συσκότιση μαύρου κουτιού**:
  Κρυπτογραφικό πρωτότυπο που θα επέτρεπε σε ένα πρόγραμμα υπολογιστή να συσκοτιστεί με τέτοιο τρόπο ώστε να είναι αδύνατο να προσδιοριστεί οτιδήποτε γι΄ αυτό εκτός από τη συμπεριφορά εισόδου και εξόδου του. Αποδείχθηκε ότι είναι αδύνατη, έστω και θεωρητικά.

- **Συσκότιση μη-διακρισιμότητας**:
  Συμβολίζεται με $i\mathcal{O}$ (Indistinguishability Obfuscation). Έχει την καθοριστική ιδιότητα ότι η απόκρυψη οποιωνδήποτε δύο προγραμμάτων που υπολογίζουν την ίδια μαθηματική συνάρτηση οδηγεί σε προγράμματα που δεν μπορούν να διακριθούν το ένα από το άλλο.

Black Box Obfuscation

Υπόθεση:

- $\vec{\alpha}, \vec{\beta} \in \{0,1\}^k$, με $\vec{\beta} \neq \vec{0}$
- $C : \{0,1\}^k \to \{0,1\}^k$ με

$$C(\vec{x}) = \begin{cases} \vec{\beta} & \text{για } \vec{x} = \vec{\alpha} \\ \vec{0} & \text{για } \vec{x} \neq \vec{\alpha} \end{cases}$$

- $Z : \{0,1\}^k \to \{0,1\}^k$ με $Z(\vec{x}) = \vec{0}, \ \forall \vec{x} \in \{0,1\}^k$
- $D : (\{0,1\}^k)^{\{0,1\}^k} \to \{0,1\}$ με

$$D(\mathcal{X}) = \left\{ \begin{array}{ll} 1 & \text{για } \mathcal{X}(\vec{\alpha}) = \vec{\beta} \\ 0 & \text{για } \mathcal{X}(\vec{\alpha}) \neq \vec{\beta} \end{array} \right| \quad time(\mathcal{X}) \leq poly(k)$$

- $\mathcal{BBO}$ ένας black box obfuscator όπου $\mathcal{BBO}(P)$ η συσκότιση μαύρου κουτιού του P για κάθε πρόγραμμα P.
- $C' \leftarrow \mathcal{BBO}(C)$ και $D' \leftarrow \mathcal{BBO}(D)$

Black Box Obfuscation

Αποδυκνείεται ότι υπάρχει πρόγραμμα που δεν μπορεί να συσκοτιστεί από τον $\mathcal{BBO}$, άρα δεν υπάρχει συσκότιση μαύρου κουτιού.

Black Box Obfuscation

Αποδυκνείεται ότι υπάρχει πρόγραμμα που δεν μπορεί να συσκοτιστεί από τον $\mathcal{BBO}$, άρα δεν υπάρχει συσκότιση μαύρου κουτιού.

Απαγωγή σε άτοπο (α΄ μέρος):
Έστω ότι ο επιτιθέμενος έχει πρόσβαση μόνο στο πρόγραμμα C', τότε είναι πρακτικά αδύνατον να βρει τον κωδικό α, δηλαδή:

$$Pr[C(\vec{x_0}) = \vec{\beta} \mid \vec{x_0} \xleftarrow{\$} \{0,\ 1\}^k] \ = \ 2^{-k}$$

δηλαδή $\simeq 0$ για (όχι και τόσο) μεγάλο k, π.χ. $k > 50$. Άρα δεν μπορεί να ξεχωρίσει το C' από το Z σε πολυωνυμικό χρόνο επειδή $C'(\vec{x_0}) = Z(\vec{x_0}) = \vec{0}$, για κάθε $\vec{x_0} \in \{0,\ 1\}^k$ εντός πολυωνυμικού χρονικού περιθωρίου.

Black Box Obfuscation

Απαγωγή σε άτοπο (β΄ μέρος):
Όμως, αν έχει πρόσβαση και στο D' μπορεί να εξάγει το συμπέρασμα ότι $D'(C') = 1$ ενώ $D'(Z) = 0$, άρα μπορεί να διακρίνει ότι η εσωτερική υλοποίηση του C' δεν είναι αυτή του Z. Επομένως, γνωρίζει παραπάνω πράγματα για το C' πέρα από τη συμπεριφορά εισόδου-εξόδου. Τα προγράμματα C και D μπορούν να εννοποιηθούν στο πρόγραμμα:

$$F(m, \vec{x}) := \begin{cases} C(\vec{x}) & \text{για } m = 0 \\ D(C) & \text{για } m = 1 \end{cases}$$

ή αλλιώς

$$F(\mathcal{X}, m, \vec{x}) := \begin{cases} \mathcal{X}(\vec{x}) & \text{για } m = 0 \\ D(\mathcal{X}) & \text{για } m = 1 \end{cases}$$

Άρα, με βάση τα προηγούμενα $\nexists\, \mathcal{BBO}(F)$.

Συμβιβασμός

Η καλύτερη δυνατή Συσκότιση:
Ο obfuscator $\mathcal{O}$ λέγεται βέλτιστος όταν εγγυάται ότι κάθε πληροφορία που δεν κρύβεται από το συσκοτισμένο πρόγραμμα, δεν αποκρύπτεται ούτε από οποιοδήποτε άλλο πρόγραμμα ίδιου μεγέθους που υπολογίζει την ίδια λειτουργία. Έτσι η συσκότιση είναι (κυριολεκτικά) η καλύτερη δυνατή.

Αυτό επιτυγχάνεται μέσω της παραλαγής της συσκότισης μαύρου κουτιού, της συσκότισης **μη διακρισιμότητας**.

Indistinguishability Obfuscation

Έστω $i\mathcal{O}$ κάποιος ομοιόμορφος πιθανοτικός αλγόριθμος πολυωνυμικού χρόνου. Τότε ο $i\mathcal{O}$ ονομάζεται Indistinguishability Obfuscator αν και μόνο αν ικανοποιεί τις συνθήκες **Πληρότητας** και **Μη Διακρισιμότητας**, δηλαδή:

Indistinguishability Obfuscation

Έστω $i\mathcal{O}$ κάποιος ομοιόμορφος πιθανοτικός αλγόριθμος πολυωνυμικού χρόνου. Τότε ο $i\mathcal{O}$ ονομάζεται Indistinguishability Obfuscator αν και μόνο αν ικανοποιεί τις συνθήκες **Πληρότητας** και **Μη Διακρισιμότητας**, δηλαδή:

1. Για κάθε λογικό κύκλωμα $C(\vec{x}) : \{0, 1\}^n \rightarrow \{0, 1\}$, ισχύει:

$$Pr[C'(\vec{x}) = C(\vec{x}) : C' \leftarrow i\mathcal{O}(C)] = 1$$

Indistinguishability Obfuscation

Έστω $i\mathcal{O}$ κάποιος ομοιόμορφος πιθανοτικός αλγόριθμος πολυωνυμικού χρόνου. Τότε ο $i\mathcal{O}$ ονομάζεται Indistinguishability Obfuscator αν και μόνο αν ικανοποιεί τις συνθήκες **Πληρότητας** και **Μη Διακρισιμότητας**, δηλαδή:

1. Για κάθε λογικό κύκλωμα $C(\vec{x}) : \{0, 1\}^n \rightarrow \{0, 1\}$, ισχύει:

$$Pr[C'(\vec{x}) = C(\vec{x}) : C' \leftarrow i\mathcal{O}(C)] = 1$$

2. Για κάθε ζεύγος λογικών κυκλωμάτων C_0, C_1 ίδιου μεγέθους και για κάθε εχθρικό, πιθανοτικό αλγόριθμο πολυωνυμικού χρόνου, $\mathcal{A}$ (adversary):

$$Pr[\mathcal{A}(i\mathcal{O}(C_0)) = 1] \simeq Pr[\mathcal{A}(i\mathcal{O}(C_1)) = 1]$$

Δηλαδή, ο $\mathcal{A}$ δεν μπορεί να διακρίνει αν το αρχικό πρόγραμμα είναι το C_0 ή το C_1.

Ύπαρξη του $i\mathcal{O}$

Το αν είναι δυνατόν να υπάρξει ένας $i\mathcal{O}$ obfuscator εξαρτάται από τις απαντήσεις σε άλλα ανοιχτά ερωτήματα της επιστήμης υπολογιστών, όπως η ύπαρξη των **One-way functions** και η απάντηση στο **P vs NP**. Όλες οι σημερινές υλοποιήσεις βασίζονται σε ευρετικές μεθόδους.

Ύπαρξη του $i\mathcal{O}$

Το αν είναι δυνατόν να υπάρξει ένας $i\mathcal{O}$ obfuscator εξαρτάται από τις απαντήσεις σε άλλα ανοιχτά ερωτήματα της επιστήμης υπολογιστών, όπως η ύπαρξη των **One-way functions** και η απάντηση στο **P vs NP**. Όλες οι σημερινές υλοποιήσεις βασίζονται σε ευρετικές μεθόδους.

Οι «πέντε κόσμοι» του Russell Impagliazzo:

1. **Algorithmica:** P $=$ NP, υπάρχει $i\mathcal{O}$
2. **Heuristica:** Τα NP προβλήματα είναι σχετικά εύκολα, δεν υπάρχει $i\mathcal{O}$
3. **Pessiland:** BPP $\neq$ NP, δεν υπάρχουν one-way functions, δεν υπάρχει $i\mathcal{O}$
4. **Minicrypt:** υπάρχουν one-way functions, δεν υπάρχει ασφαλής κρυπτογραφία δημοσίου κλειδιού, δεν υπάρχει $i\mathcal{O}$
5. **Cryptomania:** υπάρχει ασφαλής κρυπτογραφία δημοσίου κλειδιού, δεν υπάρχει $i\mathcal{O}$
6. (extra) **Obfustopia:** υπάρχει $i\mathcal{O}$

Βιβλιογραφικές Αναφορές

1. Wikipedia
   "Black-Box obfuscation"
   "Indistinguishability obfuscation"
2. Wikibooks
   "Τεχνική Νομοθεσία Για Μηχανικούς Πληροφορικής/Τεχνικά μέσα για την προστασία του λογισμικού από παράνομη αντιγραφή - πειρατεία"
3. Aayush Jain; Huijia Lin; Amit Sahai (2021)
   "Indistinguishability Obfuscation from Well-Founded Assumptions"
4. Tim Robinson (2022)
   "Obfuscation Vs. Encryption (Easily Explained)"
5. Chandan Kumar Behera; D. Lalitha Bhaskari (2015)
   "Different Obfuscation Techniques for Code Protection"

Βιβλιογραφικές Αναφορές

6. Boaz Barak; Oded Goldreich; Russell Impagliazzo; Steven Rudich; Amit Sahai; Salil Vadhan; Ke Yang (2010)‘
   “On the (Im)possibility of Obfuscating Programs”
7. Valentine Kabanets; Shawn Andrews (2011)
   “based on ‘A Personal View of Average Case Complexity’ by R. Impagliazzo, 1995”
8. Shafi Goldwasser; Guy N. Rothblum
   “On Best-Possible Obfuscation”