ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Κρυπτογραφια
9ο Εξαμηνο

3η Σειρά Ασκήσεων

Ονοματεπώνυμο (Αριθμός Μητρώου):
Γεώργιος Γκοτζιάς (03119047)

18 Φεβρουαρίου 2024

Άσκηση 1

Έστω $n = p \cdot q, e$ με $gcd(e, \phi(n)) = 1$. Έστω αλγόριθμος Α τέτοιος, ώστε για είσοδο $c \leftarrow \$\mathbb{Z}_q = n$, να λύνει το RSA με πιθανότητα ϵ, δηλαδή $Pr[A(c) = RSA(c)] = \epsilon$.
Θεωρούμε τον παρακάτω αλγόριθμο.

Algorithm 1 Random Self-Reducible algorithm for RSA

$a \leftarrow \$\mathbb{Z}_n^*$
$c_1 \leftarrow a^e \cdot c$
$m_1 \leftarrow A(c_1)$
if $m_1^e \neq c_1$ **then**
    return "fail"
else
    return $a^{-1} \cdot m_1$
end if

Ο παραπάνω αλγόριθμος με πιθανότητα ϵ λύνει το RSA, ενώ διαφορετικά επιστρέφει ότι δεν το έλυσε για οποιοδήποτε στοιχείο, επομένως με διαδοχικές επαναλήψεις λύνουμε με μεγάλη πιθανότητα το RSA.
Ορθότητα
Όταν δεν αποτυγχάνει ο αλγόριθμος: Είναι $c_1 = m_1^\epsilon$, άρα $m_1 = c_1^{\frac{1}{e}}$
Όμως $c_1 = a^e \cdot c$.
Άρα, ο αλγόριθμος επιστρέφει $m = a^{-1} \cdot m_1 = a^{-1} \cdot c_1^{\frac{1}{e}} = a^{-1} \cdot (a^e \cdot c)^{\frac{1}{e}} = a^{-1} \cdot a \cdot c^{\frac{1}{e}} = c^{\frac{1}{e}}$, το οποίο είναι η αποκρυπτογράφηση του c.
Σημειώνεται ότι με a^{-1} συμβολίζεται ο αντίστροφος του a στην ομάδα $\mathbb{Z}_n^*$ γι΄ αυτό και είναι εύκολος ο υπολογισμός.

Άσκηση 2

Αναγωγή από SDH σε CDH
Έστω αλγόριθμος Α που λύνει το CDH, δηλαδή για εισόδους g, g^α, g^β επιστρέφει $A(g, g^\alpha, g^\beta) = g^{\alpha \cdot \beta}$.
Δίνοντας ως είσοδο g, g^x, g^x επιστρέφει $A(g, g^x, g^x) = g^{x^2}$, δηλαδή λύνει το SDH.
Αναγωγή από CDH σε SDH
Έστω αλγόριθμος Α που λύνει το SDH, δηλαδή για εισόδους g, g^x επιστρέφει $A(g, g^x) = g^{x^2}$.
Με βάση τον παραπάνω αλγόριθμο υπολογίζουμε τα $A(g, g^{x_1}) = g^{x_1^2}$, $A(g, g^{x_2}) = g^{x_2^2}$ και $A(g, g^{x_1+x_2}) = g^{(x_1+x_2)^2}$, όπου $g^{x_1+x_2} = g^{x_1} \cdot g^{x_2}$.
Όμως έχουμε ότι $g^{(x_1+x_2)^2} = g^{x_1^2} \cdot g^{2x_1x_2} \cdot g^{x_2^2}$, οπότε υπολογίζεται το $g^{x_1 \cdot x_2}$, άρα λύνεται το CDH για εισόδους g, g^{x_1}, g^{x_2}.
Από τις δύο παραπάνω αναγωγές έχουμε ότι τα προβλήματα CDH και SDH είναι ισοδύναμα.
Αναγωγή από SDH σε IDH
Έστω αλγόριθμος Α που λύνει το IDH, δηλαδή για είσοδο g, g^x επιστρέφει $A(g, g^x) = g^{x^{-1}}$.
Δίνουμε ως είσοδο g^x, g, οπότε παίρνουμε $A(g^x, g) = A(g^x, (g^x)^{x^{-1}}) = (g^x)^{(x^{-1})^{-1}} = (g^x)^x = g^{x^2}$, άρα λύνουμε το SDH.
Αναγωγή από IDH σε SDH
Έστω αλγόριθμος Α που λύνει το SDH, δηλαδή για είσοδο g, g^x επιστρέφει $A(g, g^x) = g^{x^2}$.
Δίνουμε ως είσοδο g^x, g, οπότε παίρνουμε $A(g^x, g) = A(g^x, (g^x)^{x^{-1}}) = (g^x)^{(x^{-1})^2} = (g^x)^{x^{-2}} = g^{x^{-1}}$, άρα λύνουμε το IDH.
Επομένως, τα προβλήματα IDH και SDH είναι υπολογιστικά ισοδύναμα.
Εφόσον το πρόβλημα SDH είναι υπολογιστικά ισοδύναμο τόσο με το πρόβλημα CDH όσο και με το πρόβλημα IDH, τότε και τα προβλήματα CDH και IDH είναι υπολογιστικά ισοδύναμα μεταξύ τους.

Άσκηση 3

Η υλοποίηση, που υπάρχει στα συνημμένα αρχεία, περιλαμβάνει το πρόγραμμα oracle.go, το οποίο πρέπει να εκτελεστεί αρχικά με την εντολή go run oracle.go και αποτελεί ουσιαστικά τον server, όπου στην αρχή επιλέγει ένα κλειδί RSA και μπορεί να επιστρέψει το Public Key που έχει δημιουργήσει, είτε να επιστρέψει το αποτέλεσμα

της συνάρτησης loc για το cipher, που δίνεται ως όρισμα στο RPC call. Το πρόγραμμα αυτό χρησιμοποιείται ως το oracle. Σημειώνεται ότι το κλειδί επιλέγεται μία φορά στην εκκίνηση του προγράμματος, οπότε για δοκιμή του προγράμματος με διαφορετικά κλειδιά πρέπει να γίνει επανεκκίνηση του προγράμματος. Επίσης, η παράμετρος ασφαλείας έχει καθοριστεί στην τιμή 2048 και μπορεί να αλλαχθεί από τη γραμμή 80 του κώδικα.
Το πρόγραμμα attacker.go προσομοιώνει τον επιτιθέμενο. Εκτελείται με την εντολή go run attacker.go και προϋποθέτει να εκτελείται το πρόγραμμα oracle.go. Αρχικά, επιλέγεται ένα μήνυμα m. Στη υποβληθείσα υλοποίηση το μήνυμα επιλέγεται από τη γραμμή 110 του κώδικα, οπότε για δοκιμή διαφορετικού m πρέπει να αλλαχθεί από τον κώδικα. Αρχικά, τυπώνουμε στην οθόνη το αρχικό μήνυμα. Στη συνέχεια, με χρήση του δημοσίου κλειδιού, το οποίο έχουμε πάρει μέσω RPC από το oracle, βρίσκουμε την κρυπτογράφηση του m. Έπειτα, εφαρμόζουμε δυαδική αναζήτηση εφαρμόζοντας διαδοχικές ερωτήσεις στο oracle, όπως περιγράφηκε στο μάθημα. Για το cipher που γίνεται η ερώτηση προς το oracle, ο υπολογισμός γίνεται ως εξής:

$$Enc_e(2m) = (2m)^e = 2^e \cdot m^e = 2^e \cdot Enc_e(m)$$

Επομένως, σε κάθε βήμα πολλαπλασιάζουμε το cipher με το 2^e, όπου e ο εκθέτης του δημοσίου κλειδιού. Όλοι οι πολλαπλασιασμοί γίνονται modulo N. Ο παραπάνω τρόπος χρησιμοποιήθηκε, γιατί δεν προϋποθέτει τη γνώση του μηνύματος m, γιατί δεν θα είναι αυτό γνωστό σε μία ρεαλιστική επίθεση. Τέλος, τυπώνουμε και την αποκρυπτογράφηση του μηνύματος, όπως αυτή υπολογίζεται από τη δυαδική αναζήτηση.

Άσκηση 4

1. Έστω ότι γνωρίζουμε τον κύκλο $c, c^e, ..., c^{e^{t-1}}, c^{e^t} = c$, τότε ισχύει $m = c^{e^{t-1}}$.
Πράγματι,

$$\begin{aligned} c^{e^t} &= c \Rightarrow \\ (c^{e^t})^d &= c^d \Rightarrow \\ c^{e^t \cdot d} &= (m^e)^d \Rightarrow \\ c^{e^{t-1}} &= m \end{aligned}$$

Γιατί $e \cdot d \equiv 1 \mod \phi(n)$.
2. Για $c = 0, c = 1$, τετριμμένες περιπτώσεις, όπου κάθε δύναμη θα ισούται με c.
Γενικά, θα δείξουμε ότι υπάρχει δύναμη t τέτοια, ώστε $c^{e^t} = c$, ανεξάρτητα από το c, χωρίς να σημαίνει ότι το μήκος του κύκλου είναι t, γιατί μπορεί να υπάρχει μικρότερος κύκλος. Γιάυτό πρώτα θα δείξουμε ότι υπάρχει δύναμη t τέτοια, ώστε $e^t \equiv 1 \mod \lambda(n)$.
Έστω ότι δεν υπάρχει τέτοιο t, θα υπάρχουν κ, λ διαφορετικά μεταξύ τους τέτοια, ώστε :

$$\begin{aligned} e^\kappa &\neq 1 \mod \lambda(n) \\ e^\lambda &\neq 1 \mod \lambda(n) \\ e^\kappa &\equiv e^\lambda \mod \lambda(n) \end{aligned}$$

Όπου η τελευταία ισχύει γιατί οι τιμές modulo $\lambda(n)$ είναι πεπερασμένες, οπότε από αρχή του περιστερώνα κάποιες δυνάμεις θα συμπέφτουν.
Χωρίς βλάβη της γενικότητας υποθέτουμε $\kappa = \lambda + t$, με $t > 0$. Επομένως, θα είναι:

$$\begin{aligned} e^\kappa &\equiv e^\lambda \mod \lambda(n) \Rightarrow \\ (e^\kappa)^{d^\lambda} &\equiv (e^\lambda)^{d^\lambda} \mod \lambda(n) \Rightarrow \\ e^t &\equiv 1 \mod \lambda(n) \end{aligned}$$

Άρα, καταλήγουμε σε άτοπο και υπάρχει $t : e^t \equiv 1 \mod \lambda(n)$. Από διαδοχικές εφαρμογές του Θεωρήματος Euler γνωρίζουμε ότι ισχύει ακόμη $c^{\lambda(n)} \equiv 1 \mod n$, επομένως θα είναι:

$$c^{e^t} \equiv c^{k \cdot \lambda(n)+1} \equiv c \cdot (c^{\lambda(n)})^k \equiv c$$

Επομένως, υπάρχει κύκλος για κάθε c.
3. Το μήκος του RSA-κύκλου είναι το πολύ $\lambda(n)$, καθώς από το προηγούμενο ερώτημα δεδομένου ότι ενδιαφερόμαστε για πράξεις modulo $\lambda(n)$ και οι αριθμοί είναι διάφοροι του 0, θα έχουμε σύγκρουση μεταξύ δύο αριθμών

που απέχουν το πολύ $\lambda(n)$, άρα θα είναι για το μήκος του κύκλου $t \leq \lambda(n)$.
4. Εφόσον η αντιστροφή της RSA κρυπτογράφησης είναι δύσκολη, θεωρείται δύσκολη και η εύρεση RSA κύκλου, καθώς και η εύρεση του μήκους του RSA κύκλου, καθώς θα ήταν εύκολο να αντισταφεί η κρυπτογράφηση RSA, με τον τρόπο που αναφέρεται στο ερώτημα 1.

Άσκηση 5

Έστω ο παρακάτω αλγόριθμος:

Algorithm 2 DL computation using Legendre symbol

$y_0 \leftarrow g^x$
for $i = 0, 1, 2, ..., [\log_2 p]$ **do**
  if $y_i^{\frac{p-1}{2^{i+1}}} = -1$ **then**
    $x_i \leftarrow 1$
  else
    $x_i \leftarrow 0$
  end if
  $y_{i+1} \leftarrow y_i \cdot g^{-x_i 2^i}$
end for

Θα δείξουμε ότι ο παραπάνω αλγόριθμος υπολογίζει τον διακριτό λογάριθμο, δηλαδή $x = x_0 + x_1 \cdot 2 + ... + x_n \cdot 2^n$, όπου $n = [\log_p]$.
Αρχικά, για $i = 0$, είναι $y_0 = g^x$, οπότε:

$$y_0^{\frac{p-1}{2}} = (g^x)^{\frac{p-1}{2}} = (g^{\frac{p-1}{2}})^x = \left(\frac{g^x}{p}\right) = (-1)^x = (-1)^{x_0}$$

Επομένως, πράγματι η δύναμη που υπολογίσαμε είναι 1 αν και μόνο αν $x_0 = 0$, ενω $x_0 = 1$ διαφορετικά.(Ουσιαστικά είναι η διαρροή του τελευταίου bit, όπως είδαμε στο μάθημα).
Για $i > 0$ έχουμε:

$$\begin{aligned} y_i &= y_{i-1} \cdot g^{-x_{i-1}2^{i-1}} \\ &= y_{i-2} \cdot g^{-x_{i-2}2^{i-2}} \cdot g^{-x_{i-1}2^{i-1}} \\ &= ... \\ &= y_0 \cdot \cdot g^{-x_0 2^0} \cdot ... \cdot g^{-x_{i-1}2^{i-1}} \\ &= g^{x - x_0 - x_1 \cdot 2 - ... - x_{i-1} \cdot 2^{i-1}} \end{aligned}$$

Επομένως, είναι:

$$\begin{aligned} y_i^{\frac{p-1}{2^{i+1}}} &= (g^{x - x_0 - x_1 \cdot 2 - ... - x_{i-1} \cdot 2^{i-1}})^{\frac{p-1}{2^{i+1}}} \\ &= g^{\frac{x - x_0 - ... - x_{i-1} \cdot 2^{i-1}}{2^i} \cdot \frac{p-1}{2}} \\ &= (g^{\frac{p-1}{2}})^{\frac{x - x_0 - ... - x_{i-1} \cdot 2^{i-1}}{2^i}} \\ &= (-1)^{x_i} \end{aligned}$$

Άρα, πράγματι υπολογίζεται σωστά το i-οστό bit.
Επομένως, από τη διαρροή αποκαλύπτεται πλήρως ο διακριτός λογάριθμος, όμως λόγω των συνεχών υψώσεων σε δύναμη σε κάθε βήμα δεν είναι ο υπολογισμός αποδοτικός.

Άσκηση 6

1. Ορίζουμε τη συνάρτηση αποκρυπτογράφησης:

$$Dec(sk, m) = Dec(sk, (c_1, c_2)) = \log_h(c_2 \cdot c_1^{-sk^{-1}})$$

Όπου $\log_h$ η συνάρτηση διακριτού λογαρίθμου. Απόδειξη ορθότητας:

$$\begin{aligned} Dec(sk, Enc(pk, m)) &= Dec(sk, (pk^r, g^r \cdot h^m)) \\ &= \log_h(g^r \cdot h^m \cdot (pk^r)^{-sk^{-1}}) \\ &= \log_h(g^r \cdot h^m \cdot (g^{xr})^{-x^{-1}}) \\ &= \log_h(g^r \cdot h^m \cdot (g^{-xrx^{-1}})) \\ &= \log_h(g^r \cdot h^m \cdot (g^{-r})) \\ &= \log_h(h^m) \\ &= m \end{aligned}$$

2. Η παραλλαγή έχει την ιδιότητα OW-CPA:
Θα δείξουμε ότι το CDHP ανάγεται στην παραλλαγή του ElGamal. Πράγματι, αν υποθέσουμε ότι έχουμε ένα oracle που λύνει τη δοθείσα παραλλαγή και θέλουμε να υπολογίσουμε το $g^{x_1 \cdot x_2}$ από δοθέντα g^{x_1}, g^{x_2}, τότε εργαζόμαστε ως εξής:
Ρωτάμε το oracle για την αποκρυπτογράφηση του μηνύματος (g, a), για κάποιο $a \in \mathbb{G}$, για δημόσιο κλειδί το g^{x_1}. Παίρνουμε μήνυμα m τέτοιο, ώστε $a = h^m \cdot g^{(x_1)^{-1}}$, γιατί είναι $pk^r = g \Rightarrow g^{x_1 r} = g \Rightarrow r = x_1^{-1}$. Όμως, a γνωστό, αφού επιλέγεται από εμάς και m γνωστό από oracle, επομένως, υπολογίζεται το $g^{(x_1)^{-1}}$.
Στη συνέχεια, ρωτάμε το oracle για δημόσιο κλειδί το $g^{(x_1)^{-1}}$ την αποκρυπτογράφηση του (g_2^x, a), οπότε θα είναι $r = x_2 \cdot x_1$, ενώ το a επιλέγεται από την $\mathbb{G}$ ξανά. Έστω m η απάντηση που επιστρέφεται.
Τότε, έχουμε ότι $h^m \cdot g^r = a \Rightarrow g^r = a \cdot h^{-m}$, όπου όλες οι μεταβλητές του δεύτερου μέρους είναι γνωστές, άρα είναι γνωστό το g^r. Όμως, το g^r είναι το $g^{x_1 \cdot x_2}$, άρα λύσαμε το CDHP.
Η παραλλαγή έχει την ιδιότητα IND-CPA:
Έστω ότι ο A προκαλεί με δύο μηνύματα m_0, m_1, και ο $\mathbb{C}_{IND-CPA}$ επιλέγει ομοιόμορφα το m_b και επιστρέφει κρυπτογράφηση $(g^c, h^{m_b} \cdot g^b$, τότε ο A επιστρέφει την τιμή b^* και ο B εξάγει το b^*.
Σε περίπτωση που έχουμε τριάδα Diffie-Hellman, τότε έχουμε έγκυρη κρυπτογράφηση για την παραλλαγή του ElGamal επομένως ο A έχει μη αμελητέα πιθανότητα να μαντέψει σωστά (σε σχέση με το $\frac{1}{2}$), ενώ σε διαφορετική περίπτωση μαντεύει στην τύχη (με πιθανότητα επιτυχίας $\frac{1}{2}$) άρα έχουμε μη αμελητέο πλεονέκτημα.
Η παραλλαγή δεν έχει την ιδιότητα IND-CCA: Έστω ότι ο A μπορεί να αποκρυπτογραφήσει μηνύματα της επιλογής του, εκτός του $c = (G, M) = (pk^r, h^{m_b} g^r)$.
Κατασκευάζουμε κρυπτοκείμενο $c' = (G', M') = (G \cdot pk^{r'}, M^a \cdot g^{r'}) = (pk^{r+r'}, h^{a \cdot m_b} g^{r+r'})$, όπου $a \in \{1, ..., q-1\}$ επιλέγεται από τον A.
Από την αποκρυπτογράφηση του c' βρίσκουμε το $a \cdot m_b$, επομένως $m_b = a^{-1} \cdot Dec(c')$, αφού a και μπορούμε να βρούμε τον αντίστροφο του, ο οποίος υπάρχει αφού η ομάδα έχει τάξη πρώτο.
Αν $m_b = m_0$, τότε $b^* = 0$, αλλιώς $b^* = 1$.

Άσκηση 7

Το Π διαθέτει πληρότητα
Εφόσον ο prover γνωρίζει τα m, r και επιλέγει τα s_1, s_2 σύμφωνα με το πρωτόκολλο, τότε ο verifier θα αποδέχεται, αφού θα είναι:

$$\begin{aligned} g^{s_1} \cdot h^{s_2} &= g^{t_1 + em} \cdot h^{t_2 + em} \\ &= g^{t_1} \cdot (g^m)^e \cdot h^{t_2} \cdot (h^r)^e \\ &= (g^{t_1} \cdot h^{t_2}) \cdot (g^m \cdot h^r)^e \\ &= t \cdot c^e \end{aligned}$$

Το Π διαθέτει ειδική ορθότητα
Έστω μία επανάληψη του πρωτοκόλλου για (t_1, t_2, s_1, s_2, e) και μία ακόμη εκτέλεση για (t_1, t_2, s_1', s_2').

Έχουμε $s_1^{'} - t_1 = e^{'}m \Rightarrow t_1 = s_1^{'} - e^{'} \cdot m$, οπότε θα είναι:

$$s_1 - t_1 = em \Rightarrow s_1 - s_1^{'} + e^{'} \cdot m = em \Rightarrow m = \frac{s_1 - s_1^{'}}{e - e^{'}}$$

Αντίστοιχα, ισχύει $s_2^{'} - t_2 = e^{'}r \Rightarrow t_2 = s_2^{'} - e^{'} \cdot r$, οπότε θα είναι:

$$s_2 - t_2 = er \Rightarrow s_2 - s_2^{'} + e^{'} \cdot r = er \Rightarrow r = \frac{s_2 - s_2^{'}}{e - e^{'}}$$

Το Π διαθέτει μηδενική γνώση για τίμιους επαληθευτές
Αρχικά, Sim δεσμεύεται στο $t = g^{t_1} \cdot h^{t_2}, t_1, t_2 \leftarrow \$\mathbb{Z}_q^*$.
Ο verifier επιλέγει ομοιόμορφα το e και το επιστρέφει στον Sim.
Αν ο Sim μπορεί να απαντήσει (αμελητέα πιθανότητα να γίνεται), τότε το πρωτόκολλο συνεχίζει, διαφορετικά έχουμε rewind.
Sim δεσμεύεται στο $t^{'} = c^{-e} \cdot t$.
Εφόσον ο verifier είναι τίμιος στέλνει ξανά το ίδιο e.
Sim στέλνει $s_1 = t_1$ και $s_2 = t_2$.
Ο verifier αποδέχεται, γιατί:

$$g^{s_1} \cdot h^{s_2} = (g^{t_1} \cdot h^{t_2} = g^{t_1} \cdot h^{t_2} \cdot c^{e^{-1}}) \cdot c^e = t^{'} \cdot c^e$$

Άρα, το Π είναι Σ-πρωτόκολλο.
3. Μετά την τροποποίηση, ο verifier πρέπει να ελέγξει τη σχέση:

$$g^{s_1} \cdot h^{s_2} = a \cdot b \cdot c^e$$

Εξακολουθούν να ισχύουν οι παραπάνω ιδιότητες με τις κατάλληλες τροποποίησεις (π.χ. για το HVZK το c^{-e} πολλαπλασιάζεται με ένα από τα a, b), άρα είναι το Π′ Σ-πρωτόκολλο.

Άσκηση 9

Επισυνάπτεται το αρχείο schnorr.py, το οποίο αρχικά δημιουργεί την υπογραφή για το μήνυμα που βρίσκεται στο αρχείο mes.txt και στη συνέχεια ελέγχει ότι η υπογραφή αυτή είναι έγκυρη.
Συγκεκριμένα, βρίσκουμε μέσω του Rabin-Miller test, δύο πρώτους p, q τέτοιους, ώστε $p = 2 \cdot q + 1$. Η χρήση safe prime έγινε για λόγους απλότητας και δεν είναι απαραίτητη για υπογραφές Schnorr. Το μήκος του q είναι 512 bits (παράμετρος ασφαλείας, γραμμή 108 στον κώδικα).
Στη συνέχεια, με τυχαίες δοκιμές βρίσκουμε στοιχείο g που είναι γεννήτορας της ομάδας $\mathbb{Z}_p^*$, οπότε το τετράγωνο του είναι γεννήτορας μιας ομάδας τάξης q, από τον τρόπο που επιλέχθηκαν.
Στη συνέχεια, παράγουμε το κλειδί, επιλέγοντας τυχαία το ιδιωτικό κλειδί και υπολογίζοντας βάσει αυτού το δημόσιο.
Έπειτα, δημιουργούμε την υπογραφή (T, s), όπου $T = g^t$, για t ομοιόμορφα επιλεγμένο και $s = (t + c \cdot sk) \mod q, c = H(T||pk||m)$.
Τέλος, επαληθεύουμε την παραπάνω υπογραφή ελέγχοντας αν $g^s = T \cdot pk^c \mod p$.

Άσκηση 10

Έστω τυχαίο μήνυμα m και δημόσιο κλειδί $y = g^x$. Τότε αντίπαλος Α φτιάχνει την παρακάτω υπογραφή:

$$Sign(m) = (m, (-H(m)) \mod p - 1, g^{-H(m)} \cdot y^{-1})$$

Όπου χρησιμοποιούνται μόνο οι δημόσιες πληροφορίες $y, m, H(\cdot)$.
Επειδή για το h ισχύει η σχέση $H(m) + x + h \equiv 0(mod\ p-1)$ θα ισχύει ότι $-H(m) \equiv x + h \mod p - 1$ και $h \equiv -H(m) - x \mod p - 1$.
Δείχνουμε ότι η παραπάνω υπογραφή περνάει τους ελέγχους. Πράγματι, για τον πρώτο έλεγχο είναι:

$$\begin{aligned} yb \equiv g^a \mod p \Leftrightarrow \\ yg^{-H(m)}y^{-1} \equiv g^{-H(m)} \mod p \Leftrightarrow \\ g^{-H(m)} \equiv g^{-H(m)} \mod p \end{aligned}$$

Επίσης, για τον δεύτερο είναι:

$$\begin{aligned} g^{H(m)}yb &\equiv 1 \mod p \Leftrightarrow \\ g^{H(m)}yg^{-H(m)}y^{-1} &\equiv 1 \mod p \Leftrightarrow \\ g^{H(m)-H(m)} \cdot y^{1-1} &\equiv 1 \mod p \Leftrightarrow \\ g^0 \cdot y^0 &\equiv 1 \mod p \end{aligned}$$

Το οποίο ισχύει, άρα η υπογραφή που δημιούργησε ο αντίπαλος είναι έγκυρη για το μήνυμα m. Επειδή το μήνυμα m είναι τυχαίο ο αντίπαλος μπορεί να δημιουργήσει εγγραφή για οποιοδήποτε μήνυμα, άρα το σχήμα δεν προστατεύει από επίθεση καθολικής πλαστογράφησης.