SAE Crypto - Défi 2 : Logarithme discret et attaque Meet-in the-Middle

Etudiez le problème du logarithme discret

Consultez par exemple : http://images.math.cnrs.fr/Le-probleme-du-logarithme-discret-en-cryptographie.html

pour une présentation claire et détaillée.

Le calcul du logarithme discret est appliqué notamment dans les protocoles de Diffie-Hellman ou El Gamal.

Dans les deux cas, la force de ces cryptosystèmes est basée sur la difficulté de trouver le logarithme discret dans un groupe cyclique, c'est-à-dire que même si nous connaissons les valeurs de g^a et g^k dans ce groupe, il est extrêmement difficile de calculer celle de g^{ak}.

Détaillez un des deux protocoles

1. Détaillez et expliquez un de ces deux protocoles et explicitez en particulier son lien avec le problème du logarithme discret.
2. Proposer le code Python correspondant
3. Pour le protocole choisi, explicitez une méthode en "force brute" possible en théorie, si on intercepte les messages chiffrés en connaissant la taille de la clef, mais difficilement exploitable en pratique.
4. Proposer le code Python correspondant

Meet-in-the-middle

Avec une méthode de chiffrement dont les algorithmes de chiffrement et de déchiffrement sont publics, la sécurité repose sur une clé secrète K, qui peut servir à la fois à chiffrer et à déchiffrer. Pour chiffrer le message M à l'aide de la clé secréte K, Alice calcule le chiffré $C = \text{encode}(K, M)$. Pour déchiffrer le message chiffré C à l'aide de la clé K, Bob calcule $M = \text{decode}(K, C)$.

Supposons qu'un attaquant à découvert M et C, mais le chiffrement ne permet pas d'en déduire la clé K. Or l'attaquant voudrait connaitre cette clé K pour pouvoir lire les prochains messages chiffrés d'Alice. II sait seulement que K est un nombre composé de n bits en binaire.

Pour découvrir K, l'attaquant décide d'essayer de chiffrer M avec toutes les valeurs possibles de la clé pour tenter de retrouver C : il tente donc une attaque par force brute (recherche exhaustive).

Puisqu'il sait que la longueur de la clé qu'il cherche est de n bits, il devra tester, dans le pire des cas, tous les nombres composés de n bits, et il y en a 2^n.

Pour avoir une meilleure sécurité, Alice à l'idée de chiffrer deux fois son message avec deux clés différentes K_1 et K_2 :

$$C = \text{encode}(K2, \text{encode}(K1, M))$$

De cette façon, elle se dit que, en supposant que K_1 et K_2 comportent toutes les deux n bits, l'attaquant devra effectuer dans le pire des cas $2^n \times 2^n = 2^{2n}$ tests pour découvrir les deux clés, ce qui augmente très très vite ...

Défi

Saurez-vous vous montrer un attaquant astucieux, et trouver un moyen pour découvrir les deux clés beaucoup plus eficacement que la méthode par force brute ?

1. Expliquez quelle trouvaille astucieuce on peut employer pour réduire le nombre de cas à étudier lors d'une recherche en force brute.
2. Appliquez cette méthode pour la recherche d'un logarithme discret.

   > Indication : Ecrivez les équations de déchiffrement, puis testez l'égalité entre un chiffré et un déchiffré

Une fois cette méthode trouvée et explicitée, appliquez la pour écrire un code Python correspondant à cette attaque dans le cas du logarithme discret.

Le travail à rendre comprendra :

- Vos explications (fichier Markdown ou ODT)
- Vos codes Python
- Une petite présentation de quelques slides présentant votre synthèse

Soyez les plus clairs et pédagogues que possible !