

Passive SSH Key Compromise via Lattices

Ferrara Justin

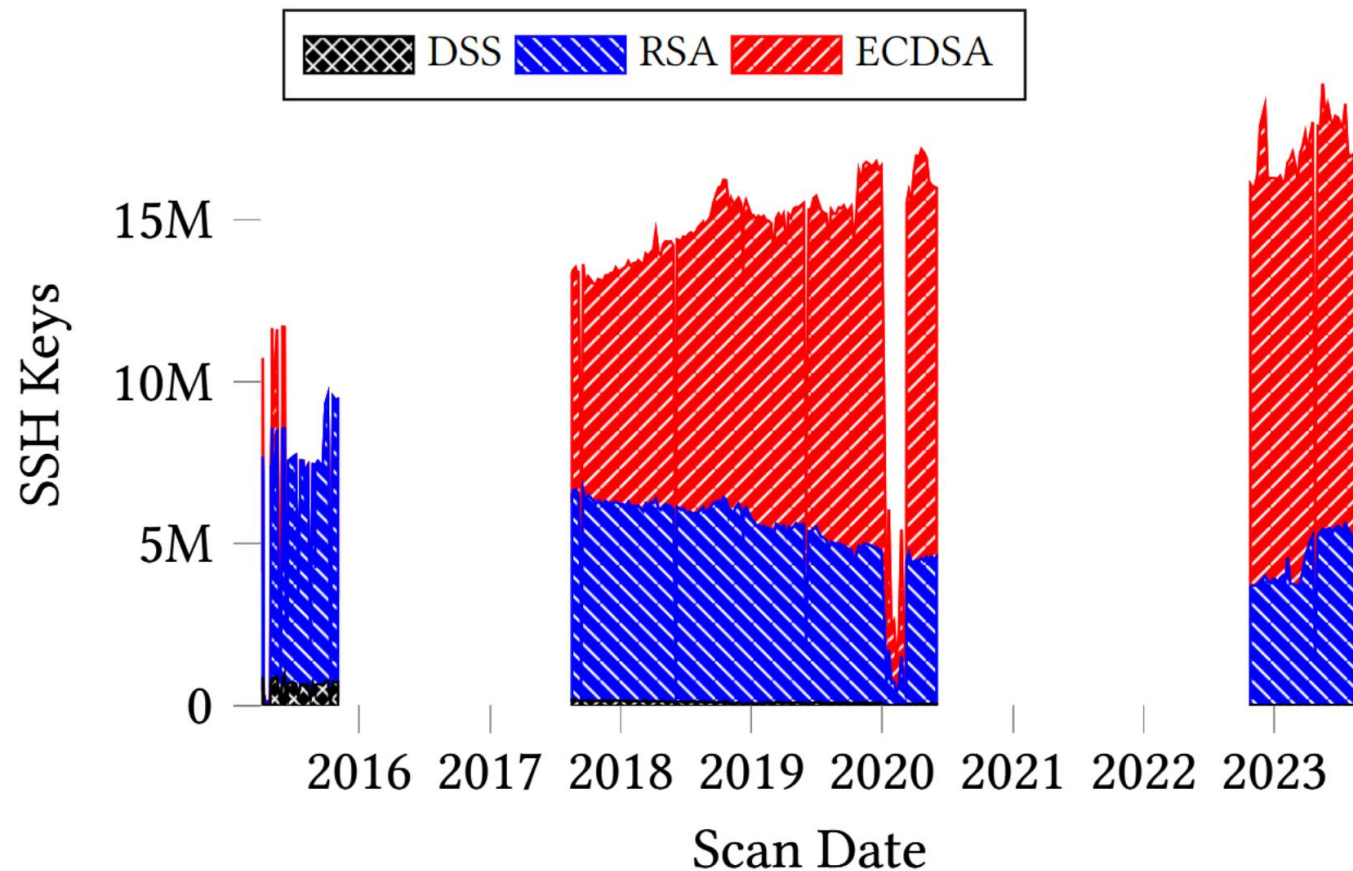
Introduction - Signatures digitales

- Signatures digitales largement répandues
- Authentifier un client, un serveur
- Souvent combiné avec d'autres algorithmes cryptographiques
- Paire de clé publique et privée

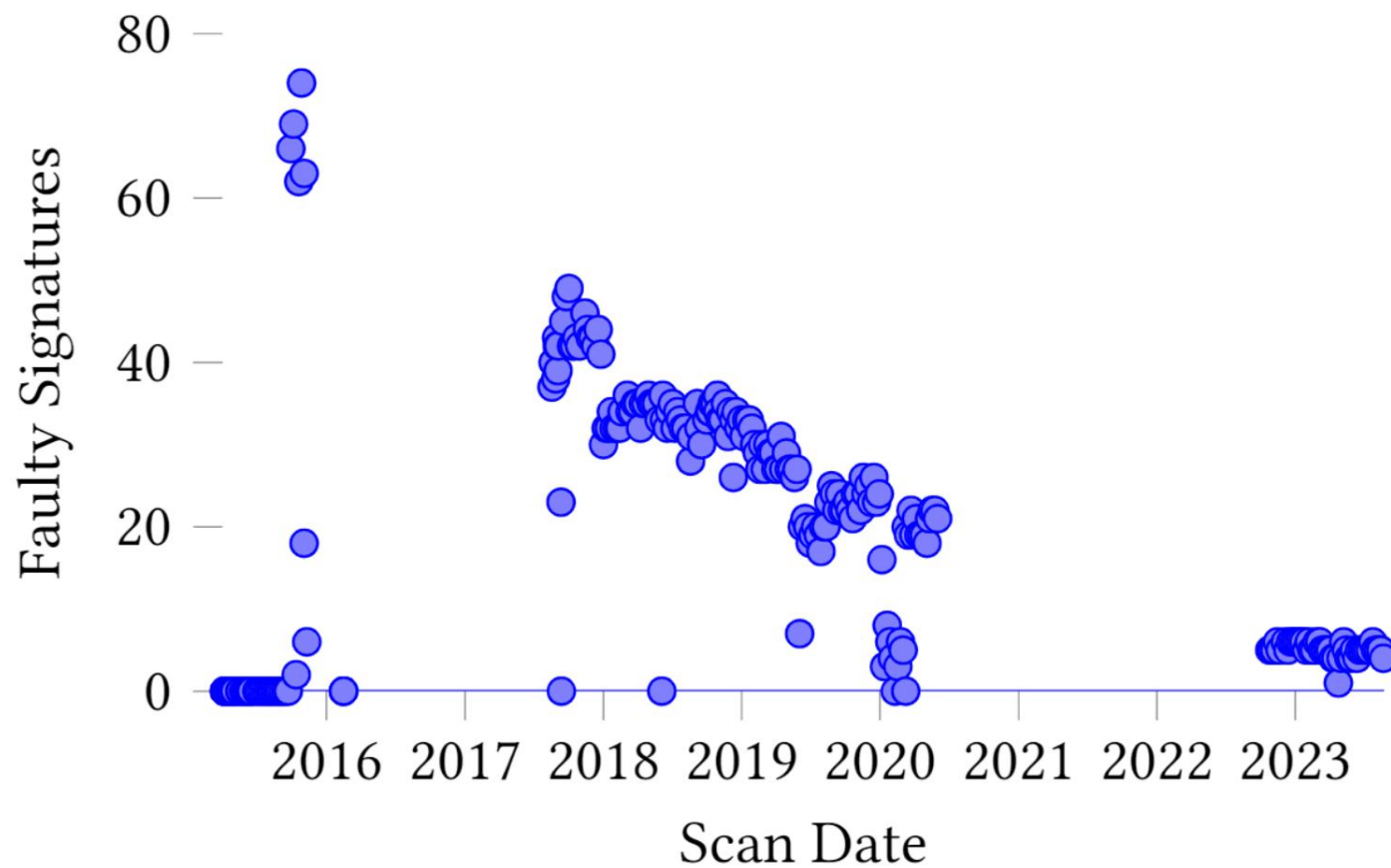
Introduction - RSA PKCS#1 v1.5

- Legacy
- Utilisé dans SSH et IPsec
- Signe un message qui est transmis de manière chiffrée
- Impossible de faire un attaque par PGCD si une erreur apparaît
- Les signatures sont calculées en utilisant le théorème des restes chinois

Algorithmes de signature dans SSH



Erreurs dans les signatures RSA



Introduction – Modèle de l'attaquant

- Une écoute passive
- Une signature contenant une erreur collectée

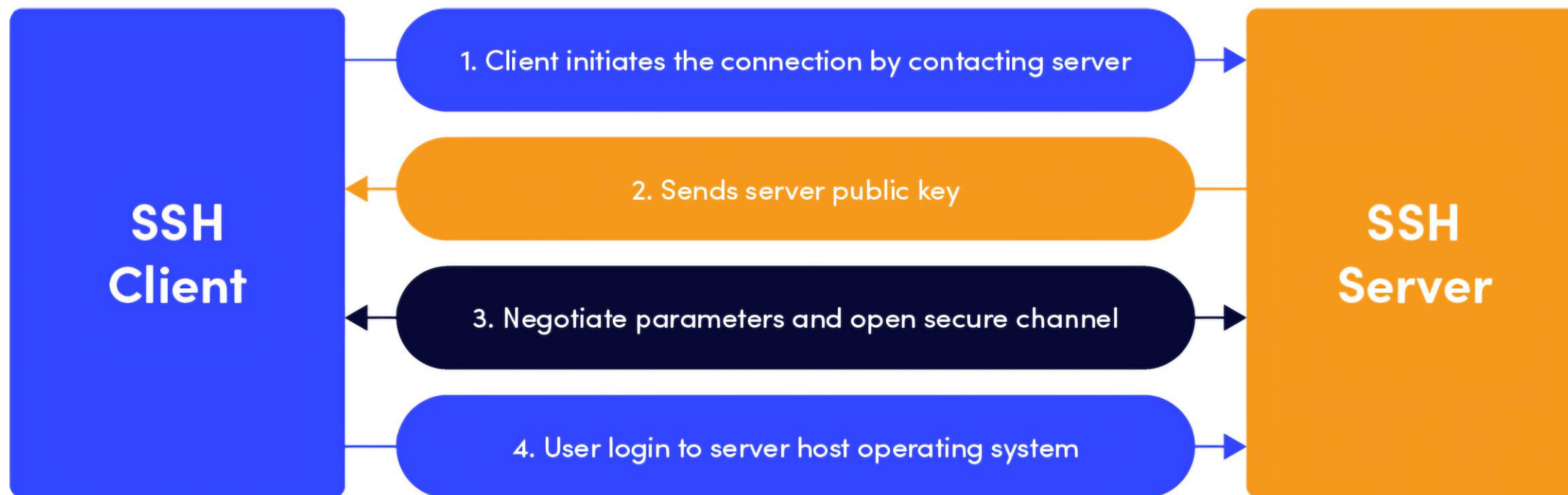
Introduction – Récapitulatif

- Écoute passive de la communication
- Protocole SSH, IPsec etc.
- Signature calculée en utilisant le théorème des restes chinois
- Une signature PKCS#1 V1.5 invalide
- Pour un message inconnu

Appareils concernés par des erreurs dans les signatures

Version SSH de l'hôte	Nombre d'erreurs
SSH-2.0-Zyxel SSH server	4705
SSH-1.99-Zyxel SSH server	168
SSH-2.0-SSHD	87
SSH-2.0-Mocana SSH 5.3.1	1
SSH-1.99-Cisco-1.25	1

SSH - Intro



SSH - Protocole

1. Connection du client
2. Négociation des algorithmes
3. Échange Diffie-Hellman
4. Le serveur signe le secret dérivé du D-H, les cipher suites, l'id client et serveur
5. Signature envoyée en clair
6. Vérification de la signature par le client
7. Établissement du canal chiffré
8. Authentification du client

SSH - Protocole

1. Connection du client
2. Négociation des algorithmes
3. Échange Diffie-Hellman
4. Le serveur **signe le secret dérivé du D-H**, les cipher suites, l'id client et serveur
5. **Signature envoyée en clair**
6. Vérification de la signature par le client
7. Établissement du canal chiffré
8. Authentification du client

SSH - Authentification du client

- Effectuée après l'établissement du canal chiffré
- Par mot de passe
 - Mot de passe envoyé dans le canal chiffré
- Par clé publique
 - Le client signe l'identifiant de session

SSH – Algorithmes disponibles

- Échange des clés :
 - D-H
 - ECDH
 - RSA
- Signatures
 - DSA
 - RSA
 - ECDSA
 - EdDSA (Ed25519)

SSH – Conséquences d'une compromission d'une clé de signature

- Attaques passives :
 - -
- Attaques actives
 - Usurper l'identité du serveur
 - Récupérer le mot de passe du client
 - Récupérer les commandes envoyées par le client
 - Man-in-the-Middle complet possible si authentication du client par mot de passe (forward du mot de passe)
 - Man-in-the-Middle complet possible si authentication par clé publique du client mais nécessite de compromettre la clé de signature du client

IPsec

- Principalement utilisé par les VPNs
- Confidentialité, authenticité et intégrité
- 2 versions majeures :
 - IKEv1
 - IKEv2

IPsec - IKEv1

- Authentification :
 - Signatures digitales
 - Par clés publiques
 - Clés pré-partagées (PSK)
- Modes :
 - Main mode
 - Aggressive mode : échange initial réduit mais moins sécurisé

IPsec - IKEv1 – Implications de sécurité

- Main mode : signatures chiffrées, attaque active nécessaire pour récupérer des signatures
- Aggressive mode : écoute passive pour récupérer la signature en clair
- Compromission de la clé de signature :
 - Usurpation de l'identité
 - Man-in-the-Middle complet possible uniquement si compromission des 2 clés de signatures (signature-based auth)

IPsec – IKEv2

- Pas compatible avec IKEv1
- Toutes les signatures sont chiffrées
- Extensible Authentication Protocol (EAP) pour obtenir une signature du serveur sans s'authentifier

IPsec – IKEv2 – Implications de sécurité

- Attaque active nécessaire pour récupérer des signatures
- Compromission de la clé de signature :
 - Usurpation de l'identité
 - Man-in-the-Middle complet possible uniquement si compromission des 2 clés de signatures (signature-based auth)

Conditions pour réussir l'attaque

- Une signature PKCS#1 V1.5 invalide
- Posséder la clé publique correspondante
- Pour un message inconnu
- Le calcul de la signature utilise le théorème des restes chinois
- L'erreur de calcul apparaît dans le monde des tuples

Signatures RSA PKCS#1 V1.5

- Signature

$$s = f(m)^d \bmod N$$

$f(x) \rightarrow$ *fonction de padding*

- Vérification

$$s^e \bmod N = f(m')$$
$$f(m) \stackrel{?}{=} f(m')$$

Expression du padding

- PKCS#1 V1.5 pour signatures

00 || 01 || FF ... FF || ASN.1 || Hash(m)

$$a + x$$

- a connu
- x inconnu

Expression du padding

- PKCS#1 V1.5 pour signatures

00 || 01 || FF ... FF || ASN.1 || Hash(m)

$$a + x$$

- a connu
- x inconnu

Théorème des restes chinois - Dédution

- Posons x et x' avec $x \neq x'$
- En sachant que :

$$\begin{aligned}x \bmod p &= x' \bmod p \\ x \bmod q &\neq x' \bmod q\end{aligned}$$

Que peut-on en déduire ?

Nous considérons donc ici une erreur de calcul dans l'anneau \mathbb{Z}_q

Théorème des restes chinois - Dédution

$$(x - x') \bmod p = 0$$

$$(x - x') \bmod q \neq 0$$

$$(x - x') \bmod N = k * p \bmod N$$

Avec x une signature valide et x' une signature invalide par exemple.

Cassez la construction !

Ce que nous connaissons

- Une signature invalide : s'
- La clé publique : (n, e)

Que pouvons-nous faire avec ces valeurs ?

Utilisation des valeurs connues

$$s'^e \bmod N = padded_message_error$$

$padded_message_error = 00 || 01 || \text{FD} \dots FF || \text{ASN.1} || \text{Hash}(m)$

$padded_message_error = 00 || 00 \dots FF || \text{ANN.1} || \text{Hash}(m)$

$padded_message_error = 00 || 01 || FF \dots FF || \text{ASN.1} || \text{Hash}(m)$

Extraction de l'erreur

$$\begin{aligned} \textit{mean_pad} &= 00 \parallel 01 \parallel \text{FF} \dots \text{FF} \parallel \text{ASN.1} \parallel 100\dots \\ \textit{padded_message_error} &= 00 \parallel 0\mathbf{0} \parallel \text{FF} \dots \text{FF} \parallel \text{ASN.1} \parallel \text{Hash}(m) \\ \\ \textit{diff} &= \textit{padded_message_error} - \textit{mean_pad} \\ \\ \textit{diff} &= 00 \parallel 0\mathbf{1} \parallel 00 \dots 00 \parallel \mathbf{XX\dots XXX} \end{aligned}$$

PACD

- Partial Approximate Common Divisors

$$N_0 = p * q$$

$$N_1 = p * q_1 + r_1$$

$$N_0 = p * q$$

$$N_1 = (s'^e - a \bmod N) = k * p + r_1$$

Lattices

$$B = \begin{pmatrix} -2^{2 \log r} & 2^{\log r} * N_1 & 0 \\ 0 & -2^{\log r} & N_1 \\ 0 & 0 & N_0 \end{pmatrix}$$

Avec r qui est la taille de l'espace de l'erreur

LLL

- Réduire la matrice
- Trouver une base plus courte
- Trouver une base plus proche de l'orthogonalité

$$reduced_matrix = LLL(B)$$

Résultat

$$\textit{reduced_matrix} = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{pmatrix}$$

$$g(x) = \frac{a_i * x^2}{2^{2 \log r}} + \frac{b_i * x}{2^{\log r}} + c_i$$

$$g(r_1) = 0$$

Résultat

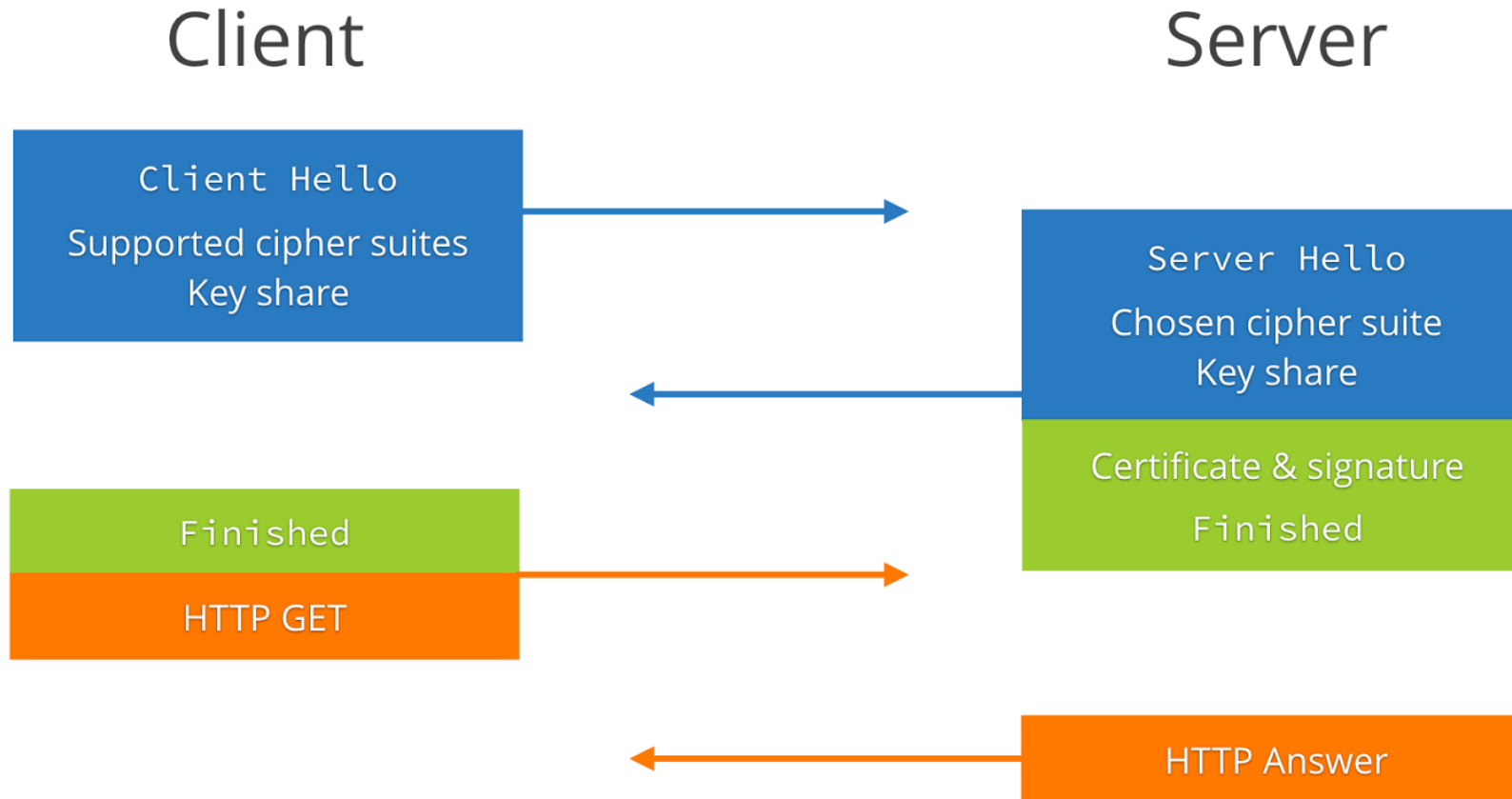
$$p = \gcd(N_0, N_1 - r_1)$$

$$q = \frac{N_0}{p}$$

Mesures de protections

- Faire les mises à jour
- Valider toutes les signatures avant de les envoyer
- Ne pas utiliser PKCS#1 v1.5
- Ne pas utiliser un padding déterministe
- Protocole :
 - Chiffrer la communication le plus tôt possible
 - Se baser sur TLS 1.3

Mesures de protections – TLS 1.3



Questions ?

Bibliographie

- **Keegan Ryan, Kaiwen He, George Arnold Sullivan, Nadia Heninger.** *Passive SSH Key Compromise via Lattices*. Cryptology ePrint Archive, Paper 2023/1711, 2023. [DOI: 10.1145/3576915.3616629](https://doi.org/10.1145/3576915.3616629), URL : <https://eprint.iacr.org/2023/1711>.
- **Duc, Alexandre, 2024.** *Asymmetric Cryptography Standards* [PDF]. Support de cours : Cryptographie avancée appliquée, HEIG-VD, 2024.
- **Tatu Ylonen.** SSH - Secure Login Connections over the Internet. Proceedings of the 6th USENIX Security Symposium, pp. 37-42, USENIX, 1996.