



420KBG

SURVOL DU PROTOCOLE OAUTH

En introduction

2

- ▶ Ma petite mésaventure de hameçonnage à l'automne 2020
 - ▶ Survol de ce qu'est le hameçonnage, ou *Phishing*

Survol (bref) du chiffrement

Survol (bref) du chiffrement

- ▶ Le chiffrement n'est pas la sécurité
- ▶ Le chiffrement nous gagne du temps
 - ▶ Il ne protège pas nos informations pour toujours

Survol (bref) du chiffrement

5

- Il y a plusieurs moyens de nuire à une communication sans comprendre son contenu



Survol (bref) du chiffrement

6

- ▶ Il y a plusieurs moyens de nuire à une communication sans comprendre son contenu
 - ▶ Supprimer un ou plusieurs paquets



Survol (bref) du chiffrement

7

- ▶ Il y a plusieurs moyens de nuire à une communication sans comprendre son contenu
 - ▶ Supprimer un ou plusieurs paquets
 - ▶ Rejouer un paquet plusieurs fois



Survol (bref) du chiffrement

8

- ▶ Il y a plusieurs moyens de nuire à une communication sans comprendre son contenu
 - ▶ Supprimer un ou plusieurs paquets
 - ▶ Rejouer un paquet plusieurs fois
 - ▶ Permuter des paquets



Survol (bref) du chiffrement

9

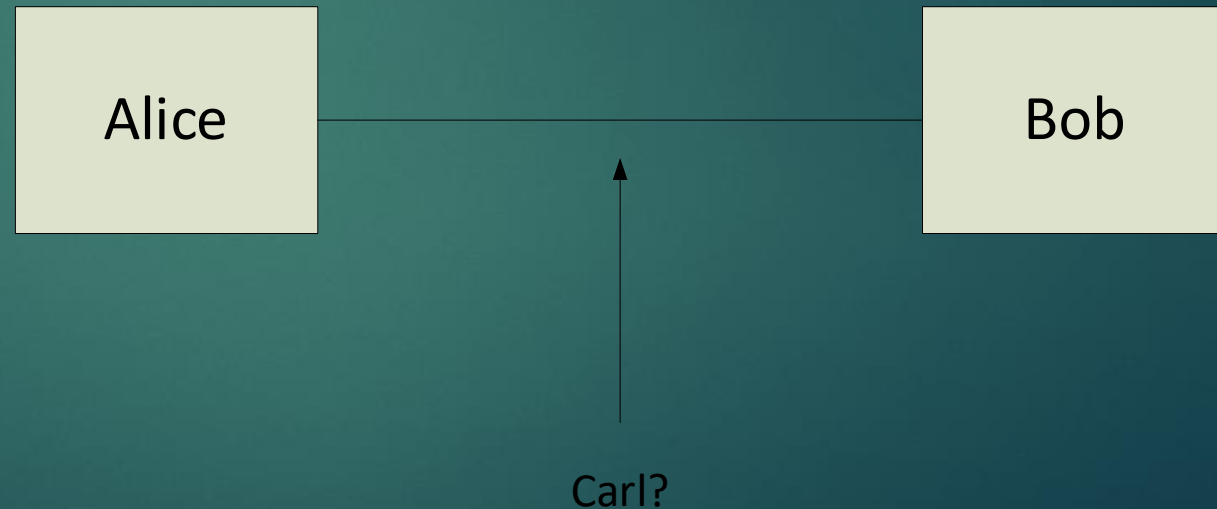
- ▶ Si Alice veut écrire à Bob de manière chiffrée, il existe plusieurs grandes familles de chiffrement



Survol (bref) du chiffrement

10

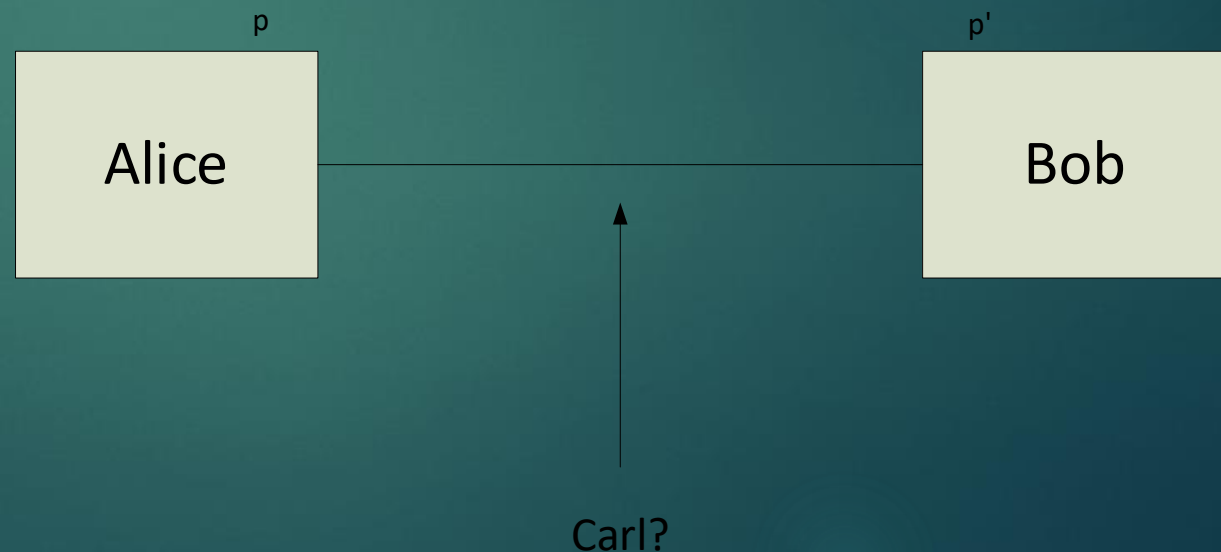
- ▶ Si Alice veut écrire à Bob de manière chiffrée, il existe plusieurs grandes familles de chiffrement
 - ▶ Utiliser une clé privée (secret partagé)
 - ▶ Risque : le partage de la clé (si Carl intercepte, brrrr...)



Survol (bref) du chiffrement

11

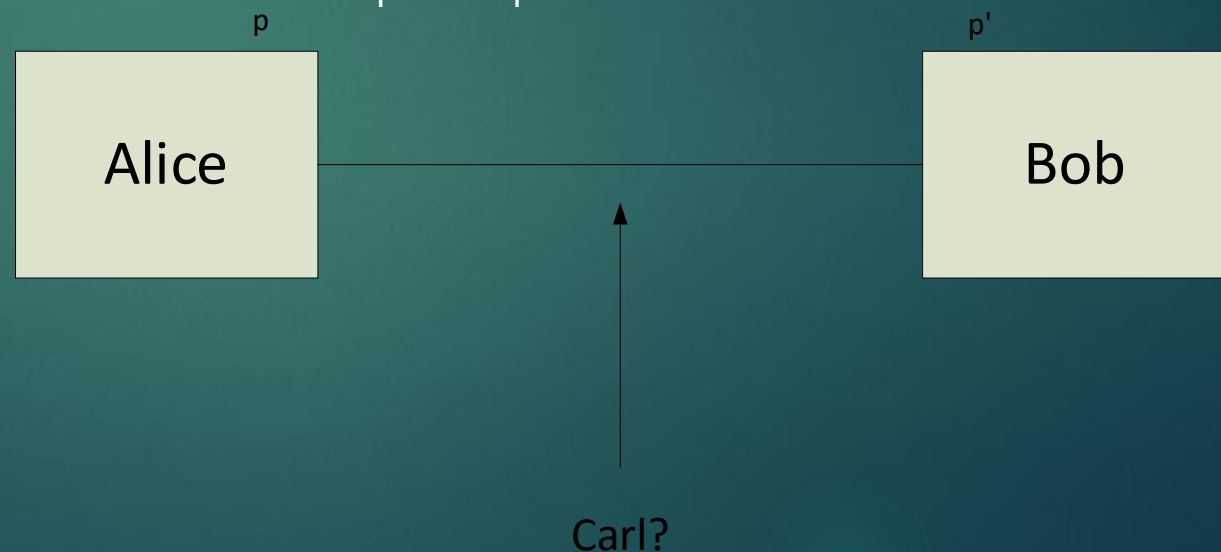
- ▶ Si Alice veut écrire à Bob de manière chiffrée, il existe plusieurs grandes familles de chiffrement
 - ▶ Utiliser une clé publique (p.ex. : RSA)
 - ▶ Alice chiffre avec la clé publique de Bob
 - ▶ Bob déchiffre avec sa clé privée



Survol (bref) du chiffrement

12

- ▶ Le chiffrement à clé publique est typiquement très coûteux en temps de calcul
 - ▶ Souvent fait avec du matériel spécialisé
- ▶ Truc :
 - ▶ Établir une communication initiale avec clé publique
 - ▶ Y passer une clé privée
 - ▶ Poursuivre avec clé privée



Survol (bref) du chiffrement

13

- ▶ Il y a d'autres avenues
 - ▶ La plus « robuste » est une clé secrète purement aléatoire, à usage unique
 - ▶ Doit être partagée par des moyens extrêmes (p.ex.: valise diplomatique)



Survol de la complexité

Survol de la complexité

15

- ▶ Pour être robuste, un chiffrement doit être difficile à briser par force brute
 - ▶ On veut des algorithmes pour lesquels la complexité algorithmique soit très élevée
- ▶ Par « complexité » on entend la manière dont croît le temps de calcul en fonction du nombre d'échantillons à traiter
 - ▶ On utilise souvent pour cela la notation « Grand-O », pour « ordre de grandeur »

Survol de la complexité

16

► Quelques exemples :

Survol de la complexité

17

- ▶ Quelques exemples :
 - ▶ Chercher si un élément e est dans un tableau tab non-trié
 - ▶ Meilleur cas : e est au début de tab (un seul test suffit)
 - ▶ Pire cas : e est à la fin de tab , ou n'est pas là (n tests sont requis)
 - ▶ En moyenne : e est à l'indice $n/2$
 - ▶ On parle alors de complexité linéaire, ou $O(n)$
 - ▶ Informellement, doubler n double le temps de calcul associé à l'algorithme
 - ▶ Les constantes comme $\frac{1}{2}$ dans la notation ne sont pas importantes
 - ▶ Elles disparaissent si on change de machine

Survol de la complexité

18

- ▶ Quelques exemples :
 - ▶ Chercher si un élément e est dans un tableau tab trié
 - ▶ On examine l'élément à la position $n/2$
 - ▶ Si $e < tab[n/2]$ alors il se trouve dans $[0, n/2)$
 - ▶ Si $e > tab[n/2]$ alors il se trouve dans $[n/2, n)$
 - ▶ Chaque test réduit la plage à examiner de moitié
 - ▶ On parle alors de complexité logarithmique, ou $O(\log_2 n)$
 - ▶ C'est excellent
 - ▶ $\log_2(100)$ vaut 7 alors que $\log_2(1000)$ vaut 10. C'est presque le même effort

Survol de la complexité

19

- ▶ Quelques exemples :
 - ▶ Faire la somme des entiers de 1 à n
 - ▶ Une solution naïve serait linéaire :

```
somme ← 0
```

```
cpt ← 1
```

```
tant que cpt ≤ n
```

```
    somme += cpt
```

```
    cpt++
```

Survol de la complexité

20

- ▶ Quelques exemples :
 - ▶ Faire la somme des entiers de 1 à n
 - ▶ Une solution moins naïve est nettement meilleure
 - ▶ Supposons $n = 100$. Alors, on a :
 - ▶ $1 + 2 + 3 + \dots + 98 + 99 + 100$
 - ▶ Mais :
 - ▶ $1 + 100 == 101$
 - ▶ $2 + 99 == 101$
 - ▶ $3 + 98 == 101$
 - ▶ ... on a donc $(n + 1)$, et on a cela $(n/2)$ fois
 - ▶ $(n+1)*(n/2)$ est un algorithme en temps constant, noté $O(1)$

Survol de la complexité

21

- ▶ Quelques exemples :

- ▶ Un tri à bulles est... douloureux

```
for(int i = 0; i < n-1; ++i)
    for(int j = i + 1; j < n; ++j)
        if(tab[j] < tab[i])
            permuter(tab[i],tab[j]); // en C#, passer avec ref
```

Survol de la complexité

22

- ▶ Quelques exemples :

- ▶ Un tri à bulles est... douloureux

```
for(int i = 0; i < n-1; ++i)
    for(int j = i + 1; j < n; ++j)
        if(tab[j] < tab[i])
            permuter(tab[i],tab[j]); // en C#, passer avec ref
```

- ▶ La boucle externe se fera approximativement n fois
 - ▶ La boucle interne se fera approximativement $n/2$ fois en moyenne *par itération de la boucle externe*
 - ▶ On a un algorithme quadratique, $O(n^2)$
 - ▶ Un bon tri comme le tri fusion est de complexité $O(n * \log_2 n)$

Survol de la complexité

23

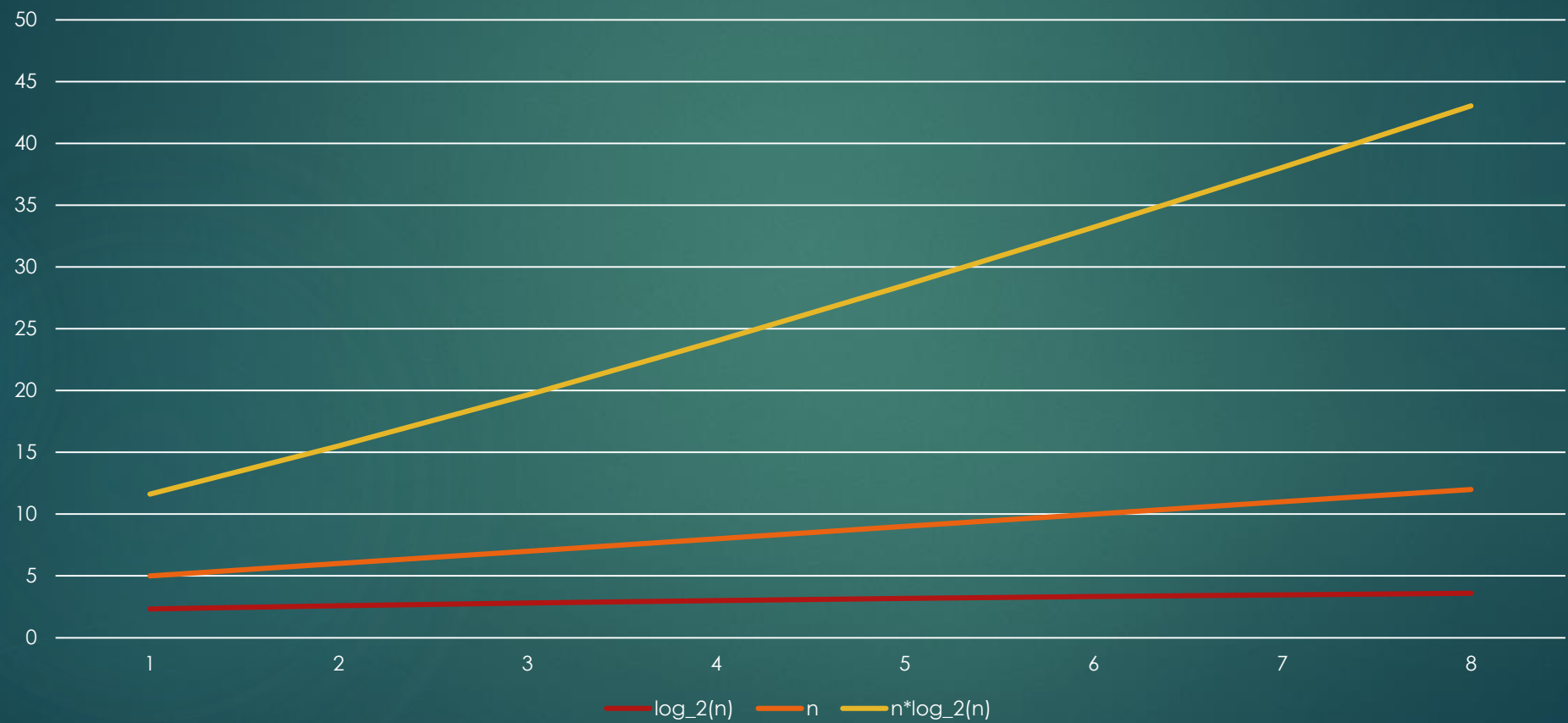
- ▶ Quelques exemples :
 - ▶ Le père Noël souhaite aller porter des cadeaux dans un village de n maisons
 - ▶ Il souhaite utiliser le plus court chemin entre elles car il est pressé
 - ▶ Initialement, il a le choix entre n maisons
 - ▶ Ensuite, il a le choix entre $(n-1)$ maisons
 - ▶ Ensuite, il a le choix entre $(n-1)$ maisons

Survol de la complexité

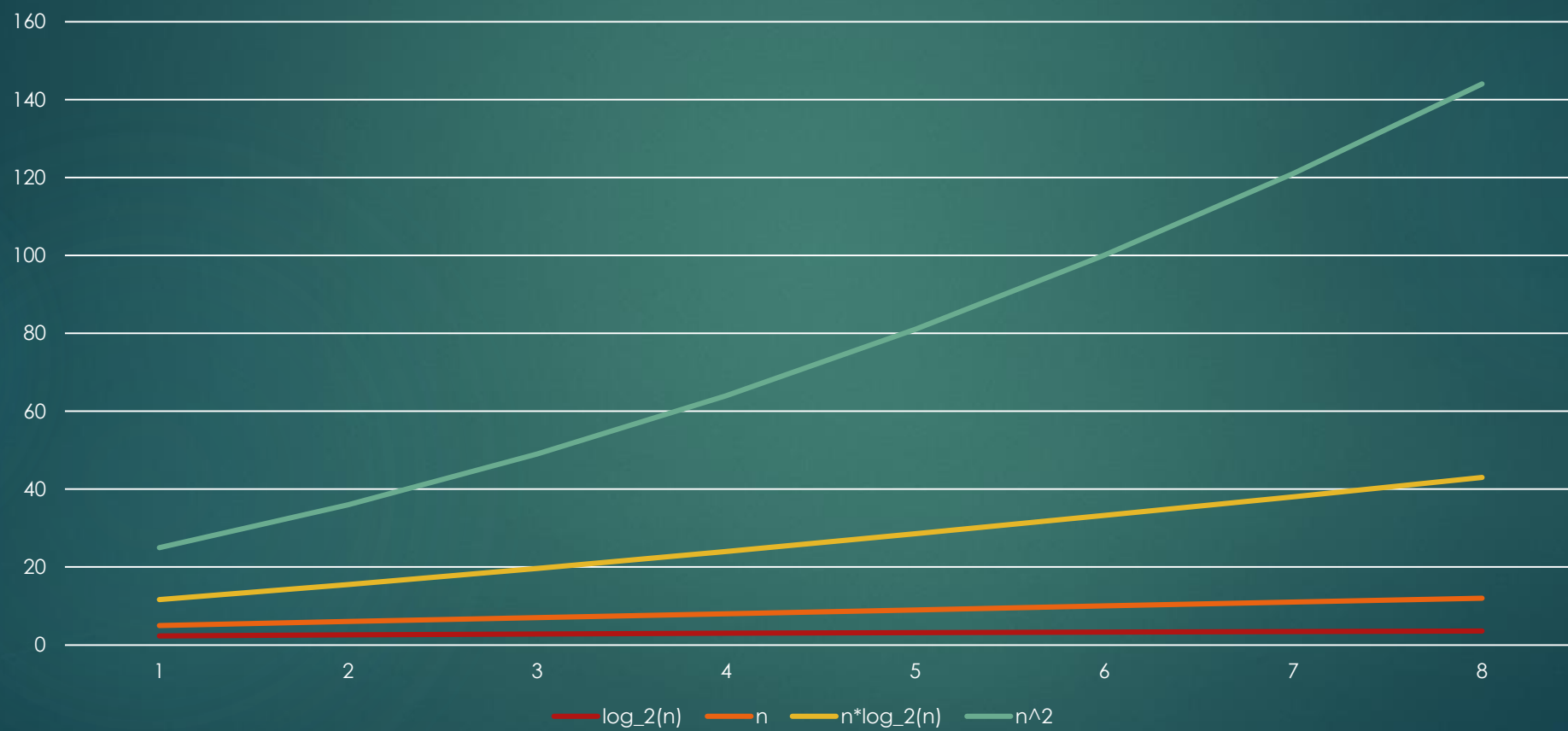
24

- ▶ Quelques exemples :
 - ▶ Le père Noël souhaite aller porter des cadeaux dans un village de n maisons
 - ▶ Il souhaite utiliser le plus court chemin entre elles car il est pressé
- ▶ ... au total, il faudra explorer $n * (n-1) * (n-2) * \dots * 2 * 1$ options, donc l'algorithme est de complexité $O(n!)$
 - ▶ C'est une catastrophe... mais c'est le genre de situation souhaitable pour le déchiffrement par force brute 😊

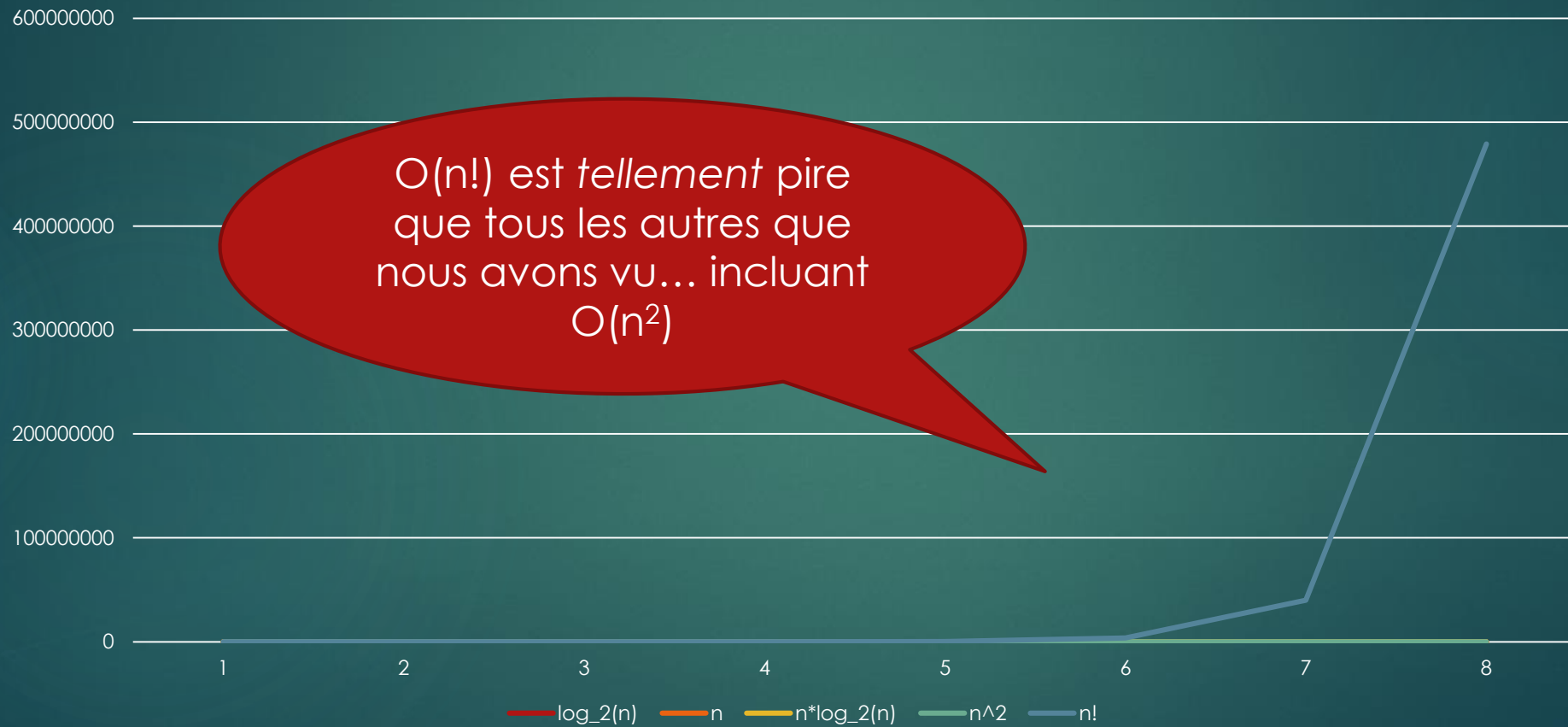
$\log_2(n)$	n	$n \cdot \log_2(n)$
2,32192809	5	11,6096405
2,5849625	6	15,509775
2,80735492	7	19,6514845
3	8	24
3,169925	9	28,529325
3,32192809	10	33,2192809
3,45943162	11	38,0537478
3,5849625	12	43,01955



$\log_2(n)$	n	$n \cdot \log_2(n)$	n^2
2,32192809	5	11,6096405	25
2,5849625	6	15,509775	36
2,80735492	7	19,6514845	49
3	8	24	64
3,169925	9	28,529325	81
3,32192809	10	33,2192809	100
3,45943162	11	38,0537478	121
3,5849625	12	43,01955	144



$\log_2(n)$	n	$n \cdot \log_2(n)$	n^2	$n!$
2,32192809	5	11,6096405	25	120
2,5849625	6	15,509775	36	720
2,80735492	7	19,6514845	49	5040
3	8	24	64	40320
3,169925	9	28,529325	81	362880
3,32192809	10	33,2192809	100	3628800
3,45943162	11	38,0537478	121	39916800
3,5849625	12	43,01955	144	479001600



Survol de OAuth

Survol de OAuth

33

- ▶ Question de vocabulaire
 - ▶ Authentification : on veut savoir si tu es celui ou celle que tu prétends être

Survol de OAuth

34

- ▶ Question de vocabulaire
 - ▶ Authentification : on veut savoir si tu es celui ou celle que tu prétends être
 - ▶ Autorisation : on veut savoir si tu as le droit de poser l'action que tu souhaites poser

Survol de OAuth

35

- ▶ Question de vocabulaire
 - ▶ Authentification : on veut savoir si tu es celui ou celle que tu prétends être
 - ▶ Autorisation : on veut savoir si tu as le droit de poser l'action que tu souhaites poser



Authentification ≠
autorisation

Survol de OAuth

36

- ▶ OAuth est un protocole d'autorisation
 - ▶ Il permet d'autoriser certaines actions, pas toutes, et de manière ponctuelle
 - ▶ C'est un peu comme avoir les clés de l'appartement d'un(e) ami(e), mais pas les clés du coffre-fort à l'intérieur

Survol de OAuth

37

- ▶ Caractéristiques :
 - ▶ Repose sur des appels à des API
 - ▶ Ne dépend pas de témoins
 - ▶ Communique avec des charges utiles JSON
- ▶ Des alternatives comme SAML prennent d'autres chemins (témoins, format XML...)

Survol de OAuth – illustration

38

- ▶ Mise en situation : soit trois intervenants
 - ▶ Usager : Bertrand
 - ▶ Consommateur : tiny.url
 - ▶ Fournisseur de service : Facebook
- ▶ La situation : le consommateur souhaite poser un geste à travers le compte du fournisseur de service appartenant à l'usager
 - ▶ Plus concrètement : tiny.url veut publier un lien raccourci sur le mure Facebook de Bertrand

Survol de OAuth – illustration

39

- ▶ Étape A : déclaration d'intention de la part de l'utilisateur
 - ▶ Bertrand informe tiny.url de son intention d'utiliser un lien raccourci sur un post Facebook
 - ▶ tiny.url doit demander la permission à Facebook

Survol de OAuth – illustration

40

- ▶ Étape B : obtention de la permission par le consommateur
 - ▶ tiny.url demande à Facebook un jeton de requête
 - ▶ Facebook offre en retour un jeton de requête et un secret
 - ▶ tiny.url utilise le secret pour chiffrer les requêtes, ce qui permettra de vérifier la provenance de ces requêtes pas la suite

Survol de OAuth – illustration

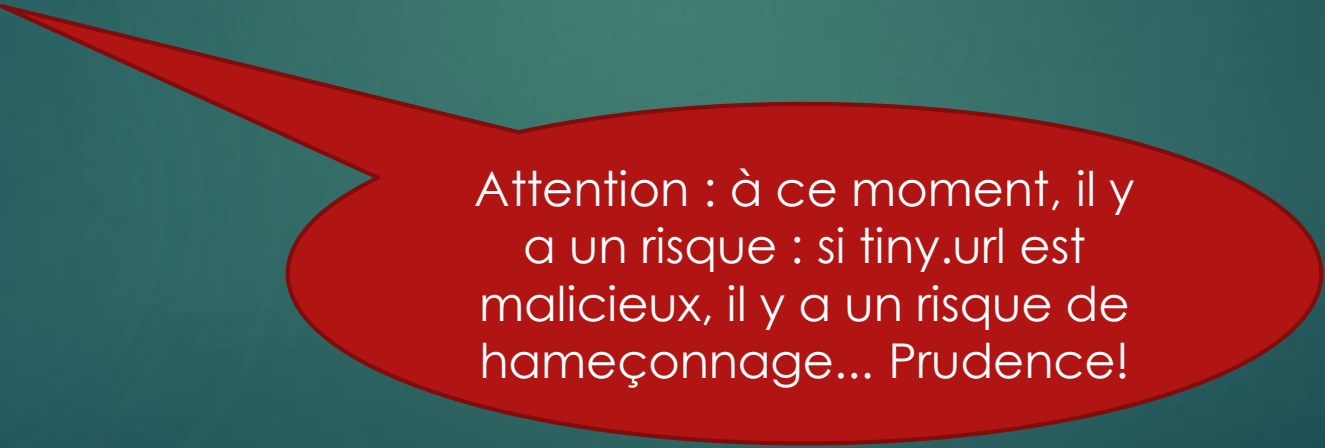
41

- ▶ Étape C : redirection de l'utilisateur vers le fournisseur de service
 - ▶ tiny.url informe Bertrand qu'il sera redirigé vers Facebook
 - ▶ Bertrand accepte
 - ▶ tiny.url dirige Bertrand vers Facebook pour fins d'authentification

Survol de OAuth – illustration

42

- ▶ Étape C : redirection de l'utilisateur vers le fournisseur de service
 - ▶ tiny.url informe Bertrand qu'il sera redirigé vers Facebook
 - ▶ Bertrand accepte
 - ▶ tiny.url dirige Bertrand vers Facebook pour fins d'authentification



Attention : à ce moment, il y a un risque : si tiny.url est malicieux, il y a un risque de hameçonnage... Prudence!

Survol de OAuth – illustration

43

- ▶ Étape D : l'utilisateur donne sa permission
 - ▶ Bertrand informe Facebook qu'il souhaite autoriser le jeton de requête offert par tiny.url
 - ▶ Facebook informe Bertrand des autorisations que cela confère à tiny.url
 - ▶ Bertrand confirme son accord
- ▶ À partir de ce moment, Facebook acceptera les requêtes de tiny.url faites pour cet utilisateur dans la mesure où elles sont chiffrées par le secret livré en début de processus

Survol de OAuth – illustration

44

- ▶ Étape E : obtention d'un jeton d'accès par le consommateur
 - ▶ tiny.url demande à Facebook d'échanger le jeton de requête pour un jeton d'accès
 - ▶ Facebook retourne un jeton d'accès et un secret

Survol de OAuth – illustration

45

- ▶ Étape F : le consommateur accède à la ressource protégée
 - ▶ tiny.url informe Facebook qu'il souhaite poster un lien raccourci et offre le jeton d'accès correctement chiffré
 - ▶ Facebook confirme que ce fut fait

Survol de OAuth – illustration

46

- ▶ Étape F : le consommateur accède à la ressource protégée
 - ▶ tiny.url informe Facebook qu'il souhaite poster un lien raccourci et offre le jeton d'accès correctement chiffré
 - ▶ Facebook confirme que ce fut fait

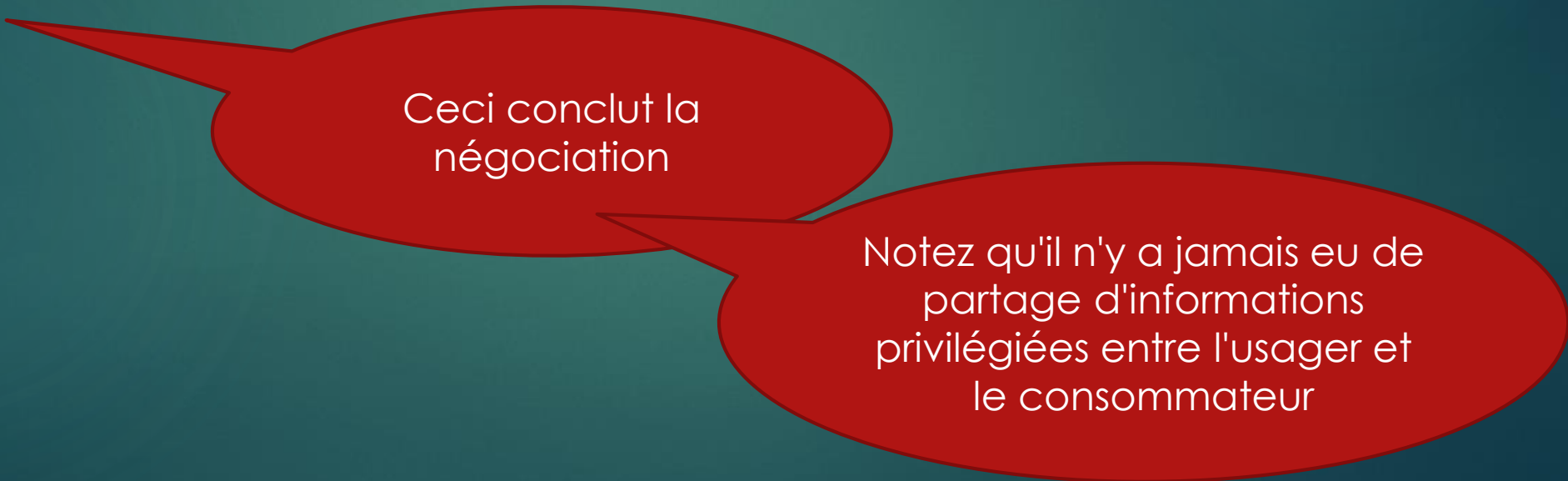


Ceci conclut la
négociation

Survol de OAuth – illustration

47

- ▶ Étape F : le consommateur accède à la ressource protégée
 - ▶ tiny.url informe Facebook qu'il souhaite poster un lien raccourci et offre le jeton d'accès correctement chiffré
 - ▶ Facebook confirme que ce fut fait



Ceci conclut la
négociation

Notez qu'il n'y a jamais eu de
partage d'informations
privilégiées entre l'usager et
le consommateur