Authentification

AB – v4.68 (02/11/2021)

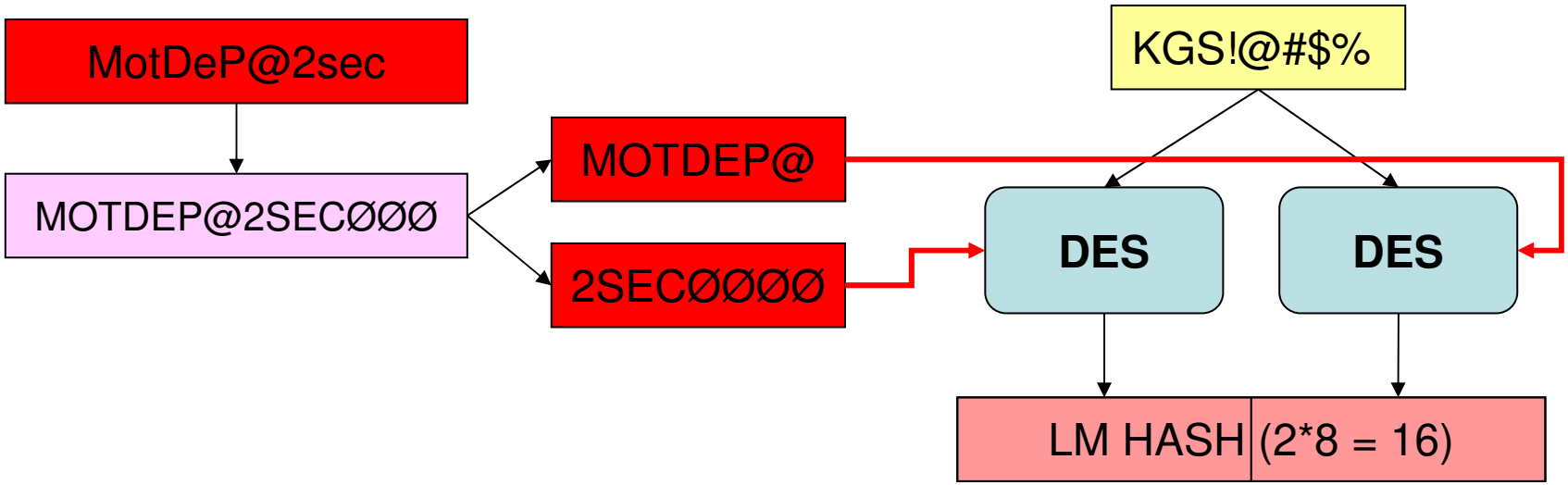
Empreinte des mots de passe

Empreinte des mots de passe

- Pour valider l'authentification d'un compte, il est nécessaire de stocker le mot de passe :
  - Soit en clair
  - Soit sous une forme non réversible sous forme d'empreinte (*hash*)

- Les deux formats « historiques », utilisés pour le calcul des empreintes des comptes locaux ou de domaine sont :
  - **Empreinte LM** (*hash LM*)
  - **Empreinte NTLM** (*hash NTLM* ou *hash NT)*

Génération du hash LM



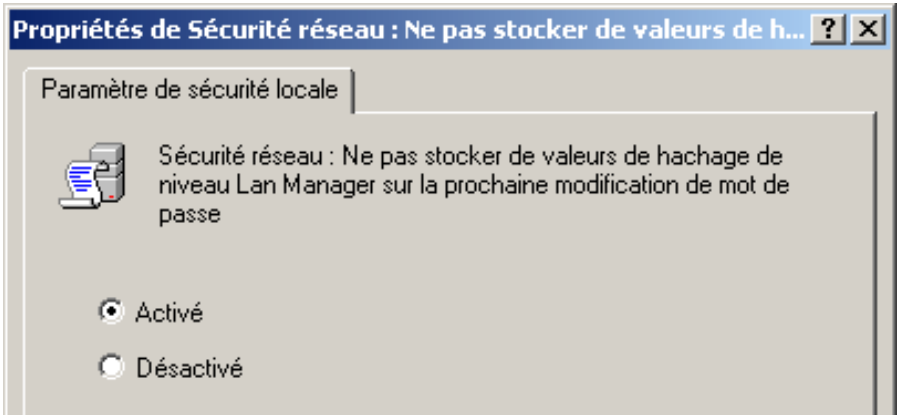
Faiblesses de LM

- Limité à 2x7 soit 14 caractères
- Alphabet réduit (`UnicodeStringToOemString`) :
  - Majuscules
  - Chiffres
  - Ponctuation
  - 32 caractères spéciaux
- Utilisation du DES

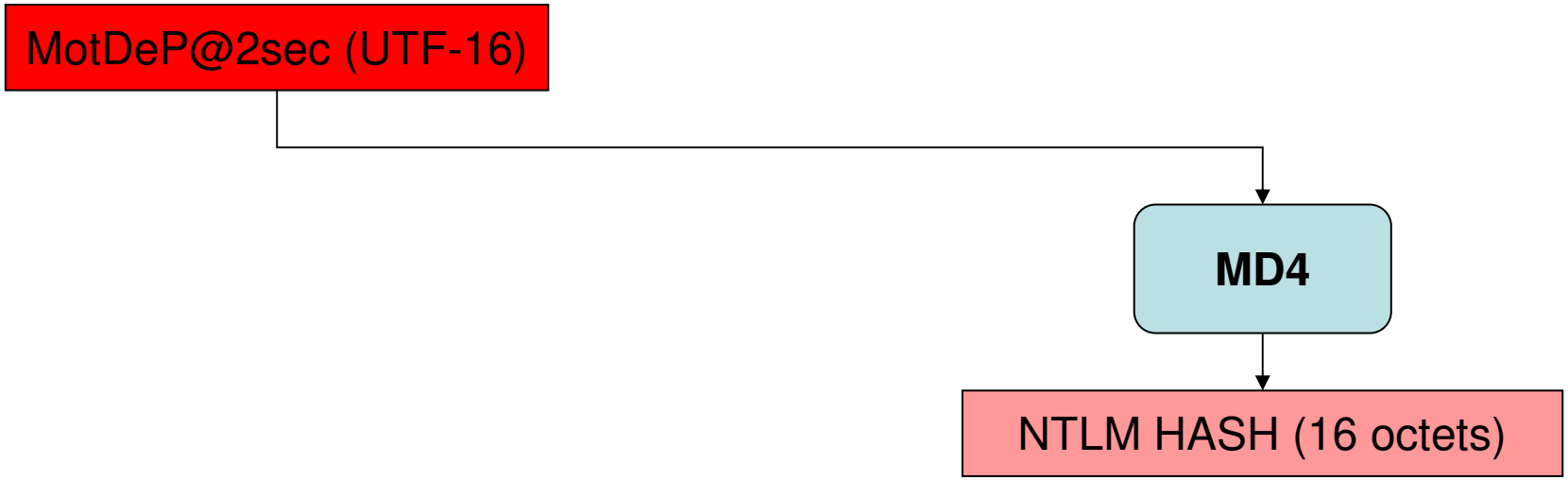
Bonnes pratiques

- Désactivation de la génération du LMHash au prochain changement (par défaut depuis Windows Vista)

- Paramétrage via GPO, clé de registre ou autre (cf. Q299656)



Génération du hash NTLM



NTLM

- 255 caractères (limité à 14 par certaines IHM)
- Alphabet complet
- Codage en du mot de passe en Unicode
- Basé sur la fonction MD4

Comparaison LM / NTLM

	LM	NTLM
Apparition	Historique	Windows NT4 SP3
Taille	14 caractères (2x7)	255 caractères
Alphabet	Restreint (OEM)	Large (Unicode)
Algorithme de calcul	DES	MD4
Graine (*salt*)	NON	NON

Problèmes

- Les empreintes LM et NTLM sont calculées sans graine
- Mots de passe identiques → Empreintes identiques (effet *master*)

- Solutions :
  - Dissocier les mots de passe sur chaque machine
    - Local Admin Password Solution (LAPS)
  - Bloquer les flux entrants
  - Réduire les droits avec UAC par le réseau

Stockage des empreintes

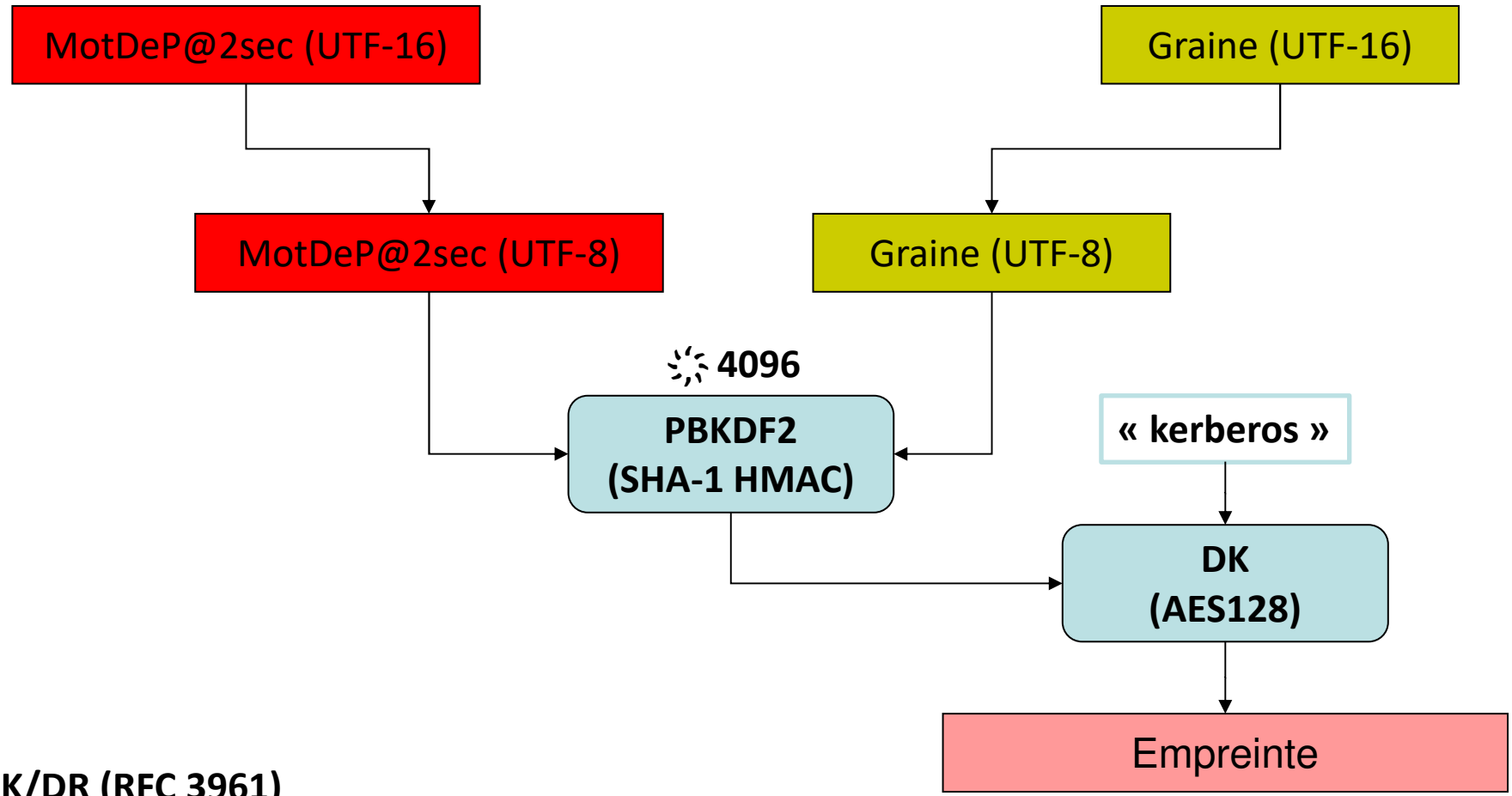
- Les empreintes LM et NTLM des comptes locaux sont stockés dans la **base SAM**

- Les empreintes LM et NTLM des comptes de domaine sont stockés dans la base **Active Directory** :
  - Attribut **`dBCSPwd`** : empreinte LM
  - Attribut **`unicodePwd`** : empreinte NTLM

Empreintes spécifiques à Kerberos (Active Directory)

- Le terme « clé » est préféré à « empreinte » pour les secrets liés à Kerberos
- L'empreinte NTLM sert de clé RC4, mais d'autres types d'empreintes, liées à Kerberos, sont également calculées :
  - **DES-CBC-MD5** (RFC3961)
  - **AES128-CTS-HMAC-SHA1-96** (RFC3962) Windows 2008
  - **AES256-CTS-HMAC-SHA1-96** (RFC3962) Windows 2008
- Les empreintes AES, calculées via PBKDF2, apportent :
  - des itérations (4096 par défaut)
  - une graine (basée sur le nom de l'utilisateur et du domaine)
- Les clés sont stockées dans l'attribut **supplementalCredentials**

AES128-CTS-HMAC-SHA1-96



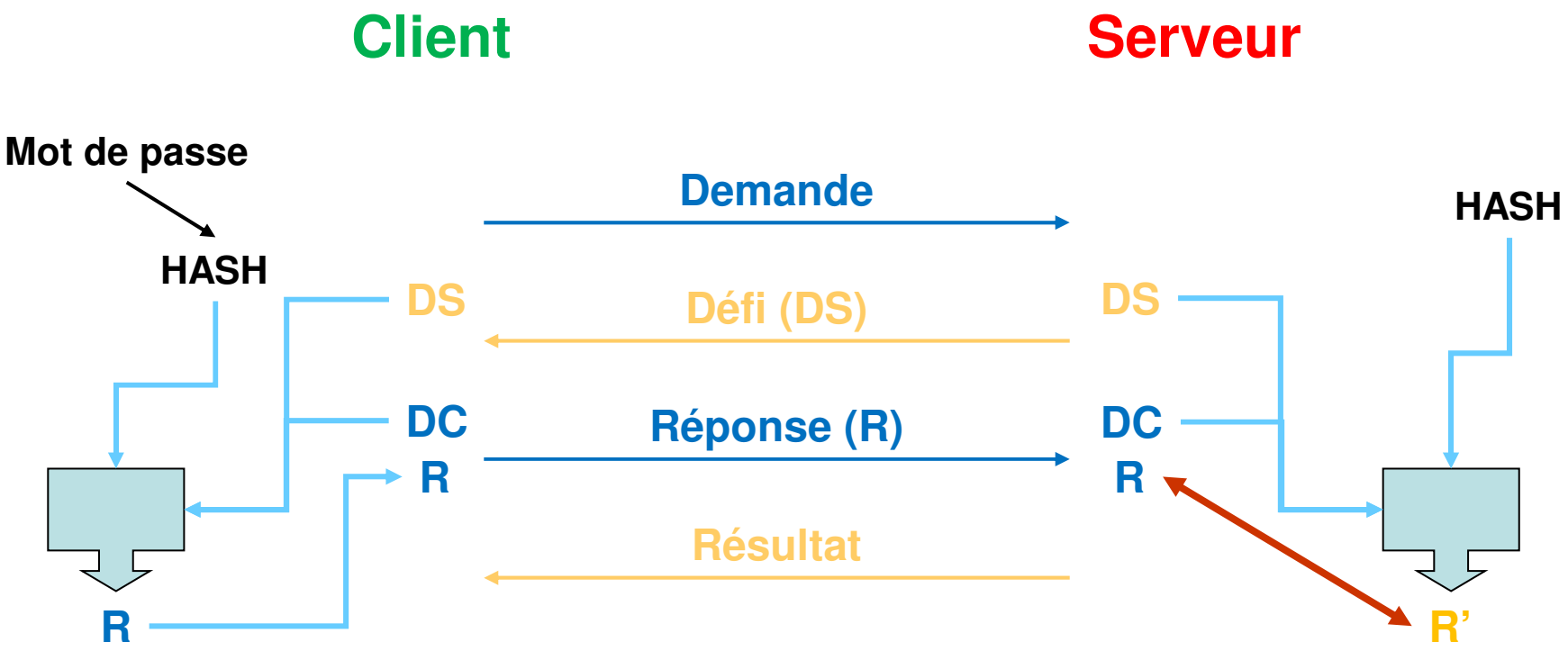
DK/DR (RFC 3961)

Authentification distante

Protocole d’authentification distante

- Un protocole d’authentification distante doit permettre :
  - d’identifier un utilisateur
  - de l’authentifier
  - de générer une clé de session entre les deux parties

Schéma général d'un défi / réponse



DS : défi serveur

DC : défi client

R : réponse client

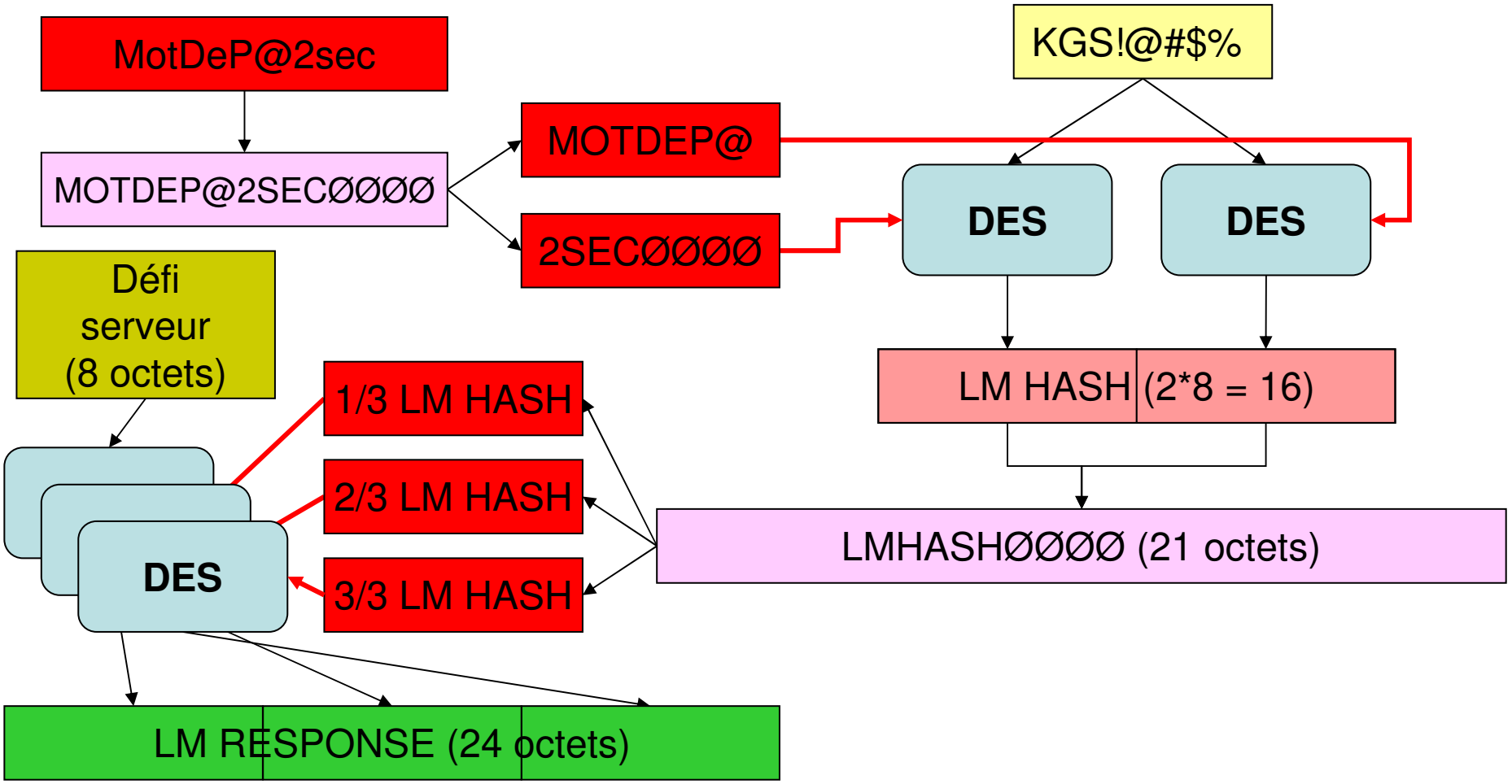
R' : réponse calculée

LM / NTLM

- Les protocoles LM et NTLM sont les protocoles historiques pour authentifier les utilisateurs à distance dans les réseaux Microsoft. Il en existe 3 générations du protocole :
  - **NTLM**
    - Réponses LM et NTLM
  - **NTLM *with Extended Session Security*** (NTLM2 *session security*)
    - Réponse NTLM
  - **NTLMv2** (NT 4 SP4)
    - Réponses LMv2 et NTLMv2

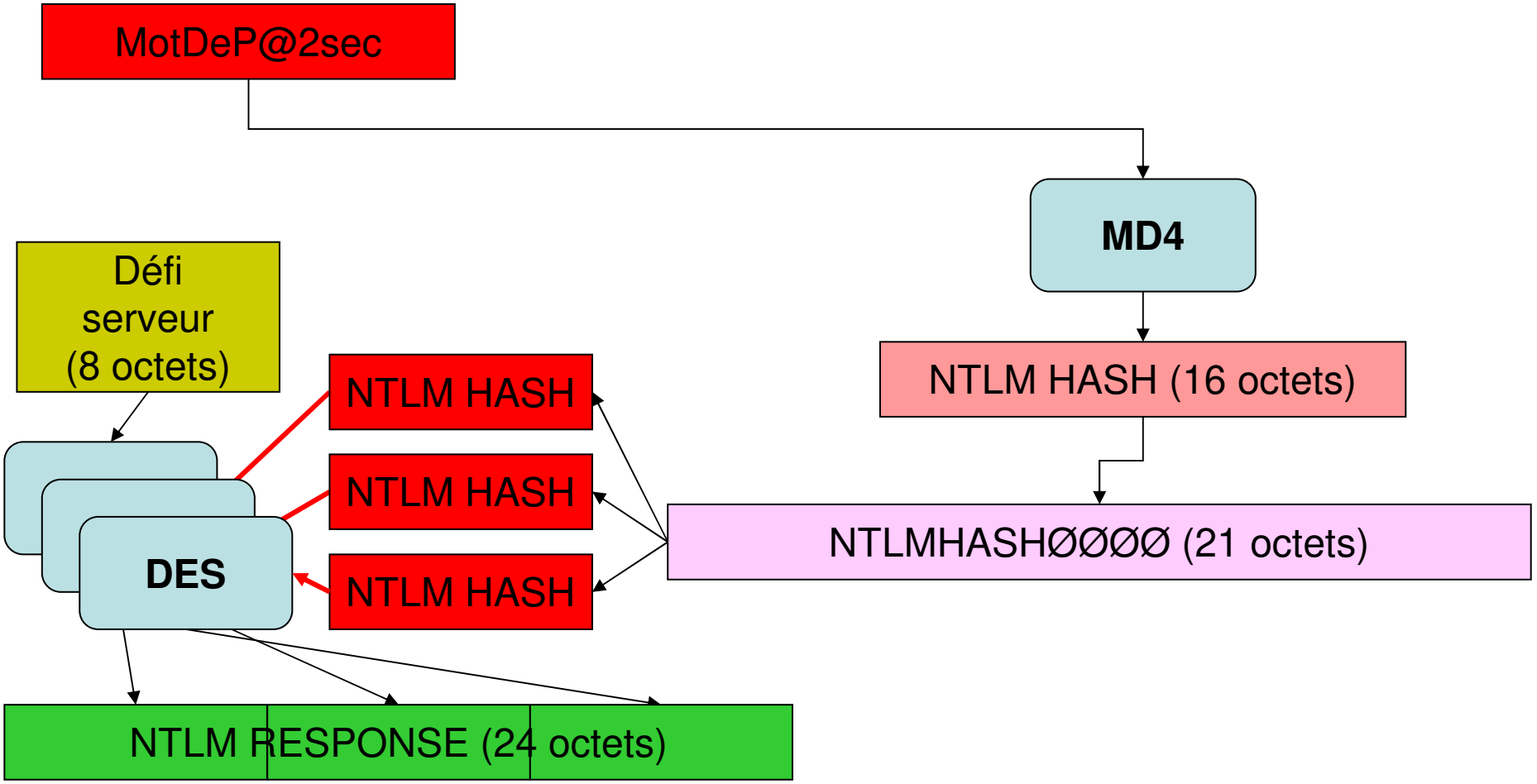
NTLM

Réponse LM



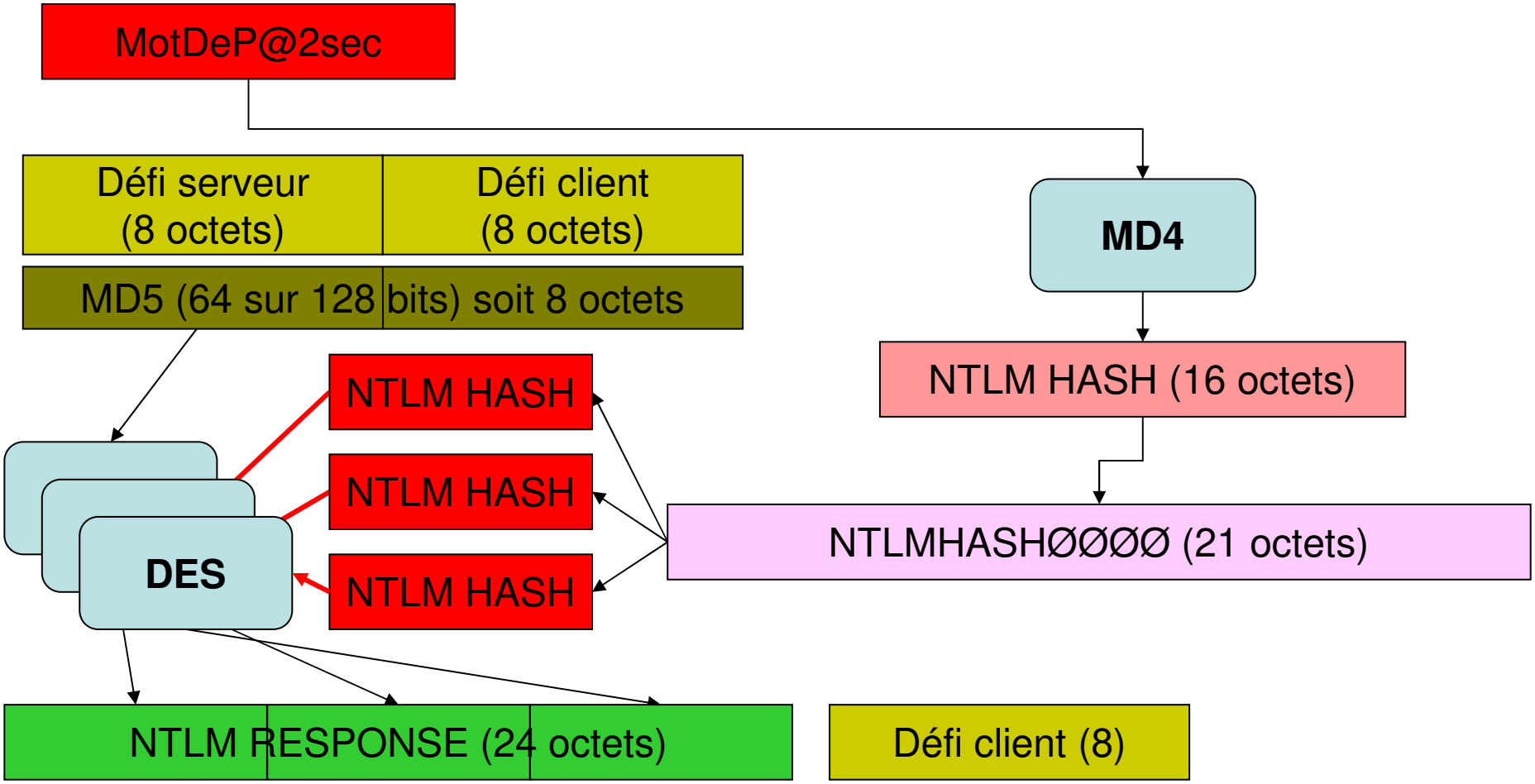
NTLM

Réponse NTLM



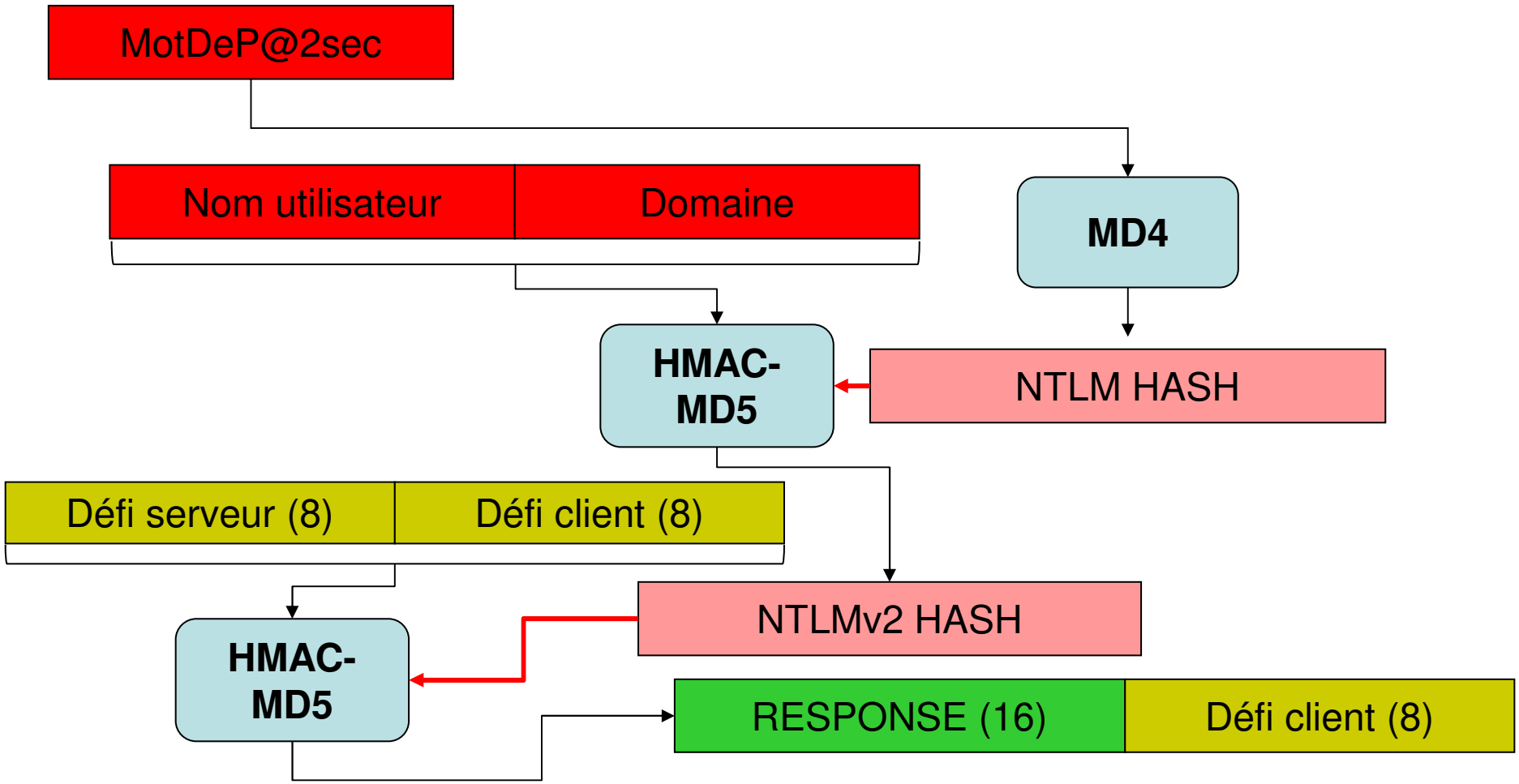
NTLM *with Extended Session Security*

Réponse NTLM



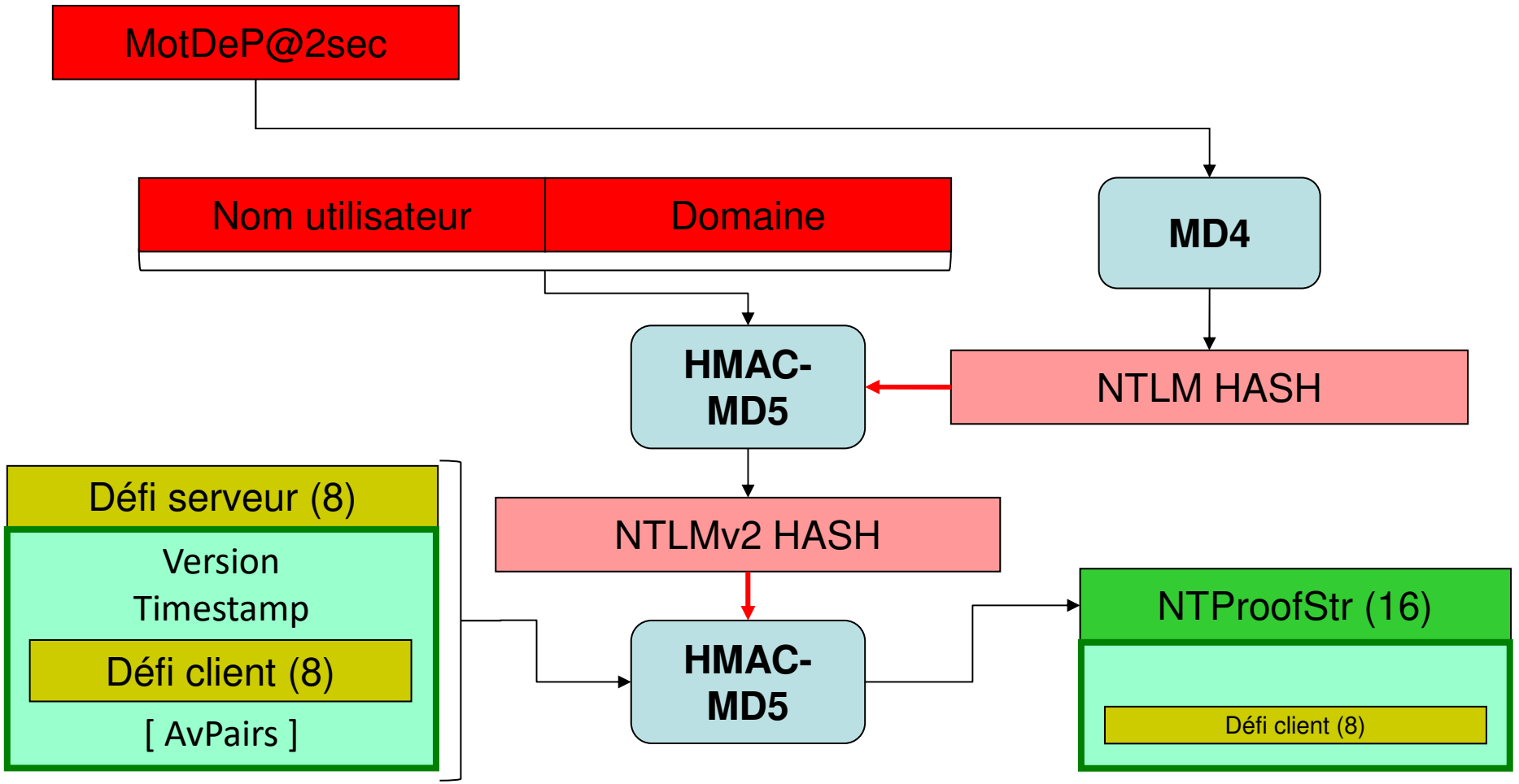
NTLMv2

Réponse LMv2



NTLMv2

Réponse NTLMv2



Comparaison des protocoles

Protocole	Réponse	Empreinte utilisée	Défi serveur	Défi client	Algorithme calcul réponse	Taille de la réponse	AvPairs
NTLM	LM	LM	OUI	NON	DES	24 octets	NON
	NTLM	NTLM	OUI	NON	DES	24 octets	NON
NTLM Extended Session Security	NTLM	NTLM	OUI	OUI	DES	24 octets	NON
NTLMv2	LMv2	NTLM	OUI	OUI	HMAC-MD5	24 octets	NON
	NTLMv2	NTLM	OUI	OUI	HMAC-MD5	>24 octets	OUI

Choix NTLM / NTLM *with* ESS / NTLMv2

- **NTLM *with Extended Session Security*** est une amélioration de NTLM
  - L'utilisation de ce protocole est négociée via le *flag* **`NTLMSSP_NEGOTIATE_EXTENDED_SESSIONSECURITY`**

- **NTLMv2** est une refonte des calculs
  - Il n'est pas négocié
  - Il doit être activé sur le client et le serveur (incapacité à s'authentifier si une des deux parties n'est pas en NTLMv2)

Protocoles utilisés

- La valeur du paramètre `LmCompatibilityLevel` détermine le protocole utilisé
- Jusqu'à Windows Server 2003, par compatibilité avec les systèmes anciens, le protocole **NTLM** est celui par défaut
- Si négocié, la variante **NTLM *with Extended Session Security*** est utilisée
- Depuis Windows Vista, c'est **NTLMv2** qui est le protocole par défaut

Choix des protocoles
Paramètre `LmCompatibilityLevel`

- La clé de registre suivante permet d'activer ou de refuser les protocoles et réponses LM, NTLM, NTLMv2 :

`HKLM\SYSTEM\CurrentControlSet\Control\Lsa\LmCompatibilityLevel`

Niveau	Client (envoie)	Serveur (accepte)	Non envoyé
0	LM, NTLM NTLMv2 (si négocié)	LM NTLM NTLMv2	NTLMv2 (2000 pré SRP1, NT4)
1	LM, NTLM NTLMv2 (si négocié)	LM NTLM NTLMv2	NTLMv2
2	NTLM NTLMv2 (si négocié)	LM NTLM NTLMv2	LM NTLMv2
3	NTLMv2	LM NTLM NTLMv2	LM NTLM
4	NTLMv2	NTLM NTLMv2	LM
5	NTLMv2	NTLMv2	LM NTLM

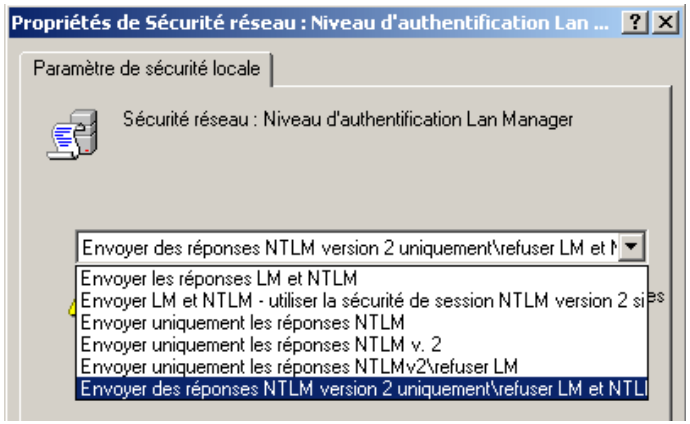
Bonnes pratiques

- Ne plus utiliser le défi/réponse LM ou NTLM pour l'authentification (niveau 3 ou +)
- Préférer l'utilisation exclusive de NTLMv2 (niveau 5)

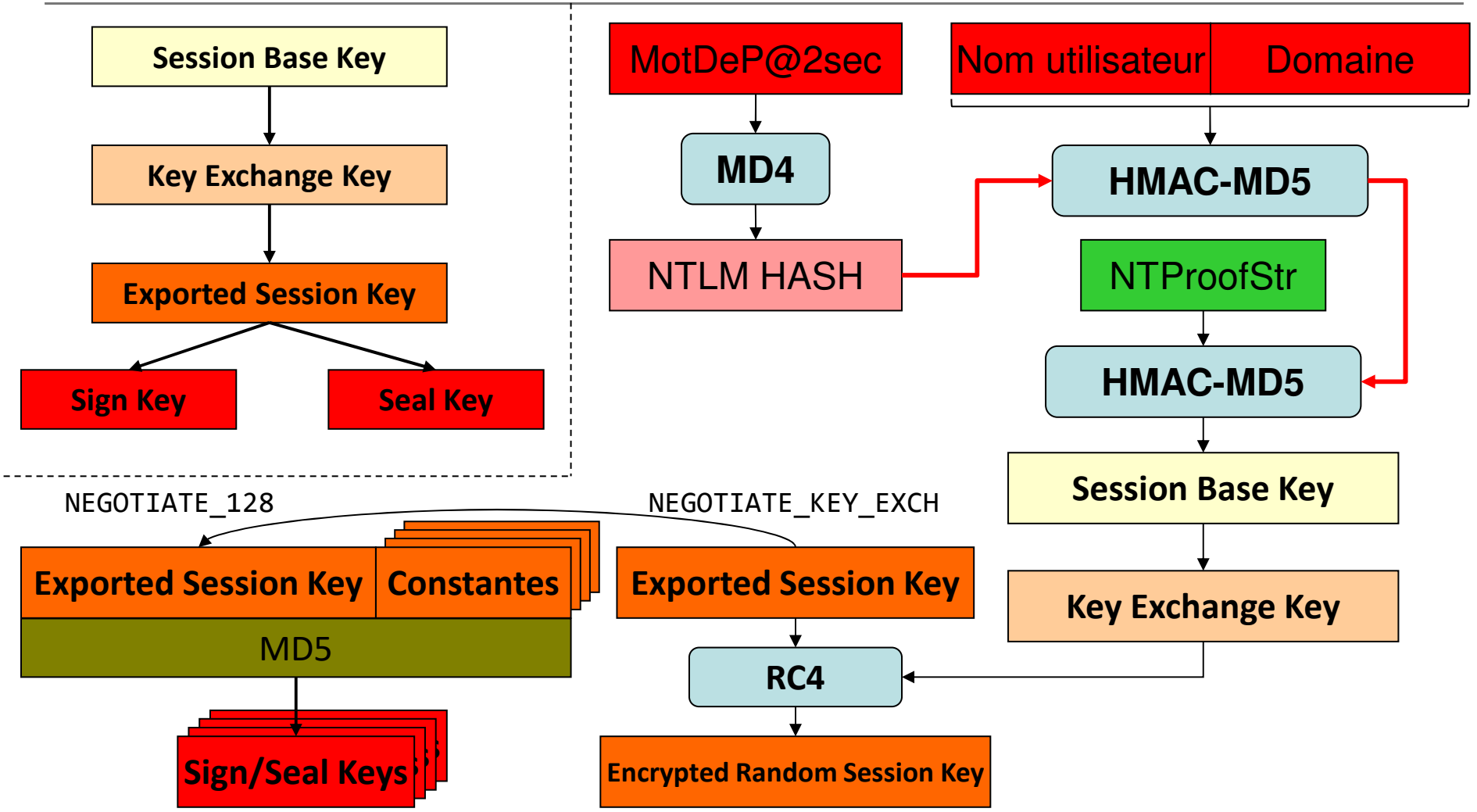
- Paramétrage via GPO ou clé de registre (cf. Q239869)

- Attention aux conséquences

  cf. Q823659



Dérivation des clés
Exemple NTLMv2



NTLMSSP

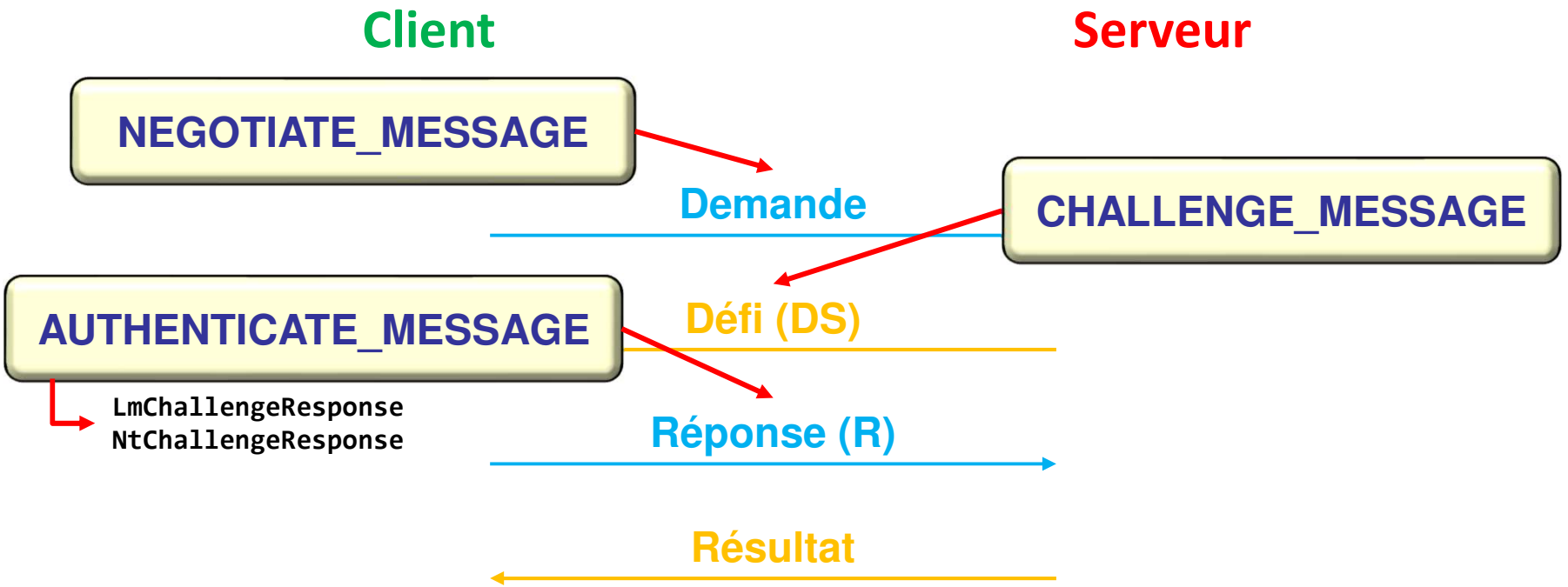
NTLMSSP

- **NTLMSSP** est un protocole réseau d'authentification distante mettant en œuvre les protocoles cryptographiques LM, NTLM, NTLMv2 (et toutes les variantes possibles)

NTLMSSP

- Chaque message commence par le marqueur « **NTLMSSP** »
- Le protocole définit trois types de messages :
  - **NEGOTIATE_MESSAGE(1)** (Client → Serveur) :
    démarre une négociation
  - **CHALLENGE_MESSAGE(2)** (Serveur → Client) :
    permet au serveur d'envoyer son défi
  - **AUTHENTICATE_MESSAGE(3)** (Client → Serveur) :
    permet au client d'envoyer ses réponses
    (champs LM et NTLM)

NTLMSSP

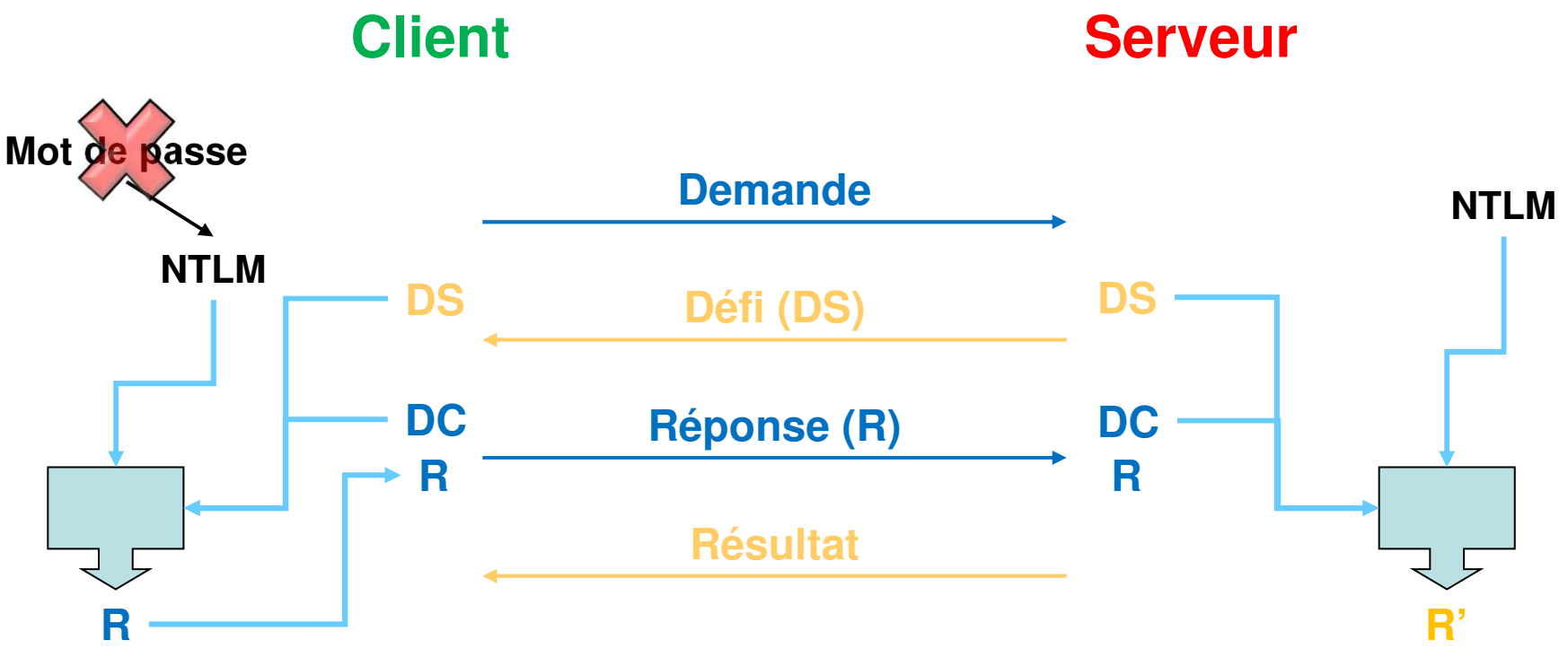

Client
Serveur
NEGOTIATE_MESSAGE
Demande
CHALLENGE_MESSAGE
AUTHENTICATE_MESSAGE
Défi (DS)
LmChallengeResponse
NtChallengeResponse
Réponse (R)
Résultat

AUTHENTICATE_MESSAGE

- NTLM :
  - **LmChallengeResponse :** LM_RESPONSE
    - **[24]** Response
  - **NtChallengeResponse :** NTLM_RESPONSE
    - **[24]** Response
- NTLM *with extended session security* :
  - **LmChallengeResponse :**
    - **[24]** : ChallengeFromClient (8) | Ø (16)
  - **NtChallengeResponse :** NTLM_RESPONSE
    - **[24]** Response
- NTLMv2 :
  - **LmChallengeResponse :** LMv2_RESPONSE
    - **[24]** : Response (16) | ChallengeFromClient (8)
  - **NtChallengeResponse :** NTLMv2_RESPONSE
    - **[>24]** Response (16) | NTLMv2_CLIENT_CHALLENGE (variable)

Faiblesses et vulnérabilités avec NTLM

Pass the hash



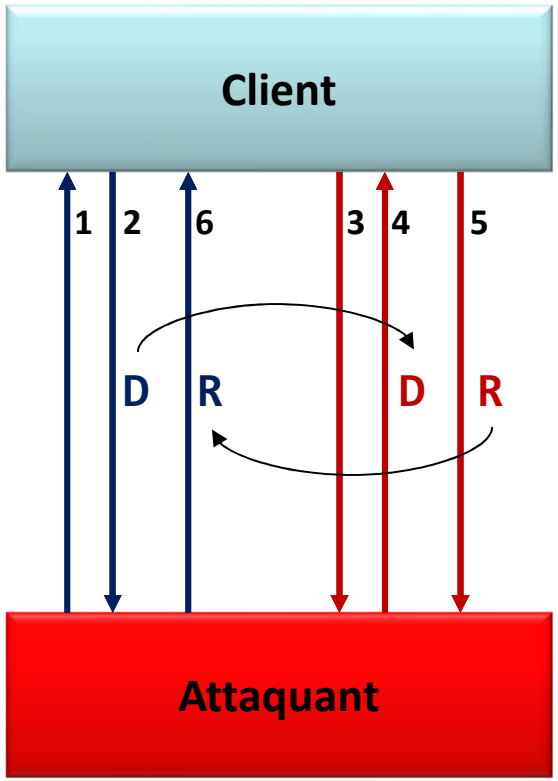
DS : défi serveur

DC : défi client

R : réponse client

R' : réponse calculée

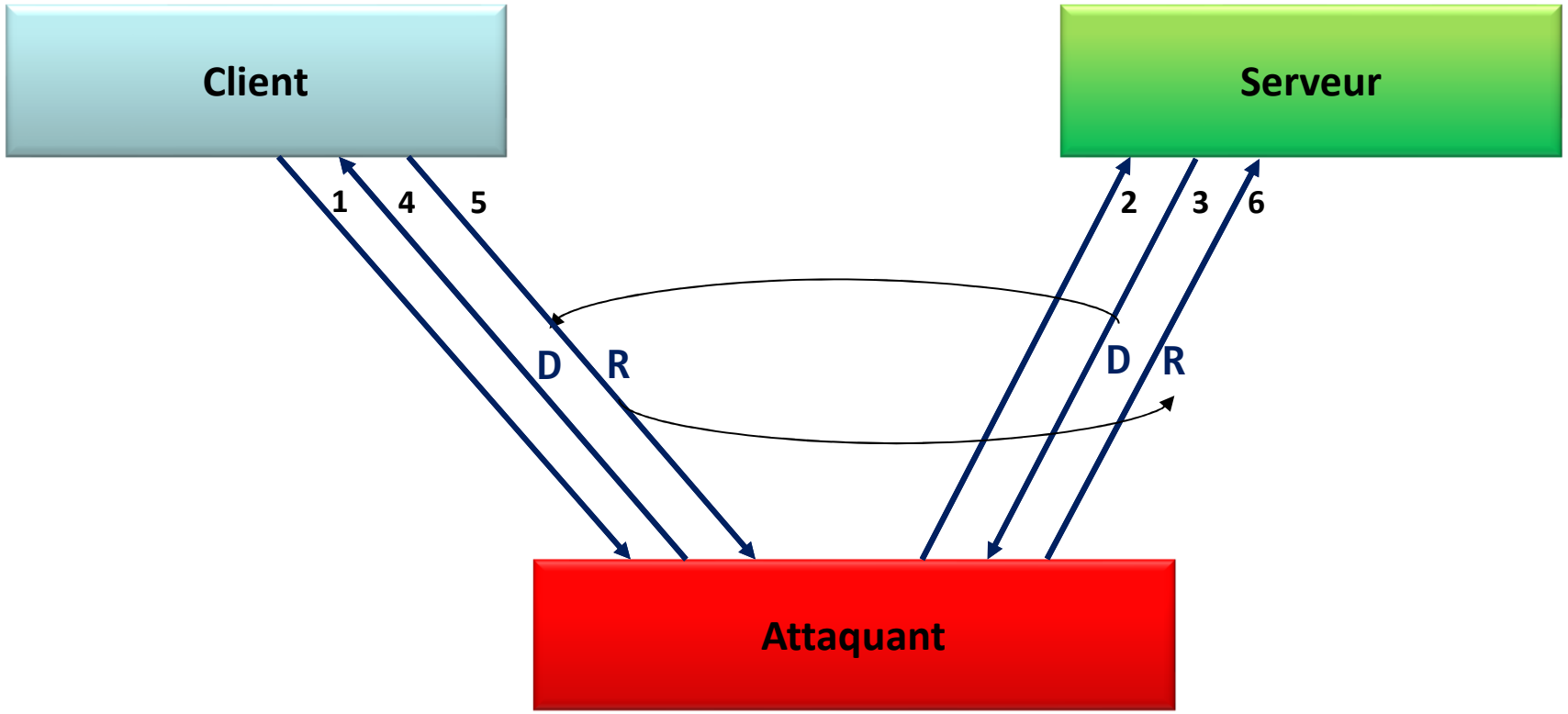
Réflexion NTLM
(*NTLM reflection*)



Protection contre la réflexion

- La réflexion NTLM peut être empêchée si le client maintient une base des défis qu'il émet
- Ces défis doivent alors être refusés s'ils sont soumis à ce même client
- Protection apportée par les correctifs MS08-068, MS08-076, MS09-013, MS09-014 et MS09-042

Relais NTLM
(*NTLM Credentials Forwarding*)



Protection contre le relais

- Il n’est pas possible, avec NTLM, de se protéger contre les attaques de type relais NTLM
- La seule parade consiste, pour les protocoles qui mettent en œuvre NTLM, à utiliser les secrets dérivés de l’authentification (signature, chiffrement ou utilisation de la clé de session)
- C’est notamment le cas pour de nombreux services Windows (SMB, RPC, etc.)

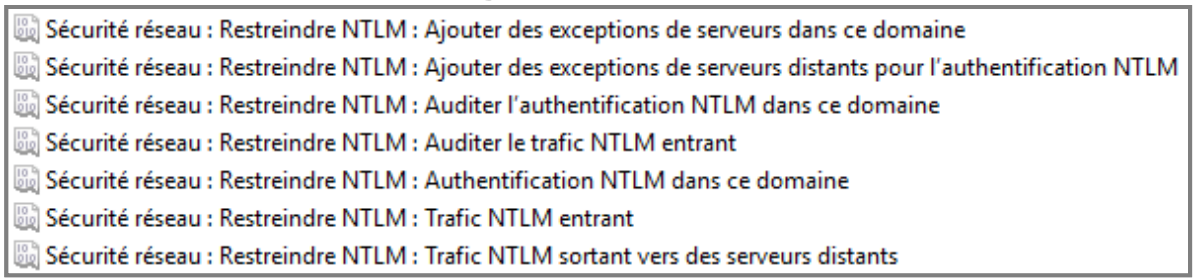
Authentification mutuelle

- NTLM ne peut structurellement pas assurer correctement l'authentification mutuelle
- Le client n'est donc pas assuré qu'il s'est authentifié sur le serveur légitime

Désactivation de NTLM

Désactivation de NTLM

- Depuis Windows 7/2008R2, il est possible de désactiver NTLM sur un système via la stratégie de sécurité locale



- Cependant, il est recommandé d'auditer préalablement l'utilisation de NTLM via les fournisseurs d'évènements :
  - LsaSrv / 6038, 6039
  - Microsoft-Windows-NTLM
- L'appartenance au groupe « *Protected Users* » dans l'Active Directory permet également de prohiber l'utilisation de NTLM

NetLogon

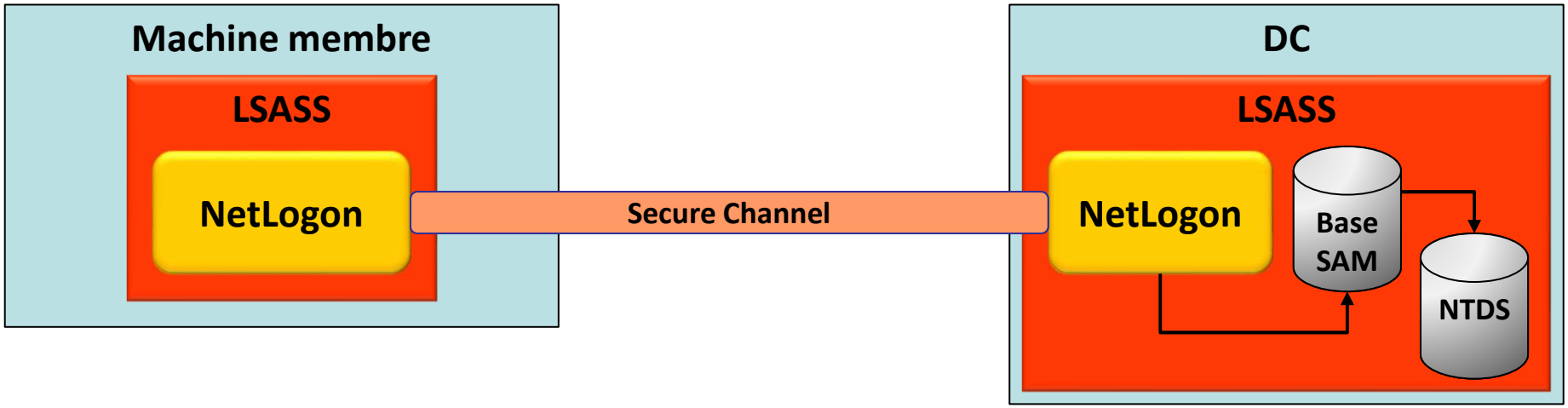
NetLogon

- Si un utilisateur s'authentifie avec NTLMSSP auprès d'une machine avec un compte d'un domaine Active Directory, celle-ci ne peut pas valider directement l'authentification
- En effet, la machine ne dispose pas des empreintes NTLM des comptes du domaine
- Seuls les contrôleurs de domaine connaissent les empreintes NTLM des utilisateurs du domaine

NetLogon

- Le protocole NetLogon permet l'interconnexion de la base SAM d'une machine avec celle d'un contrôleur de domaine
- Cela permet, à la machine membre, de pouvoir valider les authentifications auprès d'un tiers (mécanisme appelé « *Pass-Through Authentication* »)
- Ces authentifications peuvent être :
  - des **authentifications interactives** (*interactive logon*)
  - des **authentifications distantes** (*network logon*)
- Les échanges NetLogon entre une machine membre et un DC sont protégés par la mise en place du ***Secure Channel***

Échanges NetLogon



LogonInteractive:

`Identity, NtOwfPassword` →

← `LogonScript, NETLOGON_VALIDATION_SAM_INFO2`

LogonNetwork:

`Identity, LmChallenge, LmChallengeResponse, NtChallengeResponse` →

← `LMKey, UserSessionKey, NETLOGON_VALIDATION_SAM_INFO2`

Principales fonctions RPC de NetLogon

- Établissement du *Secure Channel* :
  - `NetrServerReqChallenge`
  - `NetrServerAuthenticate3`
- Authentication *pass-through* :
  - `NetrLogonSamLogonEx`
  - `NetrLogonSamLogonWithFlags`
- Changement d'un mot de passe :
  - `NetrServerPasswordSet2`
- Réplication PDC/BDC (compatibilité avec domaines NT)
  - `NetrDatabaseSync2`

Sécurité du Secure Channel

- Initialement (jusqu'à Windows NT 4) : « ***DES Session-Key*** »
  - Calcul des clés avec DES_ECB (clé de 56 bits)
  - Chiffrement avec RC4
  - Signature avec HMAC-MD5
- Windows 2000 : « ***Strong-key*** »
  - Calcul des clés avec HMAC_MD5 (clé de 128 bits)
  - Chiffrement avec RC4
  - Signature avec HMAC-MD5
- Windows 7 / 2008 R2 : « ***AES Session-Key*** »
  - Calcul des clés avec HMAC-SHA256
  - Chiffrement avec AES-128
  - Signature avec HMAC-SHA256
- Les DC sous Windows 2008 R2 ne supporte plus de clients Windows NT 4 car une taille de clé d'au moins 128 bits est imposée pour le Secure Channel

Authentification et génération des clés

- Il doit y avoir une authentification mutuelle entre le contrôleur de domaine et la machine
- Pour cela, un secret partagé est partagé entre les deux parties :
  - Machine : mot de passe (`$machine.ACC`)
  - DC : empreinte NTLM → MD4(MDP)

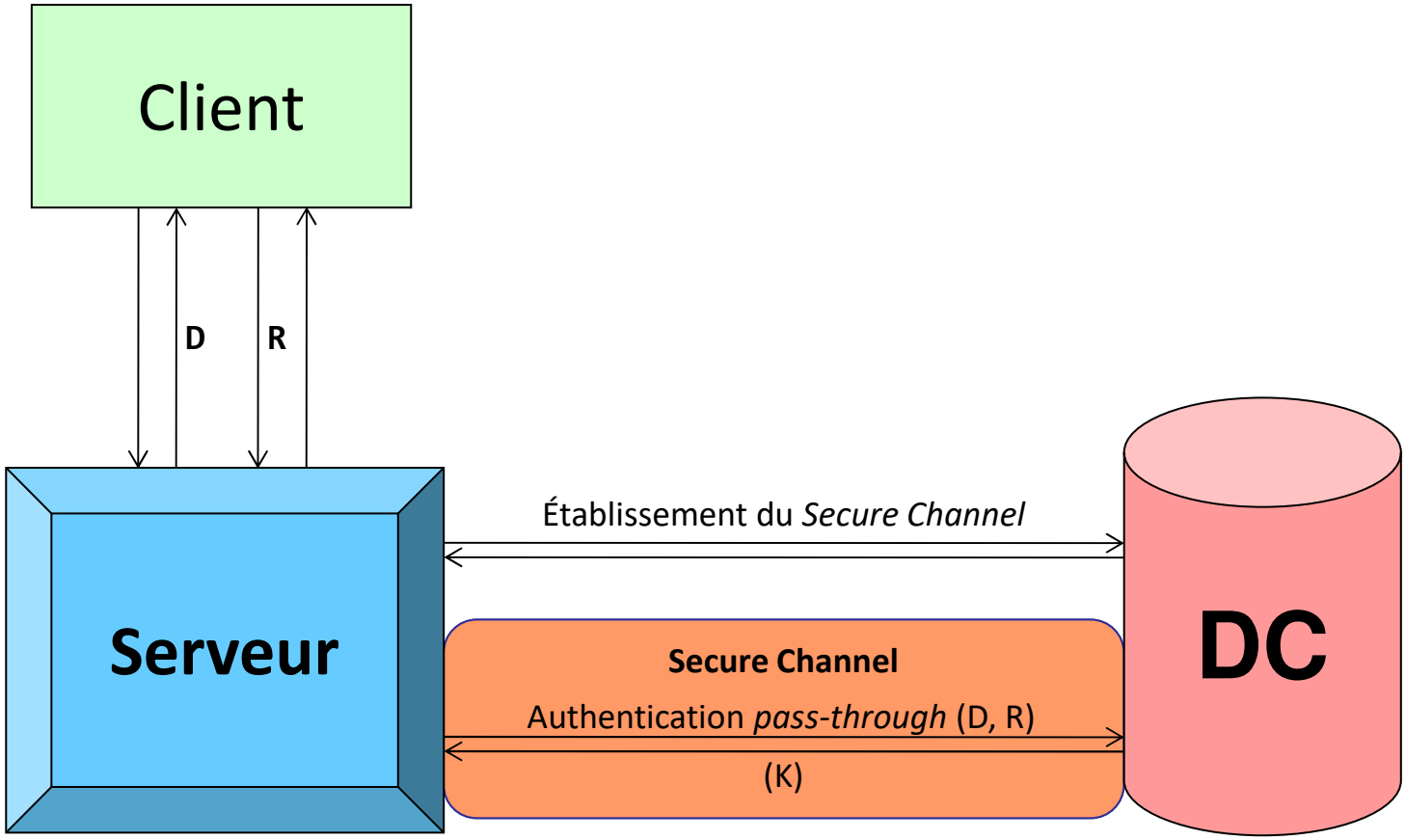
- L'échange `NetrServerReqChallenge` permet déchanger les défis (ClientChallenge et ServerChallenge)
- L'échange `NetrServerAuthenticate3` permet l'authentification mutuelle (calcul de ClientCredential et ServerCredential)

Calculs pour la protection du Secure Channel

- SessionKey = SHA256(NTLM | ClientChallenge | ServerChallenge)
  - Uniquement les 16 premiers octets sont conservés
- ClientCredential = AES-128-CFB(SessionKey, ClientChallenge)
- ServerCredential = AES-128-CFB(SessionKey, ServerChallenge)

- Tous les messages protégés par le *Secure Channel* doivent générer un bloc d'authentification

Échanges NTLM/NetLogon



Kerberos

Kerberos

- Kerberos a été introduit dans Windows 2000

- Initialement, Kerberos est un standard du MIT. L'implémentation dans Windows 2000 s'inspire de la version 5, mais y ajoute des extensions (normalisées depuis)

Principe de base

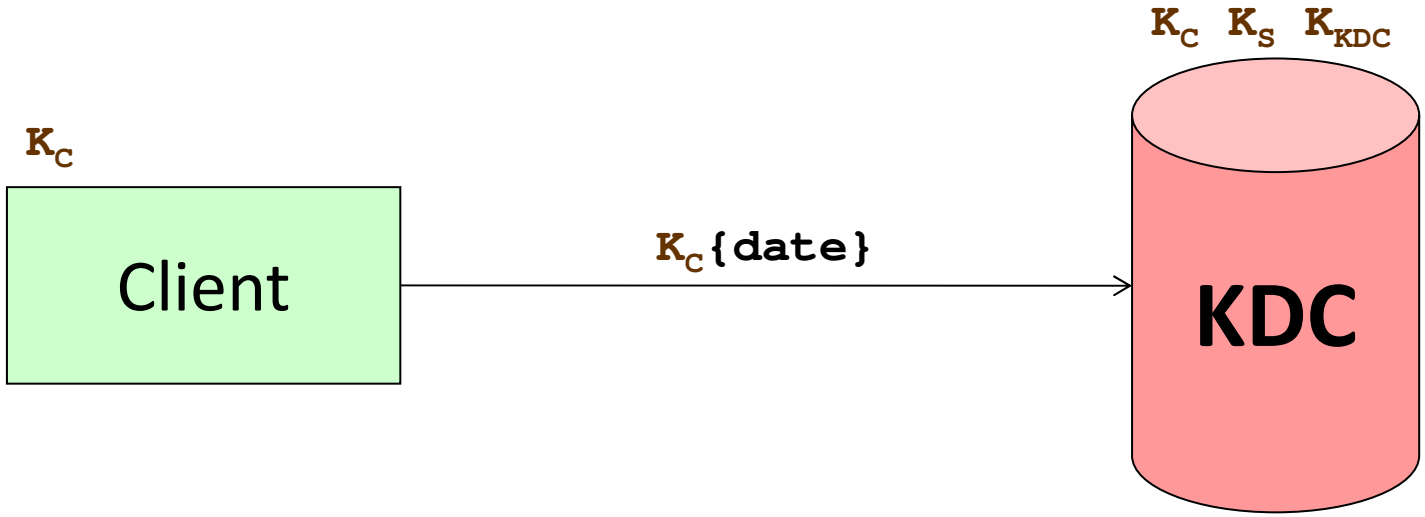
- Kerberos met en œuvre une authentification avec 3 acteurs :
  - Un utilisateur identifié par un **UPN (*User Principal Name*)**
  - Un service identifié par un **SPN (*Service Principal Name*)**
  - Un tiers de confiance, le **KDC (*Key Distribution Center*)**
- Chaque acteur dispose d'un secret (on parle de **clé Kerberos**) connu de lui et du KDC : $\mathbf{K_C}$, $\mathbf{K_S}$
- Le KDC dispose de sa propre clé : $\mathbf{K_{KDC}}$
- En réalité, il y a plusieurs types de clés :
  - RC4 : identique à l'empreinte NTLM
  - DES : supporté par comptabilité avec la RFC, mais l'utilisation n'est pas autorisé
  - AES : apparu avec Windows 2008/Vista

Kerberos

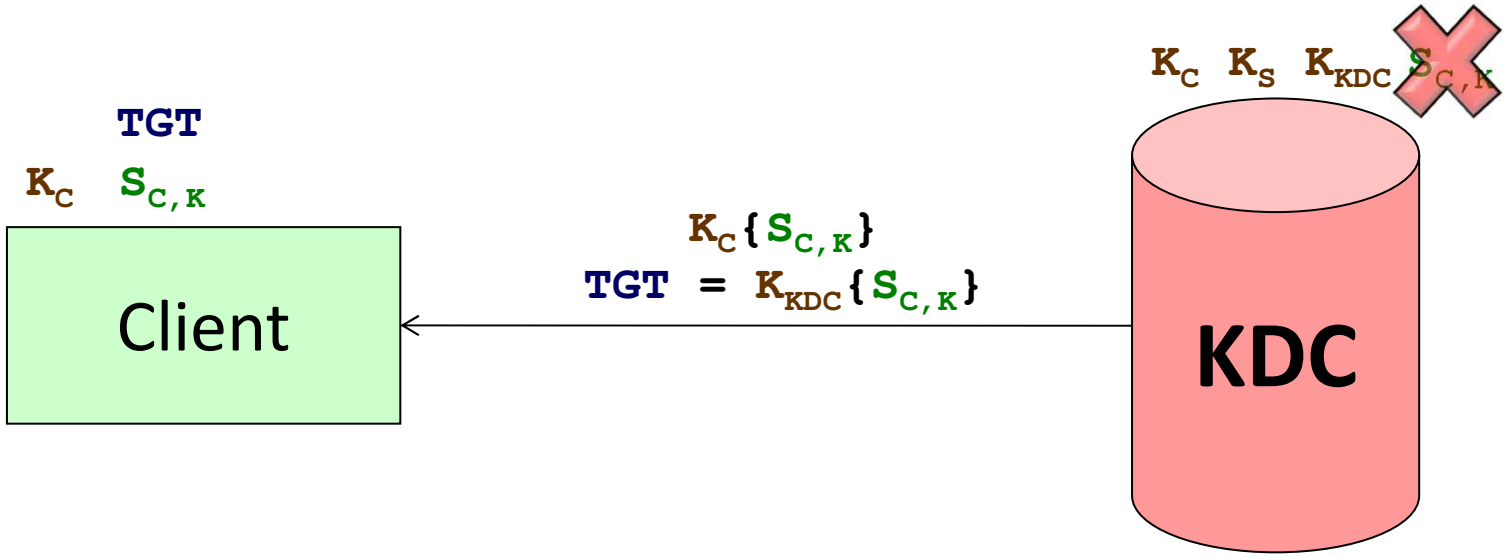
- Kerberos est composé de trois services :
  - **Authentication Service** (AS), qui délivre :
    - un *Ticket Granting Ticket* (TGT)
    - une *logon session key*
  - **Ticket-Granting Service** (TGS), qui délivre :
    - un *service ticket*
    - une *service session key*
  - **Client/Server** (CS)

    service qui présente les tickets de service d'un client à un service

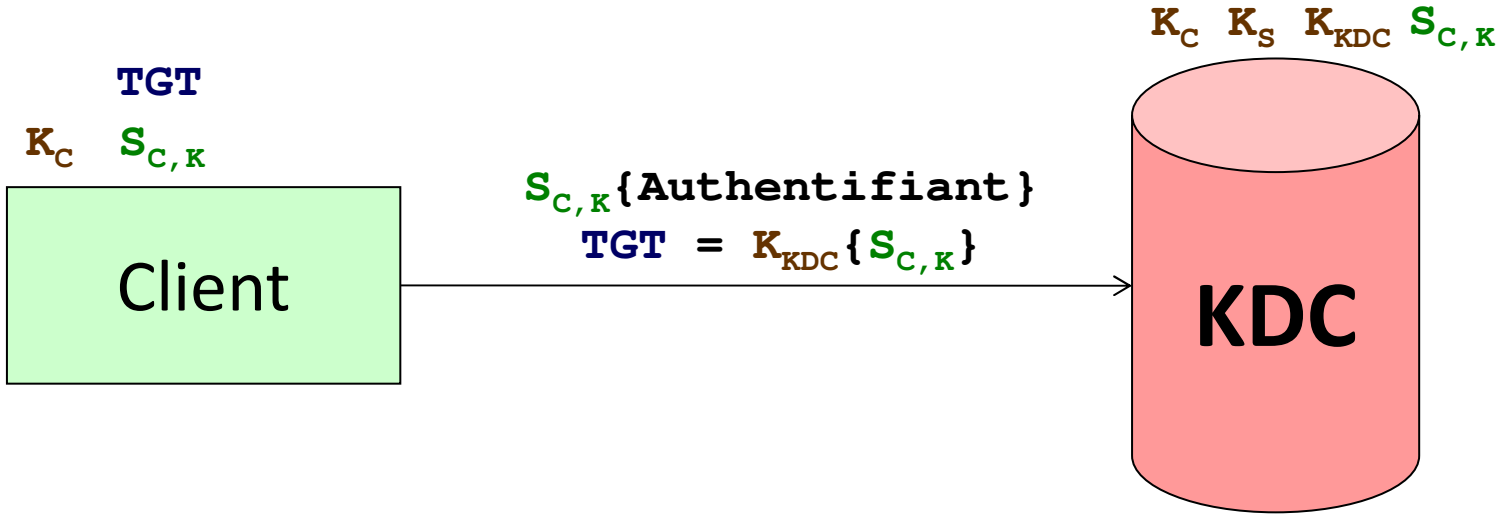
Kerberos
KRB_AS_REQ



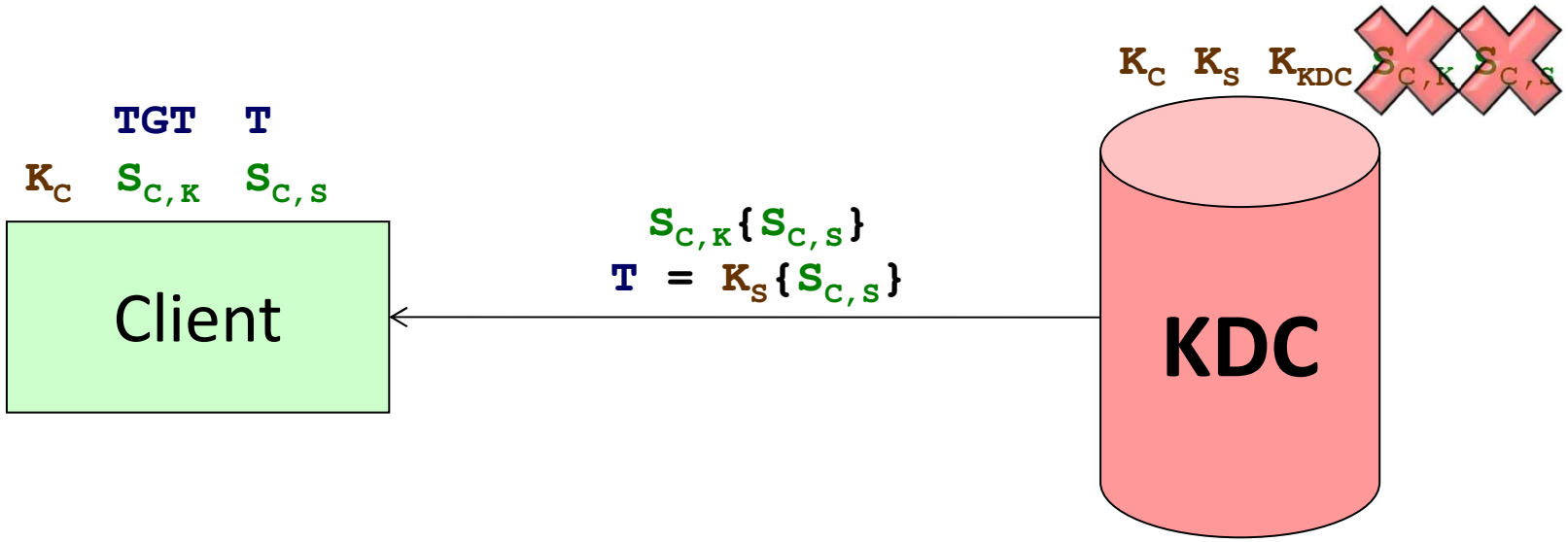
Kerberos
KRB_AS_REP



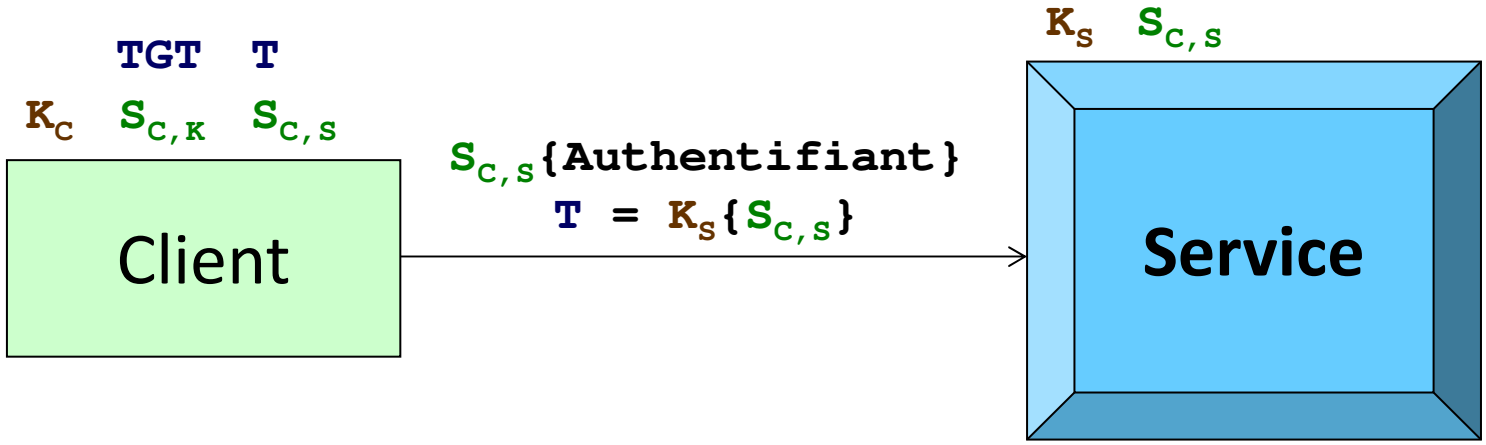
Kerberos
KRB_TGS_REQ



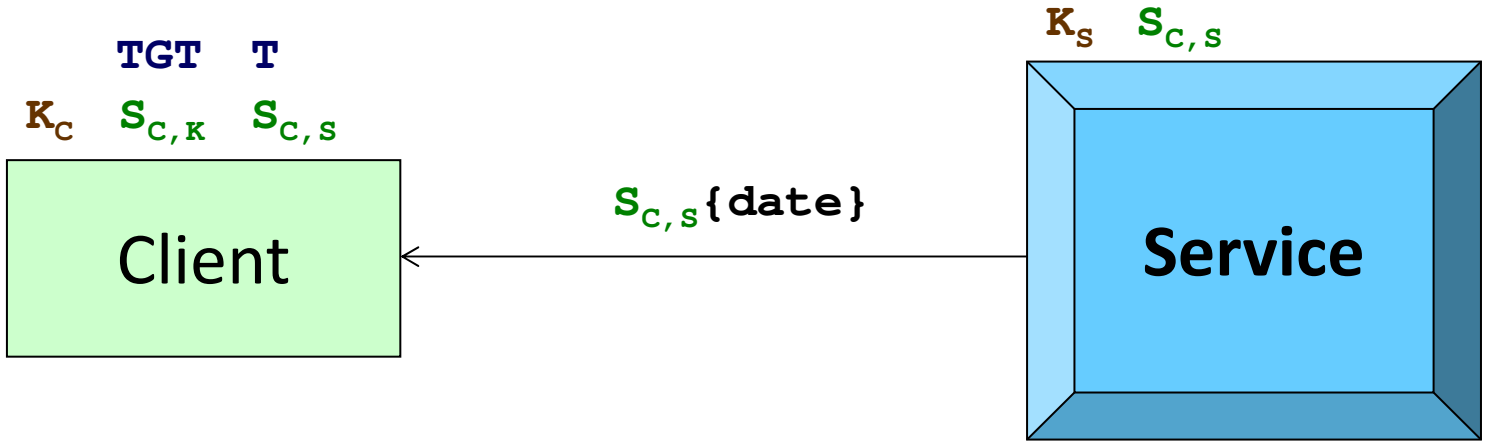
Kerberos
KRB_TGS_REP



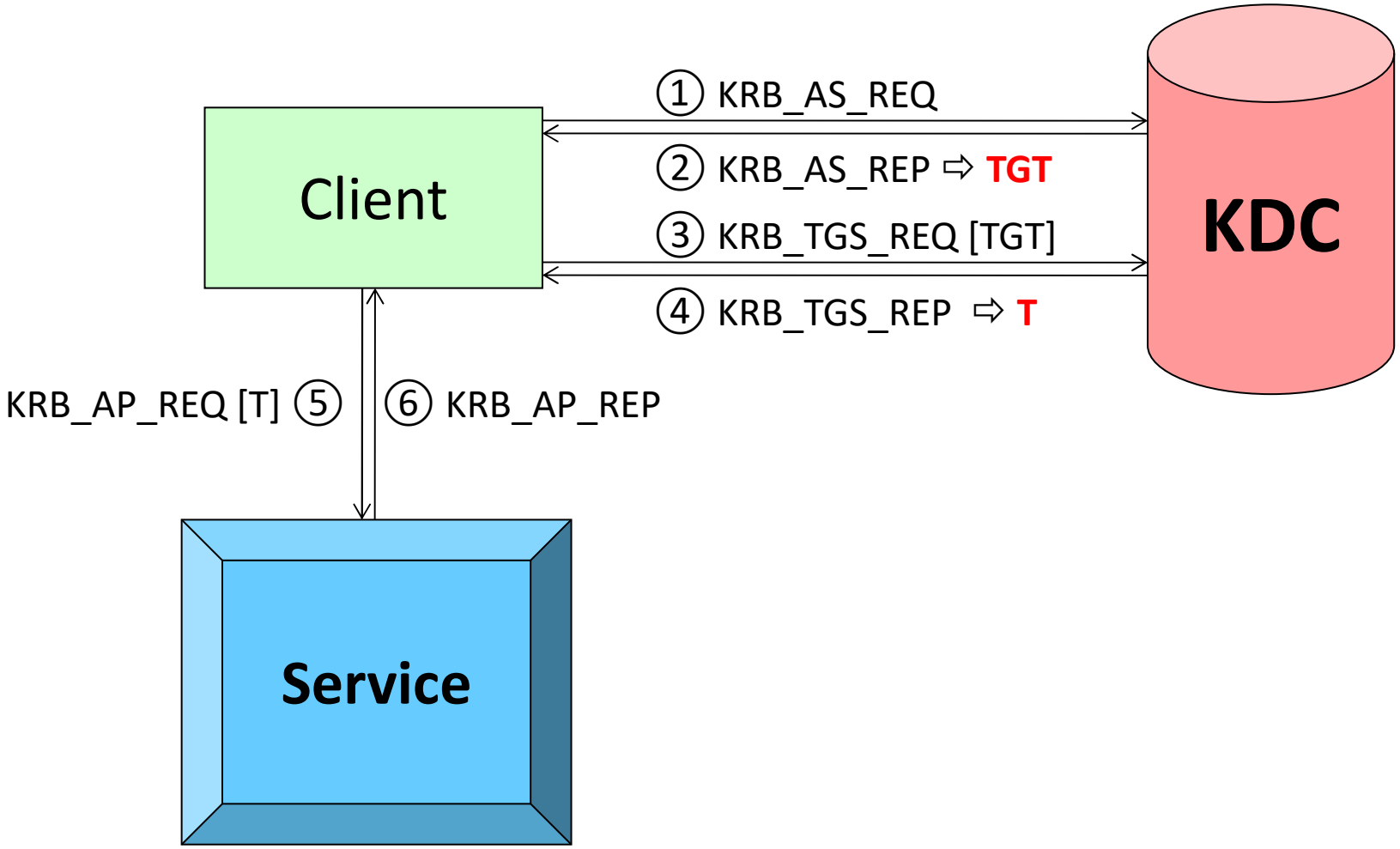
Kerberos
KRB_AP_REQ



Kerberos
KRB_AP_REP



Kerberos


① KRB_AS_REQ
② KRB_AS_REP ⇨ TGT
③ KRB_TGS_REQ [TGT]
④ KRB_TGS_REP ⇨ T
Client
KDC
KRB_AP_REQ [T] ⑤
⑥ KRB_AP_REP
Service

Comparaison NTLM / Kerberos

	LM/NTLM	Kerberos
Type de crypto	Symétrique	Symétrique
Plateformes Microsoft	Toutes	2000 et suivants
Montée en charge	Faible	Élevée
Authentification mutuelle	NON	Option
Délégation supportée	NON	OUI
Support carte à puce	NON	Extensions
Standard	Microsoft	IETF

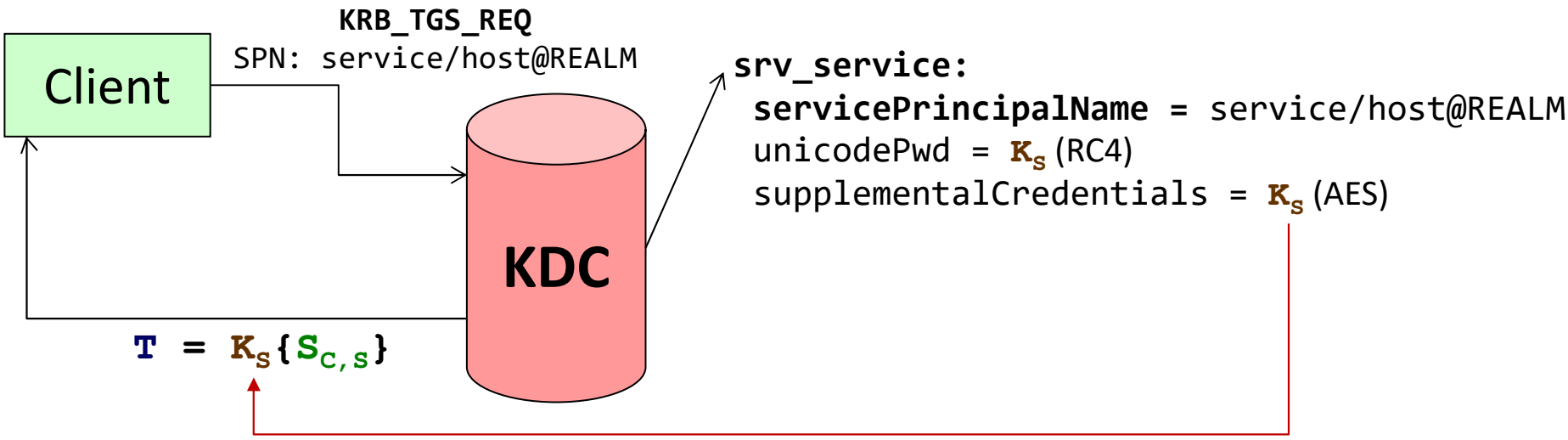
Clés K_C et K_{KDC}

- Les clés K_c sont calculées depuis le mot de passe de l'utilisateur et :
  - **Côté client** : elles sont associées à la session d'authentification de l'utilisateur et conservées dans la mémoire de LSASS
  - **Côté KDC** : elles sont stockées dans les attributs **`unicodePwd`** (empreinte NTLM) et **`supplementalCredentials`** (clés DES et AES) associés au compte de l'utilisateur
- Les clés K_{KDC} sont générés à la création du domaine et sont conservées dans les attributs de l'utilisateur krbtgt (compte utilisateur désactivé et non modifiable)

Clés K_S
Compte des services Kerberos

- Tous les services Kerberos doivent être associés à un compte (machine ou utilisateur) de l'Active Directory
- Cette association est concrétisée, pour les objets de l'AD, par l'attribut `servicePrincipalName` de la forme :

`<service class>/<host>:<port>/<service name>`



Clés K_S
Compte des services Windows

- Pour les processus s'exécutant sous les entités LocalSystem (SYSTÈME) et NetworkService (SERVICE RÉSEAU), les identifiants des sessions d'authentification sont prédéfinies :
  - SYSTÈME : 0x3e7
  - SERVICE RÉSEAU : 0x3e4
- Les clés Kerberos associées à ces deux sessions (donc K_S) sont calculées depuis le mot de passe de la machine dans le domaine (stockés dans l'attribut `$machine.ACC` des « secrets LSA »)
- Au niveau des KDC, ces clés sont associées au compte de la machine dans le domaine (exemple : SERVEUR$)

Attaques sur Kerberos
Clés K_C des utilisateurs

- Les clés de sessions ($S_{X,Y}$), du KDC (K_{kdc}) et du service (K_s) sont supposées être résistantes (car générées aléatoirement)
- En revanche, les clés des utilisateurs (K_c) sont généralement calculées à partir d'un mot de passe (donc potentiellement faible)
- Ainsi, tous les échanges réseau capturés où K_c intervient permettent de réaliser des attaques sur le mot de passe ayant généré K_c

Attaques sur Kerberos
Clés K_C des utilisateurs

- **K_c** est en particulier utilisée dans les échanges AS :
  - Sans pré-authentification, tout le monde peut récupérer un TGT chiffré par **K_c**
  - Avec la pré-authentification, une capture réseau permet de récupérer **K_c{pre-auth}** où pre-auth = *YYYYMMDDHHMMSSZ*
  - Dans tout les cas, une réponse AS_REQ contient une partie chiffrée par **K_c**

Attaques sur Kerberos
SPN associé à un compte utilisateur

- Par défaut, dans l'Active Directory, les SPN sont associés aux comptes machines. Par exemple, pour Serveur-01$, les SPN sont :
  - host/serveur-01, cifs/serveur-01, …
- Il est cependant possible de définir un SPN pour un compte utilisateur en modifiant son attribut `servicePrincipalName`
- Rappel : la partie chiffrée d'un ticket est chiffrée avec la clé Kerberos du compte associé au SPN du ticket
- Ce n'est pas gênant pour les comptes machine car leur mot de passe, à partir duquel les clés Kerberos sont calculées, est garanti être totalement aléatoire
- En revanche, les mots de passe des comptes utilisateur n'étant pas aléatoire, la robustesse des clés Kerberos n'est pas assurée

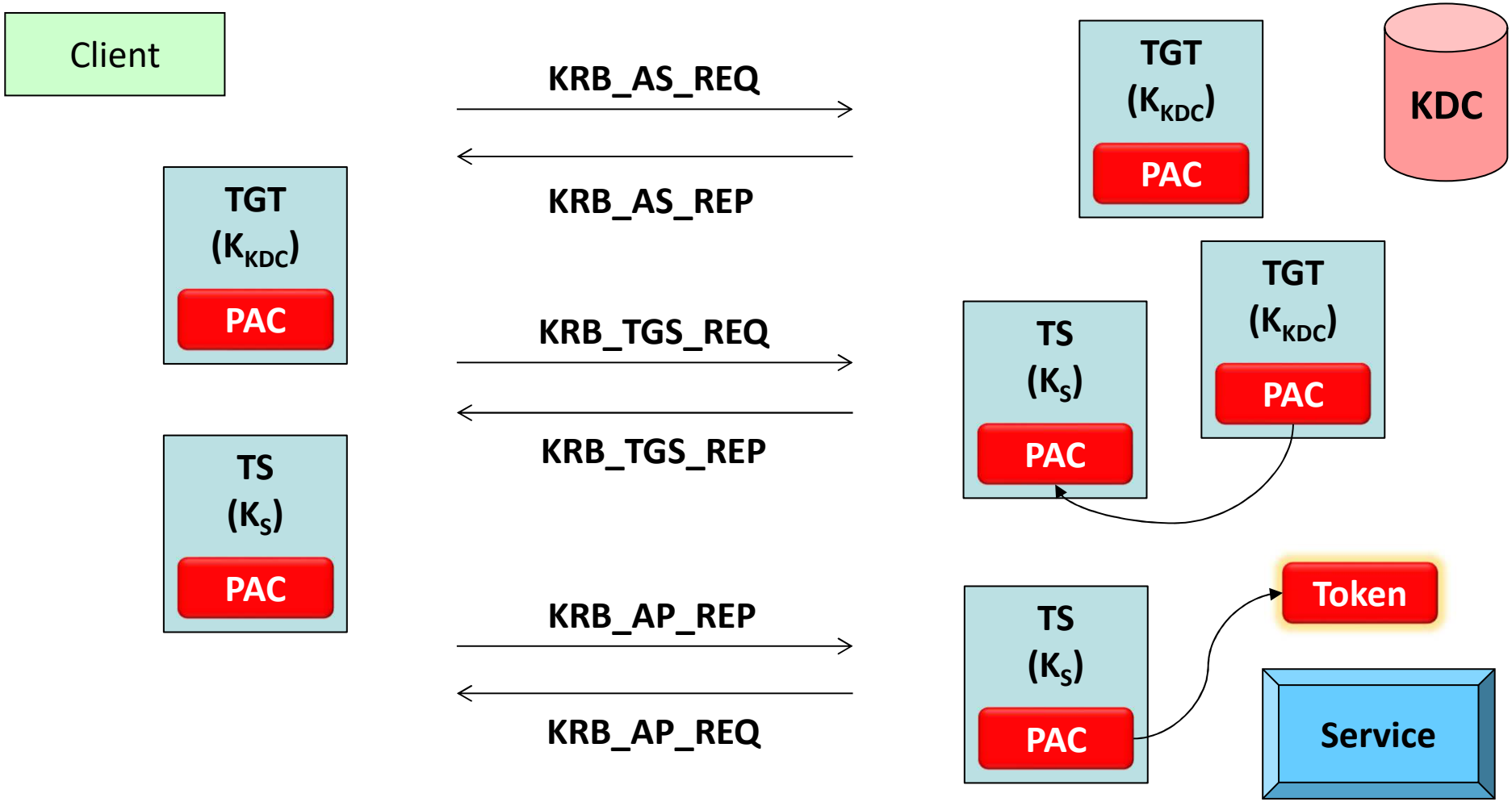
Attaques sur Kerberos
Pass-the-key

- La connaissance d'une clé utilisateur (K_c) permet de demander un TGT puis des TS
- Ainsi, comme pour *Pass-the-hash*, il n'est pas nécessaire de retrouver le mot de passe d'un utilisateur (à partir de la clé K_c) pour pouvoir s'authentifier auprès d'un service

PAC
(Privilege Attribute Certificate)

- Kerberos est un protocole d'**authentification**
- Il est également possible de l'utiliser pour transporter des données d'**autorisation** via le champ `autorization-data` des tickets
- Windows utilise ce champ pour transporter une structure de type `KERB_VALIDATION_INFO` spécifiant les données d'autorisation de l'utilisateur (PAC) :
  - `PISID LogonDomainId;`
  - `ULONG UserId;`
  - `ULONG PrimaryGroupId;`
  - `ULONG GroupCount;`
  - `PGROUP_MEMBERSHIP GroupIds;`
  - `PKERB_SID_AND_ATTRIBUTES ExtraSids;`

Transport de la PAC

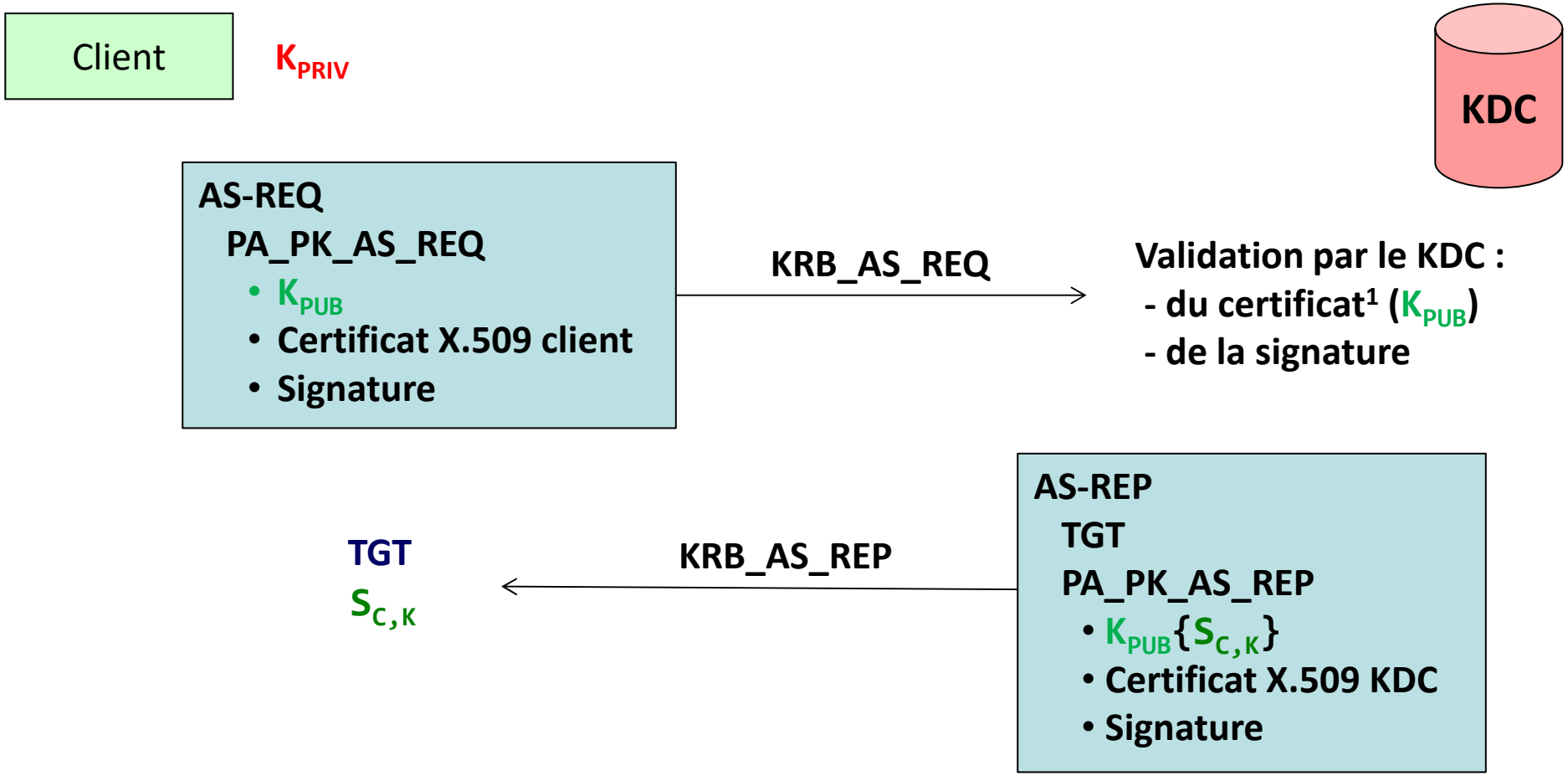


PKINIT

Public Key Cryptography for Initial Authentication

- Extensions (RFC 4556 et MS-PKCA) permettant l'authentification au moyen d'algorithmes asymétriques
- Permet la mise en œuvre :
  - d'échanges de clés basé sur *Diffie-Hellman* (option)
  - d'authentifications basées sur des clés RSA ou DSA
- Il devient alors possible d'utiliser :
  - des certificats numériques (x509)
  - des cartes à puces
  - des *tokens* d'authentification

PKINIT


Client
K_{PRIV}
KDC
AS-REQ
PA_PK_AS_REQ
• K_{PUB}
• Certificat X.509 client
• Signature
KRB_AS_REQ
Validation par le KDC :
- du certificat[1] (K_{PUB})
- de la signature
AS-REP
TGT
PA_PK_AS_REP
• $K_{PUB}\{S_{C,K}\}$
• Certificat X.509 KDC
• Signature
KRB_AS_REP
TGT
$S_{C,K}$

Mise en œuvre de l'authentification

Formes d'authentification

- Deux formes d'authentification sont à distinguer :
  - L'**authentification locale** (*Interactive Authentication*)
    - Permet d'authentifier un utilisateur puis de créer une **session d'authentification** et un **contexte de sécurité**
    - Mis en œuvre par les **AP** (***Authentication package***)
  - L'**authentification distante** (*Noninteractive Authentication*)
    - Permet d'authentifier un utilisateur distant via la mise en œuvre d'un protocole d'authentification réseau
    - Mis en œuvre par les **SSP** (***Security Support Provider***)

Sessions d’authentification

Session d'authentification (*logon session*)

- Pour chaque authentification locale réussie, une **session d'authentification** est créée
- Une session d'authentification contient le contexte d'authentification et reste valide tant que toutes les références à la session ne sont pas closes
- Les sessions d'authentification sont gérées :
  - Par LSASS (structure **`SECURITY_LOGON_SESSION_DATA`**)
  - Par le noyau (structure **`_SEP_LOGON_SESSION_REFERENCES`**)
- Tous les processus sont **obligatoirement** associés à une session d'authentification via leur contexte de sécurité

Session d'authentification

- Une session d'authentification est composée :
  - d'un identifiant unique de session (*logonID*) de type LUID
  - du nom du domaine et du nom de l'utilisateur
  - de l'identifiant de sécurité (SID) de l'utilisateur
  - du nom de l'AP (*Authentication package*) ayant validé et créé la session
  - du type de session (Interactive, Réseau, Service)
- Journalisation de sécurité associée :
  - Ouverture de session : 4624 (S) / 4625 (F)
  - Fermeture de session : 4634 (S)

Authentification locale

Mise en œuvre de l'authentification

- Les authentification locales à un système permettent :
  - De **valider** l'authentification par rapport aux comptes reconnus du système (comptes locaux ou d'un domaine Active Directory)
  - D'**autoriser** l'authentification en fonction du **type d'authentification** demandé
  - De créer une **session d'authentification** dans la base des sessions d'authentification
  - De créer un **contexte de sécurité** (*token*)

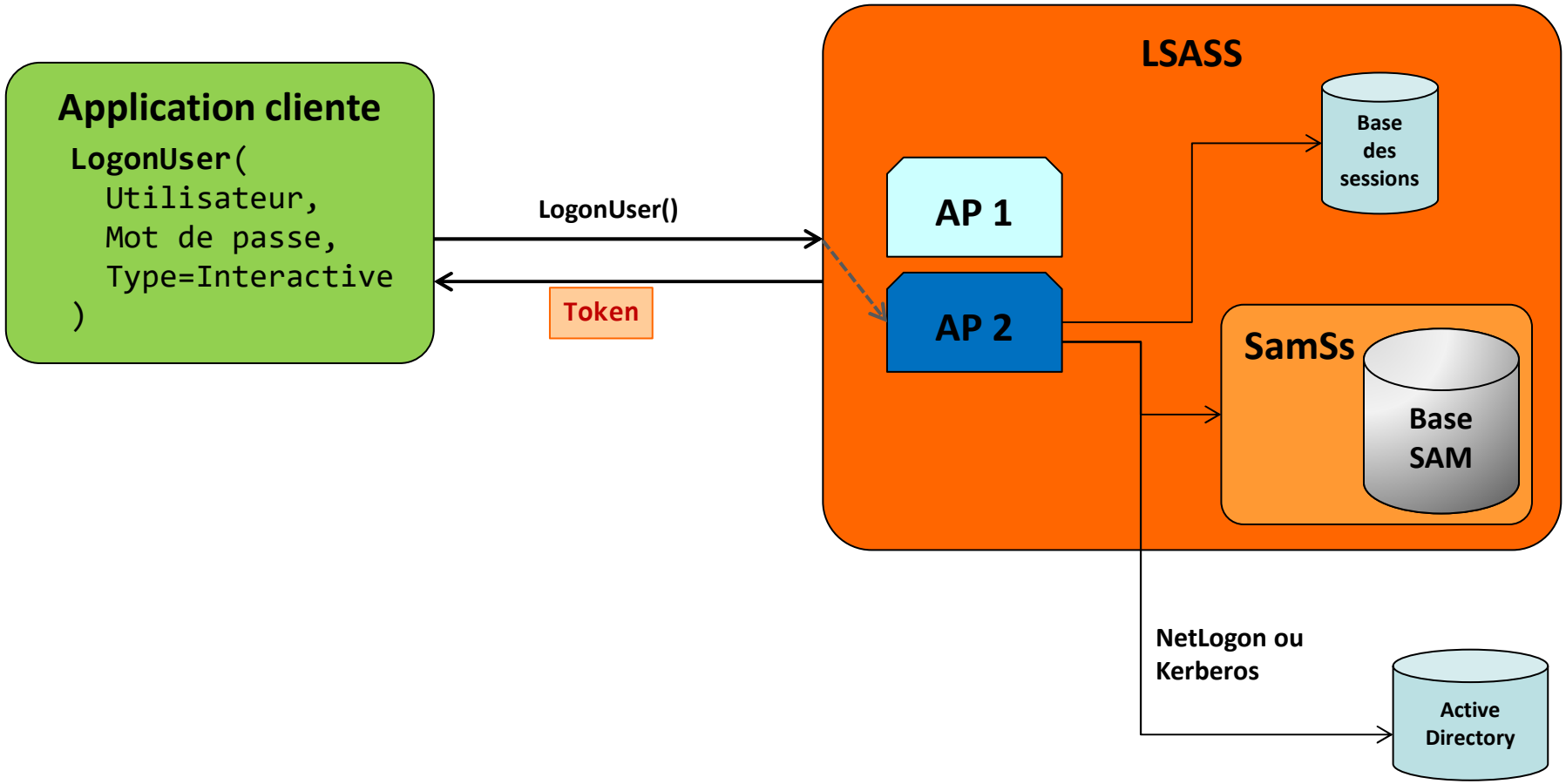
Mise en œuvre des authentifications locales

- Les authentifications locales sont validées par LSASS et déléguées aux **AP (*Authentication package*)**
- Les authentifications locales sont effectuées via la fonction **`LogonUser()`** qui nécessite :
  - Le nom du compte de l'utilisateur
  - Le type de l'authentification
  - Le type d'AP devant valider l'authentification
  - Optionnellement, le mot de passe **en clair** associé au compte

Types principaux

- **Interactive** (2) : validation d'une authentification, ouverture d'une session et création d'un contexte de sécurité
- ***Remote Interactive*** (10) : identique à *Interactive*, mais est soumis à une autorisation différente (utilisé en particulier par WinLogon lors de l'ouverture d'une session distante via *Terminal Services*)
- **Service** (5) : utilisé par le SCM pour créer le contexte de sécurité lors du lancement d'un service (nécessite TCB)
- ***Batch*** (4) : validation d'une authentification sans consentement nécessaire de l'utilisateur (utilisé en particulier par le planificateur de tâches)
- ***Network*** (3) :
  - Simple validation d'un utilisateur et de son mot de passe (pas de mise en cache des authentifiants)
  - Ouverture d'une session résultant d'une authentification par un SSP de type LOGON

Exemple du type *Interactive*



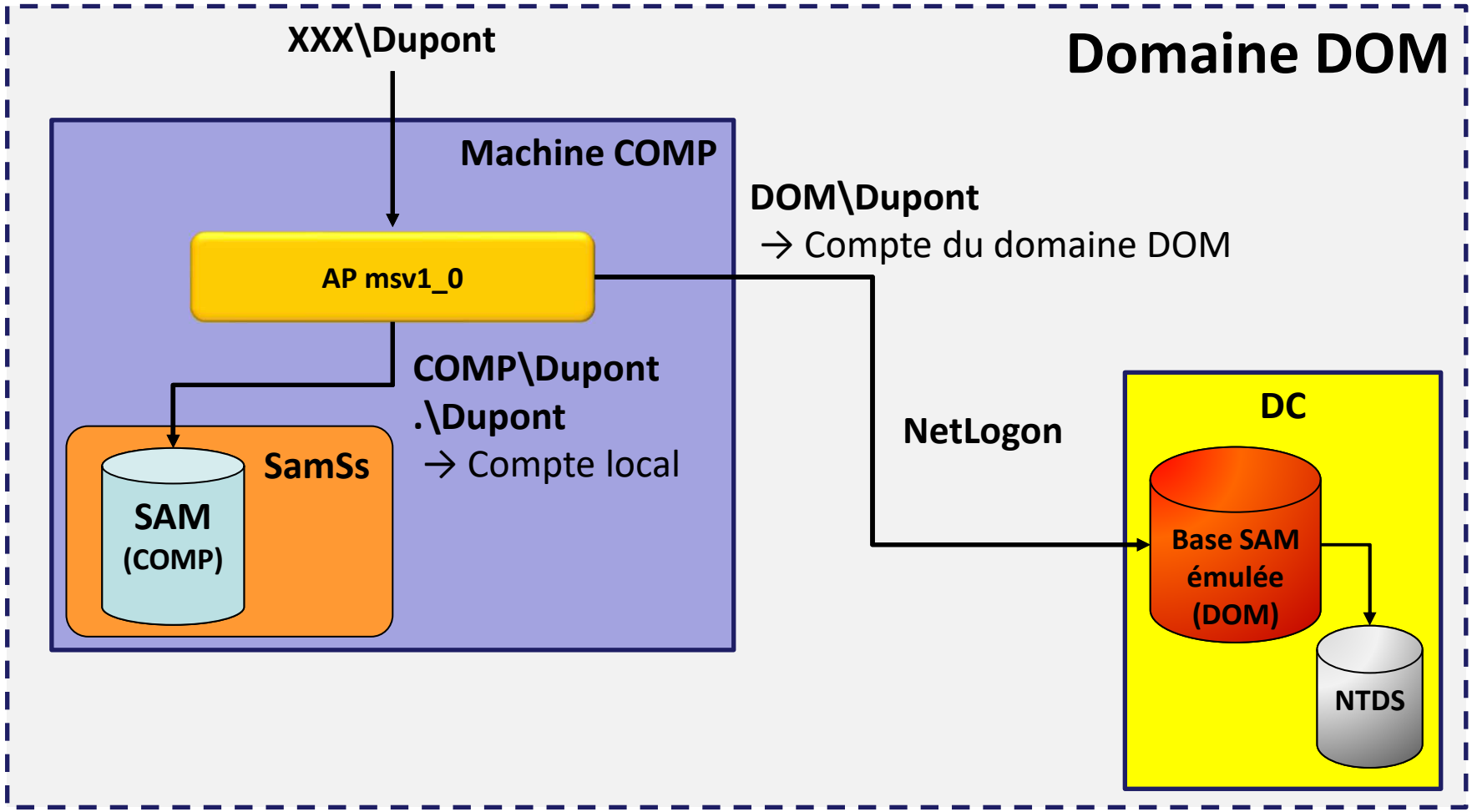
AP Microsoft

- Windows dispose en standard deux AP :
  - **msv1_0**
    - authentification des **comptes locaux** via la base SAM
    - authentification des comptes d'un domaine (forme historique compatible pré-Windows 2000 et basée sur NetLogon). Utilisé si Kerberos ne peut être utilisé
  - **Kerberos**
    - authentification *via* le protocole Kerberos (méthode privilégiée pour les **comptes d'un domaine** Active Directory)

Validation par l'AP msv1_0

- À partir du mot de passe fourni, l'empreinte NTLM est calculée puis :
  - Pour un **compte local** : l'empreinte NTLM de l'utilisateur est extraite de la base SAM (accédée via le service `SamSs` s'exécutant dans LSASS) puis comparée à celle calculée
  - Pour un **compte d'un domaine** : l'empreinte NTLM est envoyée, *via* NetLogon, à un contrôleur de domaine qui effectue la comparaison

Validation par l'AP msv1_0

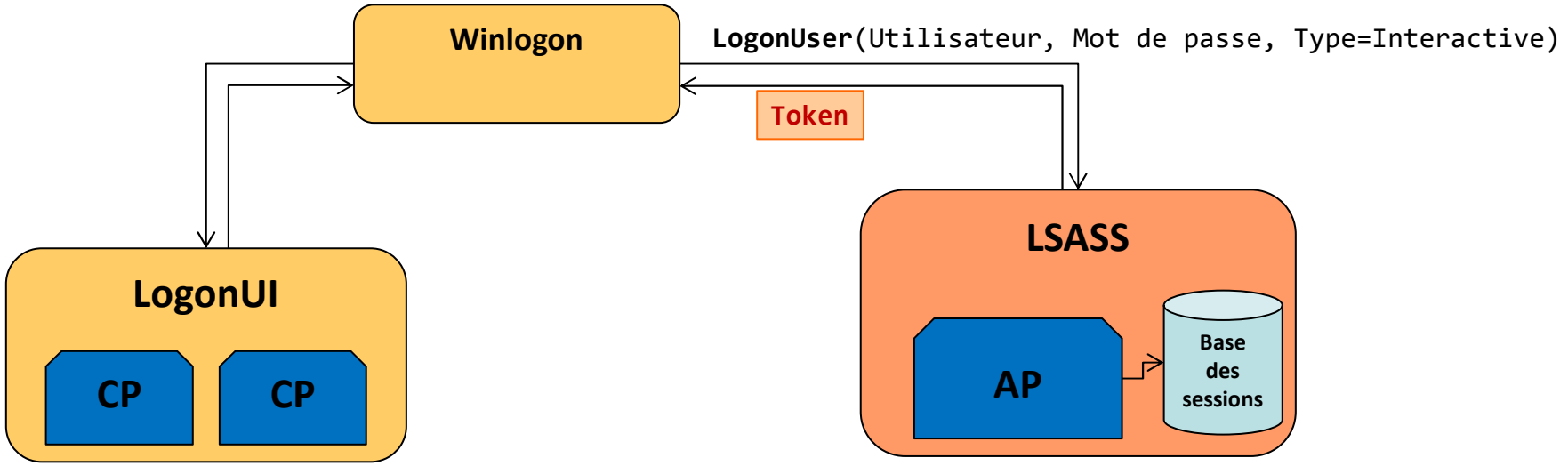


Validation par l'AP Kerberos

- L'AP Kerberos demande un ticket de service au nom de l'utilisateur pour le service **`host/nom_de_machine@REALM`**
- Cette demande nécessite préalablement la demande d'un TGT au nom de l'utilisateur

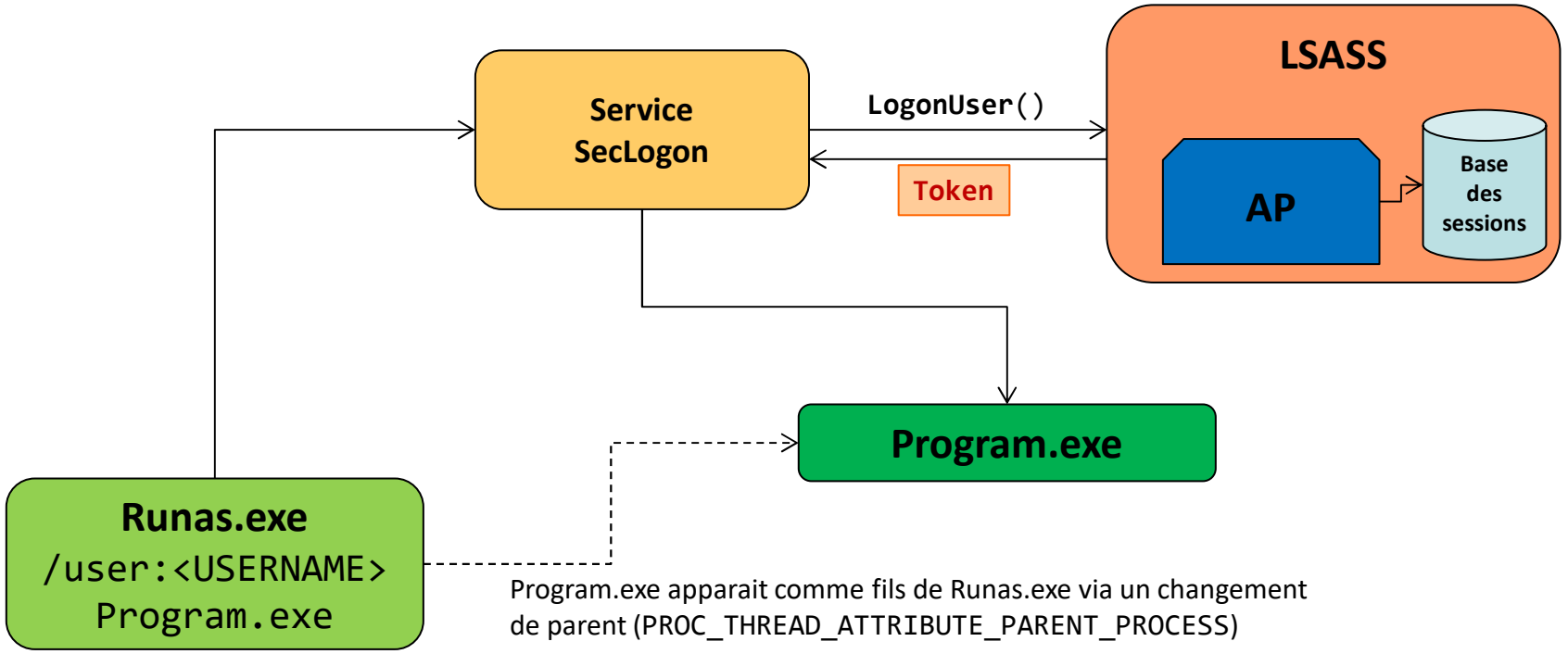
Authentifications interactives réalisées par WinLogon (Vista+)

- Les *credentials* sont récupérés par WinLogon via le programme **LogonUI (Windows LogOn User Interface)**
- Celui-ci charge les ***Credentials Providers*** (objets COM) en charge de la saisie des différents type de *credentials*



Authentifications interactives réalisées par le service SecLogon (ouverture de session secondaire)

- Permet de lancer un programme dans une nouvelle session utilisateur



Cache des authentifications

blogs.technet.com/b/instan/archive/2011/12/06/cached-logons-and-cachedlogonscount.aspx

- Pour les **authentifications locales des comptes de domaine**, msv1_0 dispose d'un mécanisme de cache pour pallier une éventuelle indisponibilité des DC
- Si une authentification est réussie, celle-ci est mise en cache (stockage d'une empreinte au format MSCache)
- Si msv1_0 doit valider une authentification d'un compte du domaine et qu'aucun contrôleur de domaine n'est joignable, l'authentification est validée par rapport à celle mise cache (si elle existe)
- Le paramètre **CachedLogonsCount** détermine le nombre d'entrée dans le cache

Algorithme des cache

Domain cached credentials (DCC)

- Deux générations d'algorithmes existent :
  - MsCache (XP/2003) :

    `DCC1 = MD4(NTLM | username)`

  - MsCache 2 (Vista+)

DCC2 = PBKDF2(SHA1, DCC1, salt=username, NL$IterationCount)

NL$IterationCount = 10240 par défaut

Authentification distante

Authentification distante (*Noninteractive Authentication*)

- Les authentifications distantes sont mises en œuvre via la SSPI et déléguées aux **SSP (*Security Support Provider*)**
- **Côté client**, cela permet :
  - De spécifier le SSP à utiliser
  - De spécifier un contexte d'authentification à utiliser
  - De s'authentifier auprès d'un serveur
- **Côté serveur**, cela permet :
  - De spécifier le SSP à utiliser (ce qui indique la méthode de validation de l'authentification)
  - De valider l'authentification en provenance d'un client
- Les échanges, qui sont distants, nécessitent la mise en œuvre d'un protocole d'authentification (exemple : NTLMSSP, Kerberos)

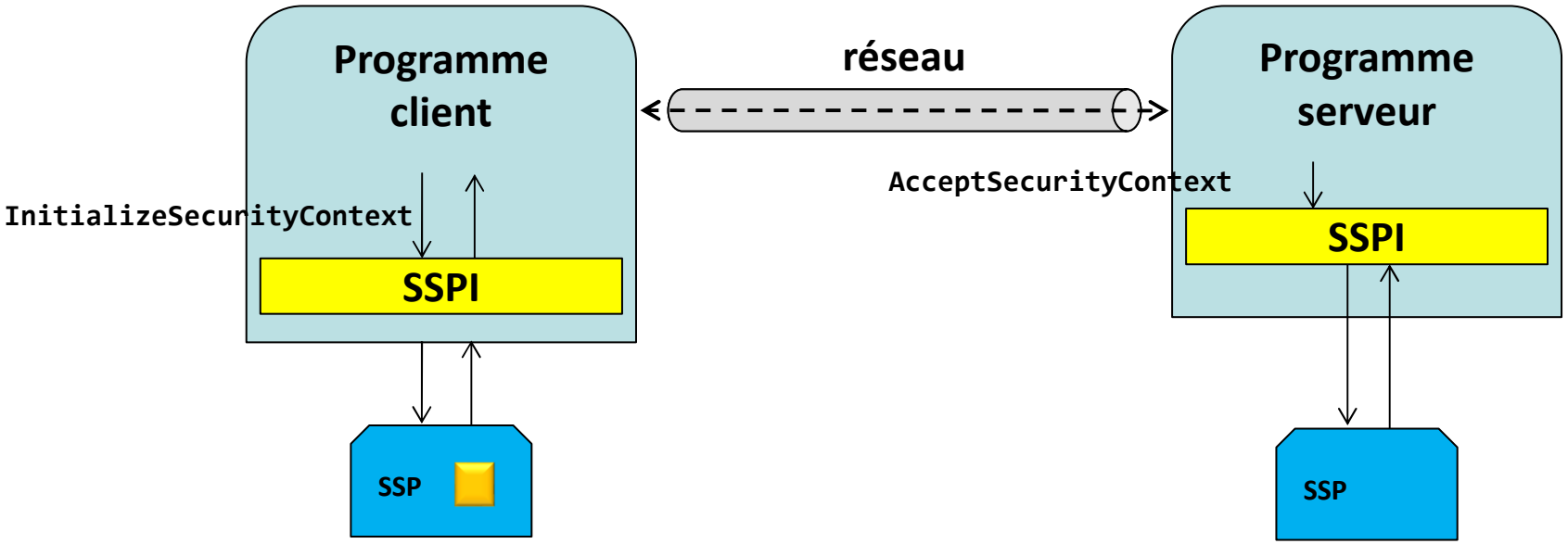
SSP Microsoft

HKLM\SYSTEM\CurrentControlSet\Control\Lsa\Security Packages

- NTLM (msv1_0)
- Kerberos
- Negotiate (SPNego)
- Schannel/Unified Security Protocol Provider (SSL/TLS)
- WDigest — Windows XP
- Credential Security Support Provider (CredSSP) — Vista (+XP SP3)
  - TSSSP (TS Service Security Package) — Vista (+XP SP3)
- NegoExtender — Windows 7
- Pku2u — Windows 7

  (authentification pair-à-pair pour les groupes résidentiels)

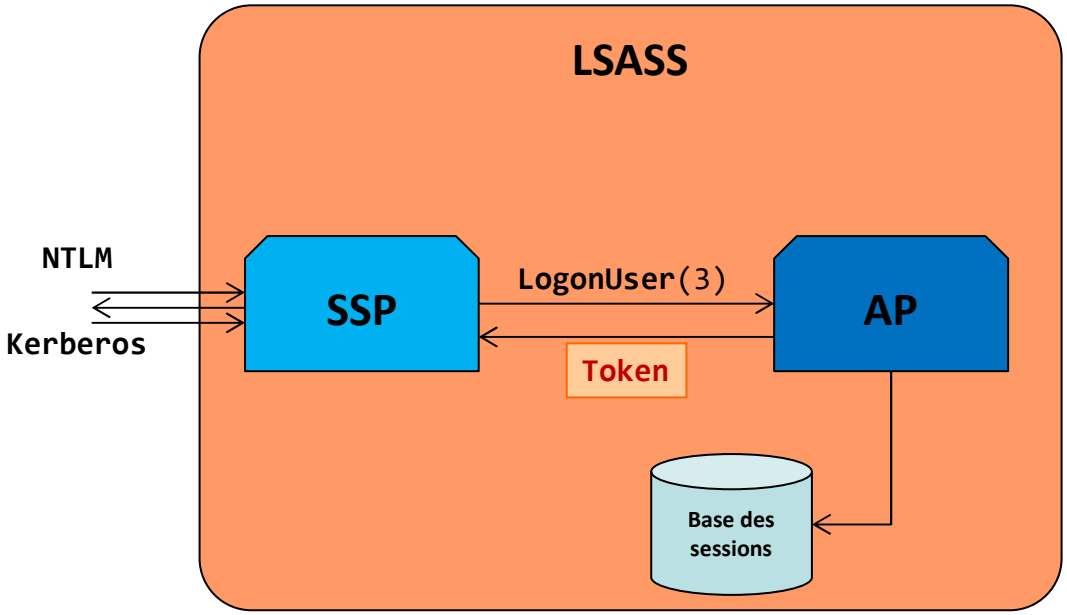
Fonctionnement général des SSP



SSP/AP

- NTLM et Kerberos sont des SSP/AP (SSP de type LOGON)
- Côté serveur, les comptes sont validés par rapport aux comptes reconnus par le système (comptes locaux de la base SAM ou compte d'un domaine Active Directory)
- Si le SSP valide une authentification distante, il demande à son AP correspondant de réaliser une authentification locale de type réseau (3), ce qui permet :
  - D'autoriser l'authentification de type réseau
  - De créer une session d'authentification
  - De créer un contexte de sécurité (*token*) représentant l'utilisateur distant authentifié par le réseau

SSP/AP


LSASS
NTLM
Kerberos
SSP
LogonUser(3)
Token
AP
Base des sessions

Contexte d'authentification et SSO Windows (*Single Sign On*)

Contexte d'authentification du client

- Côté client, trois contextes d'authentification sont possibles lors de l'initialisation du SSP :
  - **Authentification explicite** : les authentifiants (*credentials*) sont fournis explicitement lors de l'appel
  - **Authentification sauvegardée** : aucun authentifiant n'est fourni lors de l'appel. Ceux-ci sont récupérés dans le *Credential Manager*
  - **Authentification implicite** : aucun authentifiant n'est fourni lors de l'appel. Ceux-ci sont récupérés dans la session d'authentification de l'utilisateur effectuant l'appel (mise en œuvre du SSO Windows)

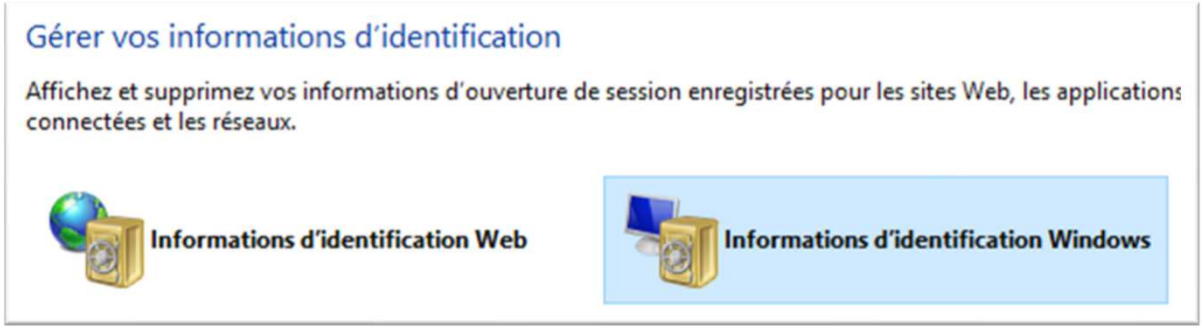
Authentification explicite

- Un nom d'utilisateur et un mot de passe doivent être spécifiés lors de la phase d'authentification

- Ce mécanisme permet le changement d'identité lors d'une authentification distante

Authentification sauvegardée

- Des authentifiants peuvent être sauvegardés dans le ***Credential Manager*** pour une cible donnée
- Si une authentification est demandée pour une cible où des authentifiants sont enregistrés, ils sont implicitement utilisés



Authentification implicite

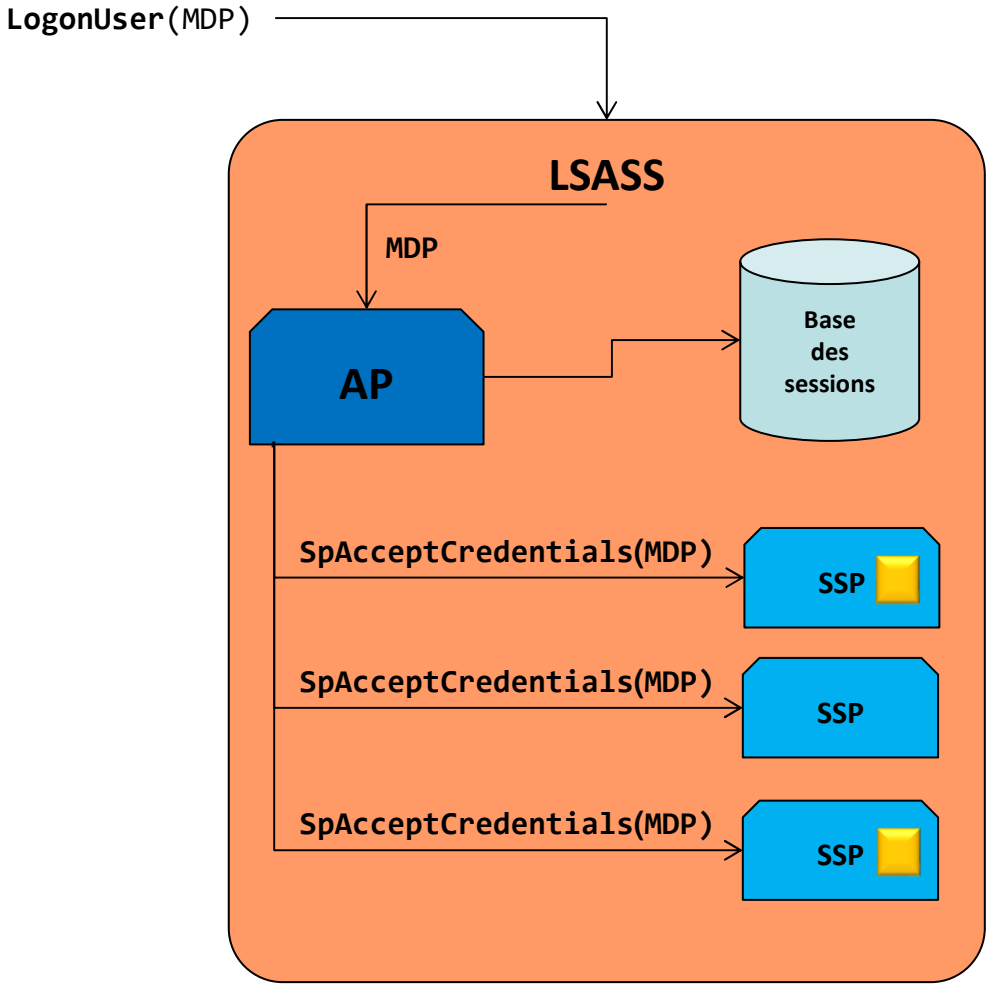
- Ce mécanisme de cache permet la mise en œuvre d'un SSO (*Single Sign-On*) Windows
- Lors de la création de la session d'authentification locale, les credentials de l'utilisateur sont mis en cache par les SSP afin d'être utilisés implicitement lors des authentifications distantes
- Ainsi, aucun nom d'utilisateur ni mot de passe n'est nécessaire lors des authentifications distantes réalisées par les SSP

Sauvegarde des *credentials*

- La récupération et la sauvegarde des *credentials* sont effectuées :
  - lors de l'appel de la fonction **`LogonUser`** pour un AP
  - via la notification **`SpAcceptCredentials`** qui est envoyée à tous les SSP enregistrés

- Un SSP peut sauvegarder les *credentials* :
  - Via les fonctions supports de LSASS (**`AddCredential`**)
  - Via un mécanisme propre au SSP

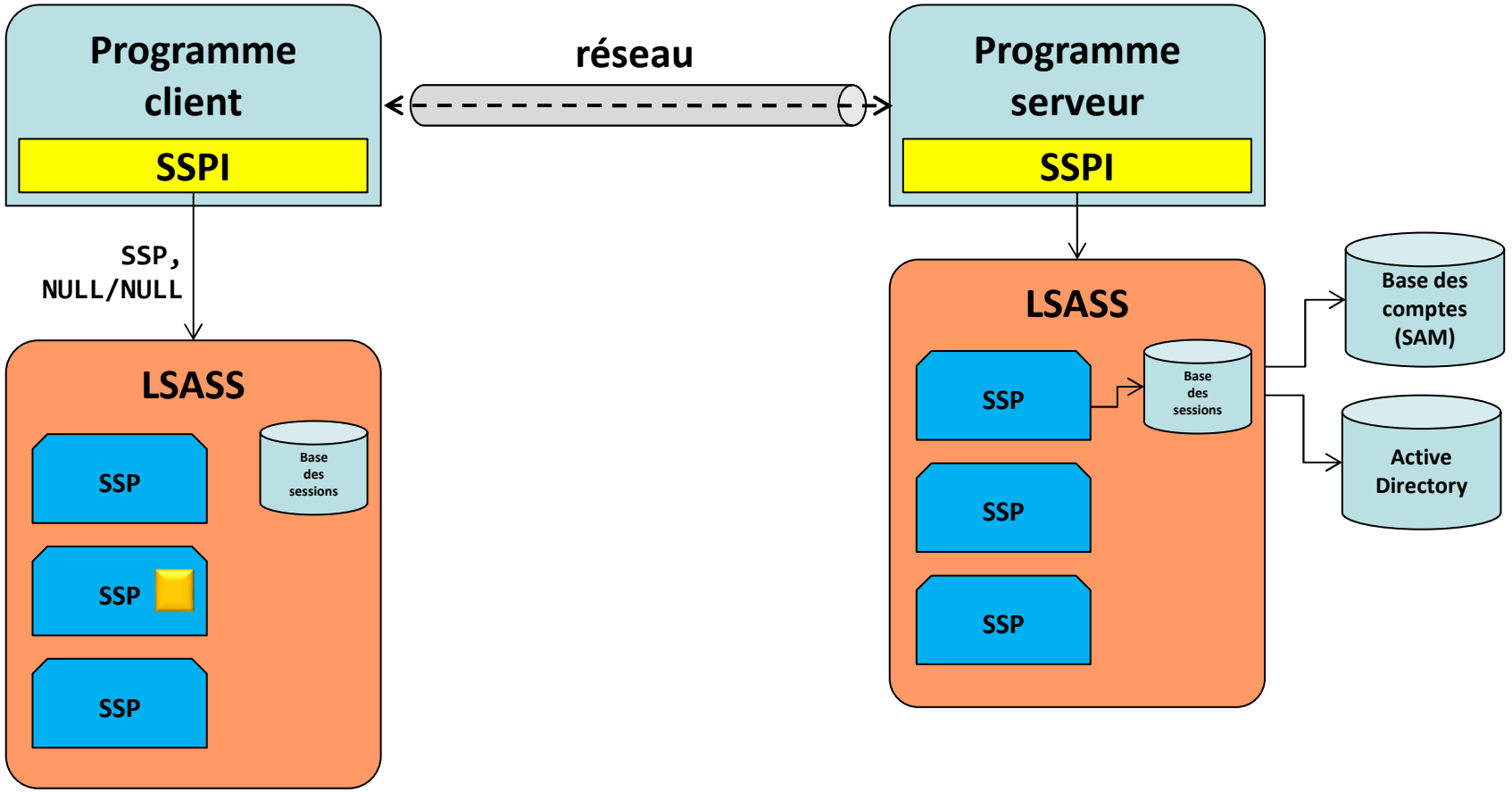
Sauvegarde des *credentials*



Tous les SSP sont notifiés et reçoivent le mot de passe en clair

Suivant leurs capacités, ils décident ou pas de mettre en cache les authentificants reçus

Implémentation authentification implicite



Secrets mis en cache ou conservés par les SSP

- **NTLM** : empreintes LM, NTLM et SHA-1
- **Kerberos** :
  - Mot de passe en clair (jusqu'à réception d'un TGT ce qui permet de valider le *salt* du compte dans le domaine)
  - Clés Kerberos (plusieurs formats)
  - Tickets reçus (+ clés de session) : TGT et TS
- **Wdigest** : mot de passe en clair
- **CredSSP** : mot de passe en clair

Windows 8.1/2012 R2 - KB2871997

- Avec Windows 8.1/2012 R2, plusieurs mécanismes visent à réduire le nombre de secrets en mémoire :
  - ***Protected Users Group*** : utilisation exclusive de Kerberos avec AES et interdiction de la délégation
  - Support client pour le ***Restricted Admin RDP mode***
  - Suppression des secrets après fermeture des sessions
  - Désactivation de WDigest (`UseLogonCredential`)
  - TSSSP ne stocke plus le mot de passe sauf si une délégation d'authentification est autorisée
- Ces fonctionnalités ont été adaptées sur les systèmes antérieurs (Windows 7 et ultérieurs) avec le KB2871997

Groupe *Protected Users Security*

Windows 8.1 / 2012 R2

- Avec Windows 8.1/2012 R2, les membres de ce groupe n'ont plus leurs authentifiants mis en cache par les SSP :
  - NTLMSSP : plus d'empreinte LM/NTLM
  - Kerberos : plus de clés (DES/LM/AES)
  - CredSSP/WDigest : plus de mot de passe en clair
- Seul un TGT (d'une durée de validité de 4 heures) est conservé utilisant exclusivement du chiffrement AES (plus de DES ou RC4)
- Impossibilité d'utiliser la délégation Kerberos (*constrained* ou *unconstrained*)
- *Backporté* aux systèmes Windows 7/2008 via le KB2871997
- RID : 525 / Niveau de fonctionnalité doit être 2012R2

Programmation des SSP

Mise en œuvre de l'authentification distante

- Les SSP sont accédés via la SSPI (*Security Support Provider Interface*)
- La fonction **`AcquireCredentialsHandle`** permet :
  - d'initialiser un SSP donné (côté client ou serveur)
  - de spécifier le contexte d'authentification ou de validation de l'authentification
- Le client génère les blocs d'authentification via la fonction **`InitializeSecurityContext`**
- Le serveur valide les blocs d'authentification via la fonction **`AcceptSecurityContext`**

Synopsis d'appel des fonctions de la SSPI