Analyse de la sécurité RDP : NLA, Quel apport pour votre sécurité ?

Geoffrey BERTOLI, Thomas BOURGUENOLLE

RDP est un protocole d'administration à distance, sa prédominance dans les environnements Windows en fait une cible de choix pour les attaquants.RDP en est actuellement à la version 10.5

Le point de départ de notre réflexion est d’étudier la possibilité d’effectuer des attaques de types homme du milieu sur l’authentification et le protocole RDP directement.

Pour rappel, il existe 3 modes d'authentification pour l'établissement d'une connexion RDP. Dans tous les cas, le client finira par envoyer ces authentifiant au serveur.

* ***RDP Standard Security***: Les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique. Cette clé étant négociée à partir d'un échange RSA établi avec une paire de clés fixe spécifié dans la documentation Microsoft. La mise en place d'une attaque de type MiTM est donc triviale et indétectable. RDP Enhanced Security :
* ***RDP Enhanced sécurity TLS*** (Apparu sous RDP 5.2 dans Windows Server 2003 SP1) : Les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique négociée dans un canal TLS. Les attaques de type Man-In-The-Middle nécessitent donc de posséder un certificat valide au sein de l'infrastructure Windows ou d'obtenir la validation du client pour continuer l'échange malgré l'erreur de certificat.
* ***RDP Enhanced security NLA*** (Apparu sous RDP 6.0 dans Windows Vista): C'est le mode d'authentification par défaut depuis Windows Server 2012R2. Dans ce mode, l'utilisateur réalise l'authentification dès l'ouverture de session RDP, la mire d'authentification Windows n'est donc pas affichée à l'utilisateur. Lors de l'authentification, les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique négociée dans un canal TLS après « *authentification mutuelle* » des deux acteurs (via Kerberos ou NTLMSSP). En pratique, l'implémentation de NLA est réalisée par CredSSP. Les attaques de type Man-In-the Middle semblent difficiles à mettre en place notamment lorsque l’authentification mutuelle est effectuée via Kerberos.

Suite à l’analyse des différentes modes d’authentification, une recherche de l’état de l’art a permis d’identifier les recherches suivantes :

L’outil SETH (<https://github.com/SySS-Research/Seth>) présenté à Hackitivity 2017, de réaliser une attaque de type Man-in-the-Middle sur une authentification RDP. Si l’authentification à lieu via RDP Enhanced security NLA, l’outil bloque les requêtes Kerberos du client puis indique que le DC ne répond pas pour forcer la connexion à se dégrader en RDP Enhanced security TLS. Suite à quoi un certificat non valide est présenté au client dans le but d’intercepter les authentifiant de la victime dans le cas où celui-ci accepte l’erreur de certificat. Il n’existe à l’heure actuelle aucun moyen de forcer l’utilisation de RDP Enhanced security NLA cotée client, cette attaque est donc toujours

L'outil PyRDP présenté (https://github.com/gosecure/pyrdp) en aout 2019 à BlackHat Arsenal, permet de générer un certificat autosigné afin de réaliser une attaque de Man-in-the-middle sur des environnements utilisant le mode d'authentification RDP Enhanced Security TLS. Cet outil permet l'enregistrement de session, l'injection de commande, le rejeu, ainsi que la capture d'entrée utilisateur et donc de secret d'authentification. Néanmoins cet outil ne supporte pas le mode d'authentification NLA.

L'article Sécurité de RDP par Aurélien BORDES, Arnaud EBALARD et Raphaël RIGO traite quant à lui de l'ensemble de la sécurité de RDP. Cet article présente notamment les différents modes authentification et détaille en quoi une attaque de type MiTM sur NLA reste possible moyennant la compromission d'un compte machine dans le domaine cible.

Dans le cas des outils existant à l’heure actuelle, aucun ne gère le mode d’authentification RDP Enhanced Security NLA, même si Seth possède un moyen de dégrader l’authentification. Toutefois depuis la vulnérabilité CVE-2019-0708, les recommandations de Microsoft sont de n’accepter (côté serveur) uniquement les authentifications via RDP Enhanced Security NLA et aucun des outils ne peut réaliser des connexions clients via NLA.

En tentant le développement d’un outil permettant d’effectuer des attaques de type Man-In-The-Middle supportant totalement l’authentification NLA. Nous avons découvert plusieurs problèmes dans ce mode d’authentification. Notre analyse présente ces faiblesses et notamment pourquoi l’utilisation de NLA met l’ensemble de l’infrastructure plus à risque qu’avec l’utilisation de RDP Enhanced Security TLS.

Ces vulnérabilités permettent notamment la récupération de réponse NTLMv2 de manière transparente pour la victime même dans le cas où celle-ci appartient au groupe "Protected Users".

**Comment fonctionne l’authentification NLA :**

L’authentification via NLA se déroule comme suit :

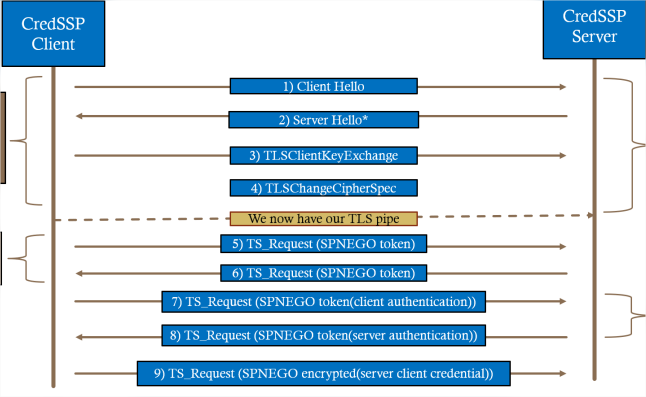


Figure 1 : Authentification NLA

Les 4 premières étapes consistent à établir une connexion TLS entre le client et le serveur. Cependant à la différence d’une authentification utilisant RDP Enhanced Security TLS, la validité du certificat n’est ici pas vérifiée. En effet le protocole CREDSSP est fait pour effectuer une authentification mutuelle du client et du serveur, la validité du certificat est donc optionnelle.

Les étapes 5 et 6 servent à négocier le SSP (Security Support Provider), celui-ci peut être NTLMSSP ou Kerberos.

* NTLMSSP est choisi : si le client n’appartient pas à un domaine ou si le client utilise l’adresse IP pour se connecter :
* Kerberos est choisi lorsque le client appartient au même domaine que le serveur **et** que le client utilise le FQDN de la cible pour se connecter.

Les étapes 7 et 8 servent à effectuer l’authentification mutuelle des deux acteurs.

L’étape 9 est l’envoi des authentifiant (login/mot de passe) du client auprès du serveur.

Pour la suite nous allons introduire les TSRequest qui sont la structure permettant le transport de l’ensemble des messages utilisées pour l’authentification CredSSP.

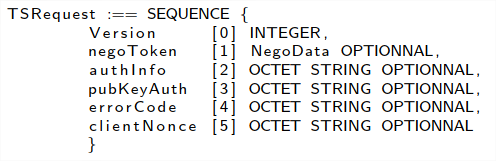


Figure 2 : Format d’une TS\_Request

*Concernant le champ version nous parlerons par la suite uniquement de la version 6 de CredSSP. En effet depuis la vulnérabilité CVE-2018-0886, les clients sont censés refuser les versions 5 et inférieurs de CredSSP. Cependant il a été vu que depuis l’update d’octobre 2019, la version 2 de CredSSP peut de nouveau être utilisé réintroduisant cette vulnérabilité de type Remote Code Exec (mais des analyses sont encore nécessaires de notre part). Les numéros d’étape sont repris depuis le schéma Figure 1.*

**Cas où nous négocions NTLMSSP comme Security Support Provider**

*Étape 5 :*

Le client envoi la requête suivante :

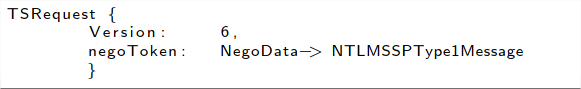


Figure 3 : Format d’une TS\_Request spécifique à l’étape 5

Le message NTLMSSPType1 contient des informations nécessaires pour l’établissement d’un échange NTLMSSP, contient notamment des informations sur la version NTLMSSP ainsi que les flags.

*Étape 6 :*

Le serveur répond comme suit :

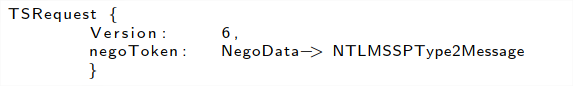


Figure 4 : Format d’une TS\_Request spécifique à l’étape 6

Le message NTLMSSPType2 contient les informations de challenge NTLM :

* NTLMChallenge
* Nom du serveur
* Nom du domaine

*Étape 7 :*

Le client envoi maintenant la réponse NTLMv2 :

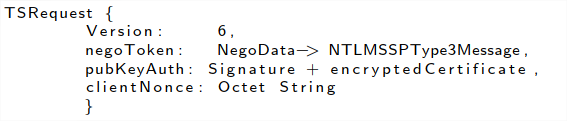


Figure 5 : Format d’une TS\_Request spécifique à l’étape 7

Le message NTLMSSPType3 contient la réponse au challenge NTLM contenant:

* NTLMv2 Response
* Username
* Nom de domaine
* Une EncryptedSessionKey. Cette valeur correspond au chiffrement via RC4 d’une RandomSessionKey avec une clé secrète. La clé utilisée pour chiffrer cette RandomSessionKey est un HMACMD5 dérivé de l'empreinte NTLM de l’utilisateur.

Le champ pubKeyAuth contient le chiffrement RC4 du SHA256 de (’CredSSP Client-To-Server Binding Hash’ + Nonce + PublicKey du certificat)) chiffré avec la Clé RandomSessionKey.

Le Nonce est renvoyé dans le champ ClientNonce

**À cette étape, un attaquant effectuant une attaque de type Man-In-The-Middle a récupéré une réponse NTLMv2 de sa victime sans lever aucune alerte de sécurité. Cette empreinte pouvant être cassée ou utilisée pour des attaques de type SMB Relay**

*Etape 8 :*

C’est à cette étape que le serveur est censé prouver son authenticité.

Pour cela il renvoie la requête suivante :

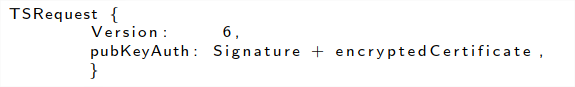


Figure 6 : Format d’une TS\_Request spécifique à l’étape 8

Le champ pubKeyAuth contient le chiffrement RC4 du SHA256 de (’CredSSP Server-To-Client Binding Hash’ + Nonce + PublicKey du certificat)) chiffrer avec la RandomSessionKey.

Pour effectuer cette requête, le serveur doit être en connaissance de la RandomSessionKey. Celle-ci est récupérée auprès du Domain Controller. Pour qu’un attaquant puisse mettre en place une attaque de type Man-In-The-Middle, il doit donc être en mesure de récupérer cette RandomSessionKey. Pour cela il lui suffit d’avoir un compte machine valide au sein du domaine.

L’authentification mutuelle effectuée si le SSP choisi est NTLMSSP n’est donc pas une véritable authentification mutuelle, mais authentifie uniquement que la machine avec laquelle le client discute appartient bien au domaine.

*Étape 9 :*

Contrairement à ce qui été identifié dans le papier de l’ANSSI, Sécurité RDP, il n’est plus possible d’obtenir l’étape et donc les authentifiant directement depuis RDP > 7.0.

En effet au lieu de passer à l’étape 9, l’authentification recommence à l’étape 1. À la différence que le client va vérifier le certificat à la fin de l’étape 2 et donc potentiellement une erreur de certificat.

NB : Si l’attaquant possède un compte machine, il est facile pour lui d’obtenir un certificat RDP signé par l’autorité de confiance de l’Active Directory. L’erreur sera uniquement un « name mismatch » ce qui est forcément le cas si l’utilisateur a rentré une adresse IP.

Après présentation de l’erreur de certificat, l’authentification recommence et va juste à l’étape 9.

À l’étape 9 le client envoie la requête suivante :

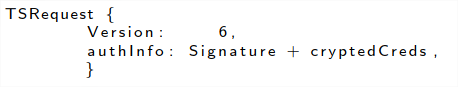


Figure 7 : Format d’une TS\_Request spécifique à l’étape 9

CryptedCreds contient les secrets d’authentifications de l’utilisateur chiffré via RC4 avec la RandomSessionKey.

**Cas où nous négocions Kerberos comme Security Support Provider**

Dans ce cas il n’est pas possible pour un attaquant en situation de Man-In-The-Middle de découvrir la RandomSessionKey, s’il ne possède pas le bon compte machine. En effet la RandomSessionKey est maintenant envoyée dans le TGS à destination du serveur qu’un attaquant ne pourra pas déchiffrer simplement avec un compte machine.

Cependant il est possible de partiellement downgrader la connexion pour revenir à une connexion utilisant NTLMSSP.

*Étape 5 :*

Le client envoi maintenant la TS\_Request suivante :

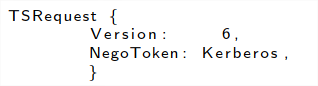


Figure 8 : Format d’une TS\_Request spécifique à la négociation Kerberos

On remarque que le client ne propose que Kerberos. Même si celui-ci supporte NTLMSSP, il ne le propose pas comme option auprès du serveur.

*Étape 6 :*

Il est tout de même possible pour un attaquant de répondre un paquet « magique » qui permet de downgrader la communication sur NTLMSSP. La TS\_Request est la suivante :

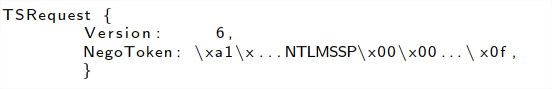


Figure 9 : Format d’une TS\_Request pour le downgrade de la communication

Suite à cette réponse, le client va autoriser la connexion via NTLMSSP mais le protocole est « cassé » et ne permet pas d’aller au bout de la connexion.

*Étape 5 (bis) :*

Le client va maintenant envoyé un message NTLMSSPType1 mais au lieu de le mettre directement dans le champ NegoToken, il va le mettre dans le champ AuthInfo contenu dans une TS\_Request présente dans le champ NegoToken :

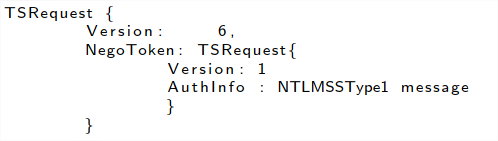


Figure 10 : Format d’une TS\_Request pour une renégociation NTLMSSP

*Étape 6 (bis) :*

Il est possible de répondre de la même manière pour envoyer un message NTLMSSPType2 auprès du client :

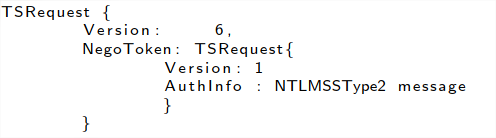


Figure 11 : Format d’une TS\_Request pour une renégociation NTLMSSP (étape 6)

*Étape 7 :*

**Le client va donc envoyer une réponse NTLMSSPType3 contenant la réponse NTLMv2 toujours sans message d’erreur sur le certificat et alors que celui-ci avait forcé l’utilisation de Kerberos. Cette empreinte pouvant être cassée ou utilisée pour des attaques de type SMB Relay**

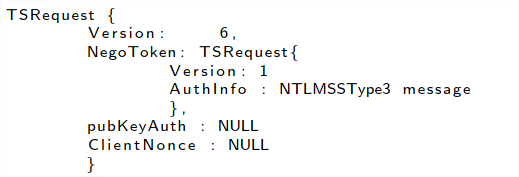


Figure 12 : Format d’une TS\_Request pour une renégociation NTLMSSP (étape 7)

L’absence de valeur dans le champ pubKeyAuth et ClientNonce, empêche l’authentification de terminer. Il n’est donc pas possible d’aller plus loin dans la communication et d’obtenir les authentifiant de la victime en clair comme c’est le cas pour le choix NTLMSSP.

Il est à noter que la victime enverra son empreinte NTLMv2 même si l’utilisateur appartient au groupe « protected users ». En effet d’après Microsoft, cette protection est faite pour protéger un utilisateur authentifié et non un utilisateur en cours d’authentification.

**Conclusion**

En conclusion l’utilisation obligatoire de NLA sur le réseau semble ajouter plus de risque au sein de l’infrastructure réseau que l’utilisation d’une authentification TLS.

En effet l’absence de vérification du certificat en première instance permet à un attaquant de récupérer une empreinte NTLMv2 sans lever d’alerte, quel que soit le SSP choisi (Kerberos ou NTLMSSP) .

En outre si le SSP utilisé est NTLMSSP alors les « chances » de réussite d’une attaque de type Man-In-The-Middle, sont les mêmes que pour une authentification TLS (acceptation du certificat par la victime) si l’on possède un compte machine au sein du domaine.

**Outil**

Un outil a été développé en python avec la bibliothèque impacket pour réaliser les POC permettant la mise en évidence de ces faiblesses.

En plus de cela l’outil propose une bibliothèque python permettant l’établissement d’une connexion RDP via authentification NLA (coté client et serveur). Cet outil est en cours d’intégration dans un outil plu complet type PyRDP permettant de faire des attaques directement sur RDP (injection de commandes, keylogger etc…)