Quelle sécurité appliquer sur le réseau

Analyse de réseau wireshark, arpscan

►BUT :

- On va récupérer et analyser les trames qui passent sur notre réseau au moyen d'un outil simple « Wireshark ».
- On va aussi débusquer des utilisateurs du réseau en utilisant un scanner « arpscan » de couche 2 (contrairement à nmap qui est plutôt de couche 3 donc utilisant des adresses IP) et distinguer les applications sécurisées et celles qui ne le sont pas, ce qui permet de faire des choix permettant de protéger sa vie privée.

Remarque : comme tout couteau qui peut être utilisé comme une arme ou comme une manière élégante de manger de la viande ou une pomme, les outils que nous allons manipuler servent autant à se connaître (informatiquement parlant) qu'à espionner des réseaux inconnus (et ceci est une action répréhensible par la loi).

► GLOSSAIRE :

- Wireshark : C'est un analyseur de paquets open source (licence GNU). Il est gratuit et fonctionne sur tous les systèmes (Windows, Linux, Mac...). Ce scanner graphique est la pierre angulaire de l'analyse de réseau.
- arp-scan : Scanner de couche 2, il sollicite les périphériques sur votre réseau et les obligent à répondre car l'adresse de destination est une adresse broadcast arp (adresse : ff:ff:ff:ff:ff:ff). Même une protection type pare feu ne peut empêcher cette réponse. Une exception cependant, un périphérique qui ne fait qu'écouter le réseau et qui ne communique pas, ou un natel peuvent être rigides aux requêtes ARP.
- Scapy : "Scapy est un logiciel libre de manipulation de paquets réseau écrit en python.Il est capable, entre autres, d'intercepter le trafic sur un segment réseau, de générer des paquets dans un nombre important de protocoles, de réaliser une prise d'empreinte de la pile TCP/IP, de faire un traceroute, d'analyser le réseau informatique... Créé par Philippe Biondi, actuellemnt développé par Pierre Lalet, Guillaume Valadon & Gabriel Potter" (fr.wikipedia.org/wiki/Scapy) <u>Autre références :</u>

Rodofile, N., Radke, K., & Foo, E. (2015). Real-time and interactive attacks on DNP3 critical infrastructure using Scapy,
 Biondi, P. (2010). Scapy documentation (!)

Maxwell, A. (2012). The very unofficial dummies guide to Scapy. *The IT Geek Chronicles*.

► La réponse à la question :

Il y a différents types de câbles sous-marins, les anciens câbles sont de type série et avec des débits lents, ils sont remplacés actuellement par des fibres optiques beaucoup plus rapides ce qui explique des temps très différents pour traverser l'Atlantique ou pour descendre le long de l'Afrique.

Réf : <u>https://www.submarinecablemap.com/</u> En cliquant sur le câble on connaît les lieux d'extrémité.



Les câbles sous-marins sont aussi espionnés : <u>https://bfmbusiness.bfmtv.com/entreprise/les-cables-sous-marins-meilleurs-amis-de-ceux-qui-nous-espionnent-897455.html</u> (2015) <u>https://ltemagazine.com/la-securite-des-cables-sous-marins-un-enjeu-majeur</u> (2018)

1



Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions

C'EST PARTI :

— Telnet est-il sécurisé ?—

Le Raspberry Pi est branché sur votre box, votre ordinateur est aussi sur le réseau filaire ou bien en Wifi peu importe.

Sur votre ordinateur connectez vous en VNC sur le Raspberry Pi. Vous pouvez fermer l'application p1 du TPA1, elle a servi simplement à lancer l'application Telnet, on verra à la fin du TP comment l'empêcher de ce lancer au démarrage.

Pour installer Wireshark il faut que la base d'information des packages soit à jour. Pour cela il faut, dans un terminal, passer la commande suivante : sudo apt-get update

puis la commande sudo apt-get install wireshark

Une fois l'installation terminée vous retrouvez wireshark dans le menu Applications-Internet- wireshark

Si vous cliquez le programme, on se trouve face à un problème, on ne retrouve pas notre interface Ethernet eth0 dans la liste des interfaces possibles à écouter (en interface graphique on n'est pas administrateur/root)

Deux solutions sont possibles :

Lancer wireshark en tant qu'admin dans un terminal : sudo wireshark & (& pour être en tâche de fond)
 On modifie le contenu du menu pour lancer wireshark correctement.

Les descriptions des applications des menus xfce4 (l'environnement graphique) sont dans le répertoire /usr/share/applications et sont nommées *nomApplication*.desktop

On va ouvrir le fichier wireshark.desktop : sudo nano /usr/share/applications/wireshark.desktop et modifier la ligne Exec=wireshark %f en Exec=sudo wireshark %f

puis sauvegarder et quitter : Ctrl+O Entrée Ctrl+x

Pour rajouter ou modifier un menu il faut au minimum le fichier suivant :

[Desktop Entry] Version=1.0 Type=Application Name=<Le_nom_de_l_application> Exec=<Le_chemin_vers_l_application> Icon=<L_icone_de_l_application> Categories=<La_catégorie_normalisée_du_menu>

Exemple de catégories : Network, Development... lire la catégorie du menu déjà installé dans lequel on veut raccrocher son propre menu.

On peut maintenant lancer wireshark correctement et on va analyser nos échanges telnet.

Lancez wireshark et double cliquer sur la ligne eth0.

Pour éviter d'avoir trop d'informations, on va filtrer les entrées de wireshark en écrivant telnet dans la barre de filtre et en validant par la touche Entrée

eth0		<u> </u>	xx		
any Loopback: lo		~	~		
wlan0					
			*e	th0	
Eichie	er <u>E</u> diter ⊻u	e <u>A</u> ller <u>C</u> apture	Analyser St	atistiques T	eleph

On se connecte en telnet (comme au TPA1) on entre son nom, on peut voir que ça enregistre côté Wireshark. Une fois que c'est terminé on clique sur le carré rouge de Wireshark pour arrêter l'enregistrement.



Quand on clique sur une trame (premier cadre en haut) on retrouve son découpage en couche osi (Open Systems Interconnection) dans le cadre en dessous.

Si on clique sur « Frame » on

« bleuit » la zone correspondante (tout) dans le troisième cadre qui fournit la trame en binaire à gauche et sa traduction en valeurs ASCII à droite par exemple 64 (hexadécimal) est représenté par d en ASCII.

En cliquant sur la deuxième ligne on sélectionne la partie correspondant à la couche Ethernet,

On y retrouve les adresses MAC source Src et destinataire Dst et le type. C'est une trame Ethernet.

En choisissant sa trame on peut voir la demande du prénom de la part du serveur telnet. On voit en bas à droite QUE LE DIALOGUE EST EN CLAIR (on peut lire « Qu el est votre pre nom?»

Sélectionner une trame et cliquer le menu « Flux TCP », on peut voir la totalité des échanges de la session.

Filtre: Affich	s d'affichage er les Filtres <u>M</u>	jacros		0	
Applio Applio Prépa Filtre	quer en Colonn quer comme u rer un Filtre de Conversatio	ne Ct n Filtre on	rl+Maj+I	, tr 6 6	Info 52751 - 23 23 - 52751 52751 - 23
Proto Décor	coles activés der <u>C</u> omme		rl+Maj+E	4 8 9 5	23 → 52751 Telnet Data 52751 → 23 Telnet Data
SCTP	irger Flugins c	ua ci	птмајтс	, 2	52751 → 23 Telnet Data
					Flux TCP
Show	Packet Bytes. nation Expert	a	rl+Maj+O		Flux UDP Flux SSL



No.	Time	Source	Destination	Protocol L	ength Info	
3	44 347.377695746	192.168.100.28	192.168.100.45	TELNET	75 Telnet Dat	ā
34	40 347.557171749 53 352 569895576	192,168,100,45	192,168,100,28	TELNET	55 Telnet Dat	a
3	58 357.316838150	192.168.100.28	192.168.100.45	TELNET	62 Telnet Data	а
34	80 357.316844556	192.168.100.28	192.168.100.45	TELNET	60 Telnet Data	а
Fran	e 344: 75 bytes	on wire (600 bit	s), 75 bytes captured (600 bits) on	interface 0	
Ethe	rnet II, Src: W	1stronC_a5:de:60	(30:65:ec:a5:de:60), Ds	t: Raspberr_	ae:89:76 (b8:27:	eb:ae:89:76
Tran Telr	ernet Protocol v Ismission Contro Net	1 Protocol, Src: 19	ort: 52751, Dst Port: 2	3, Seq: 1, A	ick: 1, Len: 21	
> W	ill Negotiate A Ill Terminal Sp	bout Window Size eed				
• W	111 Terminal Ty	pe ment Ontion				
+ D	o Echo	and operation				
* W	ill Suppress Go	Ahead				
, ,	o Suppress Go A	nead				
9996	b8 27 eb ae 89 7	'6 30 65 ec a5 de	60 08 00 45 00 ·····v	0e ···· E·		
3010	00 3d 78 a0 40 0	0 80 06 38 80 c0	a8 64 1c c0 a8 =x 0	- 8 d		
9628 2030	49 20 CE 01 00 1	0 ff fb 1f ff fb	20 ff fb 18 ff @)	· · · · · = · ₽ ·		
1948	fb 27 ff fd 01 f	f fb 03 ff fd 03	2011101011	120223		
			11.010-010-010-010-01-01-01-01-01-01-01-01-			
No.	Time	Source	Destination	Protocol L	ength Info	
3	\$44 347.37769574	9 192.168.100.28	192.168.109.45	TELNET	75 Teinet Data	
	353 352.56909557	6 192.168.100.45	192.168.100.28	TELNET	55 Telnet Data	
3	358 357.31683815	9 192.168.100.28	192.168.100.45	TELNET	62 Telnet Data	***
3	360 357.31684455	6 192.168.100.28	192.168.100.45	TELNET	60 Telnet Data	
Fra	me 346: 78 byte:	s on wire (624 bit	s), 78 bytes captured (324 bits) on	interface 0	
Eth	ernet II, Src:	Raspberr_ae:89:76	(b8:27:eb:ae:89:76), Ds	WistronC_	a5:de:60~(30:65:	ec:a5:de:60)
Tra	nemicsion Contro	1 Protocol Src P	ort: 22 Det Dort: 5275	Sog: 1 A	ck: 22 Lon: 24	
- Tel	net					
	Data: Quel est v	votre prenom?\n				
	Data: \r					
8000	30 65 ec a5 de	60 b8 27 eb ae 89	76 08 00 45 00 00	A Strategy B		
0010	00 40 d9 13 40	00 40 06 18 0a c0	a8 64 2d c0 a8 0 0	9d		
8836	01 f6 49 cd 00	80 51 75 65 6c 20	65 73 74 20 76 ··· I····	u el est v		
6846	6f 74 72 65 20	70 72 65 6e 6f 6d	3f 0a 0d otre p	re nom?··		
1		Wiresbark · Follo	w TCP Stream (tcp.stream e	1) · eth0		0 - A
						Contras and
	Qu	el est votre prenom?				
Jean	Luc					
Bonjou	ir jean, c'est une	server telnet bride	pour des questions de sec	urite.		
Nous v	errons cet aspect	le prochain TP.				
		12				
	50					
Pour q	uitter, fermer si	mplement la fenetre.				
Entrer	au clavier 'help	' voir les commandes	autorisees.			
bol-	6					
Liste	de commandes poss	ibles :				
Presed	- DOUR WATE FORDE	toire courset				
hug	- pour vorr reper	corre courant				
1s -	pour lister le c	ontenu du repertoire	/home/pi			
cat	- pour voir le co	ntenu du fichier /ho	me/pi/Exo1			
*cat	de inconnue '*cat					
	odt.					
>b"Tex	te pour l'exercic	e 1"				
		Sec.			т	
>0.110	ne 1'				1	
>b'110	ine 2'					
9 client p	kt(s), 17 server pkt(s), 9	tours.				
Entire	conversation (595 by	tes)	 Show and s 	ave data as AS	CII +	Flux 1
Trouver						Trouver Suivar
2000078			nesento II composi II.			
		Filter Out T	his Stream Imprimer	Save as	Back X Ferme	er Aide

On imagine très bien que lors de la demande de mot de passe par le switch ou le routeur, si la liaison est en Telnet toute personne lisant les trames aura votre mot de passe. On évitera donc, quand on est à distance, d'utiliser Telnet qui n'est pas sécurisé.

Effacer le fichier /home/pi/Exo1 pour supprimer le démarrage automatique de l'application p1



— Ssh est-il sécurisé ?—

On relance une capture de trame (Menu Capture \rightarrow Démarrer), on ne veut pas sauvegarder les derniers échanges Telnet donc on clique « Continue without saving »). On filtre en écrivant « ssh ».

On lançe une connexion ssh depuis Putty sur votre PC comme au TPA1.

On peut voir qu'après une phase de discussion en clair (non chiffrée pour choisir le chiffrement) les échanges sont ensuite chiffrés (« Encrypted paquet ») et donc indéchiffrables

0.	Time	Source	Destination	Protocol	Length Info		
	62 48.362155031	192.168.100.28	192.168.100.45	SSHv2	82 Client:	Protocol (SSH-2.	0-PUTTY Re
	64 48.517568936	192.168.100.45	192.168.100.28	SSHv2	97 Server:	Protocol (SSH-2.	0-OpenSSH
	65 48.525438466	192.168.100.28	192.168.100.45	SSHv2	1158 Client:	Key Exchange In:	t
	67 48.531026673	192.168.100.45	192.168.100.28	SSHv2	1134 Server:	Key Exchange In:	t
	68 48.550701044	192.168.100.28	192.168.100.45	SSHv2	102 Client:	Elliptic Curve ()iffie-Hell
	70 48.595488569	192.168.100.45	192.168.100.28	SSHv2	262 Server:	Elliptic Curve [Diffie-Hell
	72 48.848821213	192.168.100.28	192.168.100.45	SSHv2	70 Client:	New Keys	
	74 48.848823922	192.168.100.28	192.168.100.45	SSHv2	118 Client:	Encrypted packet	(len=64)
	76 48.849202354	192.168.100.45	192.168.100.28	SSHv2	118 Server:	Encrypted packet	: (len≃64)
	88 61.095846531	192.168.100.28	192.168.100.45	SSHv2	134 Client:	Encrypted packet	: (len=80)
Fri Eti In Tra SSI	ame 74: 118 bytes hernet II, Src: W ternet Protocol V ansmission Contro H Protocol SSH Version 2 (e Packet Length	s on wire (944 bits) VistronC_a5:de:60 (2 Version 4, Src: 192. D1 Protocol, Src Por encryption:aes256-ct (encrypted): 68b75), 118 bytes captured 30:65:ec:a5:de:60), [168:100.28, Dst: 190 rt: 52845, Dst Port: rr mac:hmac-sha2-256 4b7	1 (944 bits) Ost: Raspber 2.168.100.45 22, Seq: 11 compression	on interface r_ae:89:76 (b8 97, Ack: 1332, :none)	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI	ame 74: 118 bytes hernet II, Src: V ternet Protocol V ansmission Contro H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7	s on wire (944 bits) distronC_a5:de:60 (2 /ersion 4, Src: 192. ol Protocol, Src Por encryption:aes256-ct (encrypted): 68075 ket: 64e5b262661489 4eb2556af4cec8c70cf	<pre>), 118 bytes captured 30:65:ec:a5:de:60), [168.100.28, Dst: 192 rt: 52845, Dst Port: rr mac:hmac-sha2-256 4b7 5c078b229fc64b9535f2 ef9ed23f23b0b5acf81f</pre>	d (944 bits) st: Raspber 2.168.100.45 22, Seq: 11 compression b8e77a743a3e 9	on interface r_ae:89:76 (b8 97, Ack: 1332, :none)	0 :27:eb:ae:89:76) Len: 64	
Fri Eti In Tr: SSI	ame 74: 118 byte: hernet II, Src: W ternet Protocol V ansmission Contro H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89	s on wire (944 bits) distronC_a5:de:60 (2 dersion 4, Src: 192. pl Protocol, Src Pou- encryption:aes256-ct (encrypted): 68b75 ket: 64e5b262661489 4eb2556af4ce8c70eC 76 30 65 ec a5 de 6	<pre>), 118 bytes capture; 30:65:ec:a5:de:60), [168:100.28, Dst: 102, rt: 52845, Dst Port: rr mac:hmac-sha2-256 4b7 5c078b229fc64b9535f2 ef9ed23f23b0b5acf81f i0 08 00 45 00</pre>	d (944 bits) Dst: Raspber 2.168.100.45 22, Seq: 11 compression b8e77a743a3e 9 v0e E	on interface r_ae:89:76 (b8 97, Ack: 1332, none)	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI	ame 74: 118 byte: hernet II, Src: W ternet Protocol V ansmission Contro H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89 00 68 7f 9c 40	s on wire (944 bits) distronc 45.4e:60 (2 /ersion 4, Src: 192. s) Protocol, Src Por- mcryption:aes256.ct (encrypted): 68b75 ket: 64e5b262661489 4eb2556af4ce88.78cf 76 30 65 ec a5 de 6 08 80 06 31 59 c0 a), 118 bytes capture 30:65:ec:a5:de:60), 1 168.100.28, Dst: 102, tt: 52845, Dst Port: r mac:hmac-sha2-256 4b7 5c078b229fc64b9535f2 ef9ed23f23bbb5acf81f i0 08 00 45 00 	d (944 bits) st: Raspber 2.168.100.45 22, Seq: 11 compression b8e77a743a3e 9 v0e E a 1Y d	on interface r_ae:89:76 (b8 97, Ack: 1332, :none) :2c	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI 000 010 020	ame 74: 118 byte: ternet Protocol V ansmission Contro H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89 00 68 7f 9c 40 66 2d ce 6d 00	s on wire (944 bits) distronc as:de:60 (2 /ersion 4, Src: 192 b) Protocol, Src Por uncryption:aes256-ct (encrypted): 68b75 ket: 64e5b262661489 4eb2556af4ece870eC 76 30 65 ec a5 de 6 90 80 06 31 59 c0 a 16 09 73 15 78 0f f), 118 bytes captures 30:65:ec:a5:de:60), 10 168:100.28, Dst: 10; rt: 52845, Dst Port: 5:078b229fc64b9535f2 ef9ed23f23b0b5acf81f i0 08 00 45 00 1:8 64 1c 60 a8 -h 4 b 28 6c 50 18 d -m	d (944 bits) st: Raspber 2168.100.45 22, Seq: 11 compression b8e77a743a3e 9 v0e E 0 1Y d s x (1P	on interface r_ae:89:76 (b8 97, Ack: 1332, :none) :2c	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI • 010 020 030	ame 74: 118 byte: hernet II, Src: k ternet Protocol V ansmission Conto SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89 06 68 7f 9c 40 64 2d ce 6d 06 3e dc ff 90 90	s on wire (944 bits) distronc as:de:60 (2 /ersion 4, Src: 192.)) Protocol, Src Por- mcryption:aes256.ct (encrypted): 68b75 ket: 64e5b262661489 4eb2556af4cec8c70cf 76 30 65 ec a5 de 6 08 00 66 31 59 c0 a 16 09 73 15 78 0f f 30 68 b7 54 b7 64 e), 118 bytes capture; 30:65:ec:a5:de:60), 10 1:68:100.28, Dst: 19; 7t: 52845, Dst Port: 407 5:c078b2297c64b953572 ef9ed23f23b0b5acf81f 10 08 00 45 00 1:b26 26 08 1:b26 42 c6 08 1:b26 26 01 4 5 b2 62 66 14	d (944 bits) Dst: Raspber 22, Seq: 11 compression b8e77a743a3e 9 v0e E g 1Y d s x (1P h T d bf	on interface r_ae:89:76 (b8 97, Ack: 1332, :none) *2c	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI 010 020 030 040	ame 74: 118 byter hernet II, Src: V ternet Protocol V ansmission Contra H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89 00 68 7f 9c 40 64 2d ce 6d 00 3e dc ff 19 00 89 5c 07 8b 22	s on wire (944 bits) Wistron(a.5:de:60 (: /ersion 4, Src: 192)1 Protocol, Src Por- moryption:aes256.ct (encrypted): 68b75 Ket: 64e5b2664489 4eb2556af4ec8c78cf 76 30 65 ec a5 de 6 90 80 80 6 31 59 co a 16 90 73 15 78 0f 190 68 b7 54 b7 64 e 91 6c 40 95 35 72 b), 118 bytes captures 30:65:ec:a5:de:60), 1 1.68:100.28, Dst: 10; rt: 52845, Dst Port: 5c078b229fc64b9535f2 ef9ed23f23b8b5acf3f1 i0 08 00 45 00 1 28 6c 50 18 d - m 15 b2 6c 65 18 d - m 15 b2 6c 65 18 d - m 15 b2 6c 65 14	1 (944 bits) Dst: Raspber 2168 100 45 22, Seq: 11 compression b8e77a743a3e 9 v0e E 9 1Y d s x (1P -h T d bf -K 5 - zt	on_interface r_ae:89:76 (b8 97, Ack: 1332, :none) :2c	0 :27:eb:ae:89:76) Len: 64	
Eti In Tr: SSI 010 020 030 030 030 030 030	ame 74: 118 bytes hernet II, Src: k ternet Protocol 1 Ansmission Contro H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 89 00 68 7f 9c 40 64 2d ce 6d 00 3e dc ff 19 00 89 5c 07 8b 22 3e 2c eb 38 ad	s on wire (944 bits) WistronCa.5tde:60 ((Version 4, Src: 192, 1) Protocol, Src Pou- nerryption:aes256-ct (encrypted):68b75 Ket: 645022661489 4eb2556af4ce3c764c 76 30 65 ec a5 de 6 08 80 66 35 5 ct 15 9 co 08 80 fb 35 7 ct 97 c6 4b 95 35 72 b 97 c7 4b 15 c 74 b 10 c 7 4b 15 c 20 c), 118 bytes captures 30:65:ec:a5:1de:00), 10 1:68:1de:0.28, Dst: 19; 1t: 52845, Dst Port: 5:078b229fc64b9535f2 ef9ed23f23b0b5acf81f 50 08 00 45 00 16 04 10 08 01 4 00 15 26 26 50 18 d-m 15 26 26 50 18 - 10 15 25 56 af 4 c >, 8	J (944 bits) Dst: Raspber 2.168.100.45 22, Seq: 11 compression b8e77a743a3e 9 v0e E 0 1Y d 5 x (1P h T d bf " K 5 zt å N UJ	on_interface r_ae:89:76 (b8 97, Ack: 1332, :none) :2c	0 :27:eb:ae:89:76) Len: 64	
Eri Eti In Tr: SSI 010 010 020 030 040 050 050 050	ame 74: 118 byter hernet II, Src: k ternet Protocol Ansmission Contra H Protocol SSH Version 2 (e Packet Length Encrypted Pac MAC: daa226c7 b8 27 eb ae 80 00 68 7f 9c 40 64 2d ce 6d 00 3e dc ff 19 00 89 5c 07 8b 22 3e 2c eb 38 at c8 c7 6c fe f9	s on wire (944 bits) Wistron(a.5:de:60 (: /ersion 4, Src: 192)] Protocol, Src Por- neryption:aes256-ct (encrypted): 68b75 ket: 6465262661489 4eb2556af4ec82.78cf 76 30 65 ec a5 de 6 00 80 66 31 59 0.35 72 b 80 68 7 54 b7 64 e 91 66 40 95 35 72 b 88 da a2 25 c7 4e b 88 da a2 25 c7 4e b 04 23 f2 30 6b 5a c), 118 bytes capture 30:65:ec:35:de:60), 1 1.68:100.28, Dst: 19; rt: 52845, Dst Port: 5c078b229fc54b9535f2 ef9ed23fc54b953cf34 10:08 00 45 00 28 66 40 f 28 66 50 18 d - m 55 b2 62 66 18 d - m 55 b2 62 66 14 8 67 7a 74 a 25 55 6a f4 ce 7 81 f9 18 ee	I (944 bits) Dst: Raspber .168.100.45 22, Seq: 11 compression b8e77a743a3e 9 v0e E 9 iY d s x (IP h T d bf " K 5 zt & N UJ # ; Z	on interface r_ae:89:76 (bB 97, Ack: 1332, :none) :2c	0 :27:eb:ae:89:76) Len: 64	

<u>Conclusion :</u> Il faut préférer une connexion Ssh lorsqu'on se connecte à distance à un périphérique qu'il soit switch, routeur, PC, IoT ... Vous bénéficiez alors d'une intégrité des données (elles ne sont pas modifiées, sinon on le détecte), de la confidentialité des données (à cause du chiffrement), et de l'authentification (vous avez sauvegardé la clé publique du périphérique lors de la première connexion). Cette clé publique identifie de manière quasiment unique votre périphérique et ne vous sera plus proposée sauf si elle a été changée.

— Qu'en est-il en http ?—

On lance une connexion http depuis un navigateur sur votre PC comme au TPA1 (en utilisant l'adresse IP de votre RaspberryPi dans la barre d'adresse).

On peut voir que la discussion est en clair.

On reconnaît très bien le code html de la page visualisée dans le navigateur.

l t	ttp					×	Expression	+
No.	Time	Source	Destination	Protocol	Length Info			
+	15 6.190091384	192.168.100.28	192.168.108	.45 HTTP	460 GET / 1	HTTP/1.1		
-	38 6.198655318	192.168.100.45	192.168.100	.28 HTTP	608 HTTP/1	.1 200 OK (t	ext/html)	
2	36 6.244397409	192.168.100.28	192.168.100	.45 HTTP	381 GET /fa	avicon.ico HT	TP/1.1	
	38 6.245110215	192.168.100.45	192.168.106	.28 HTTP	383 H11P/1	.1 404 Not ⊨o	und (text/ht	(m1)
FEIT	rame 30: 608 byte: hternet II, Src: i nternet Protocol ' ransmission Contri 13 Reassembled TG ypertext Transfer interbased toot und 	s on wire (4864 bits Raspber ac:89:76 (b) Version 4, Src: 192, ol Protocol, Src Por P Segments (18074 by Protocol th: text/ntrol (14 11 Nn Réponse http<, page d'exemple avec), 608 bytes c 8:27:ebiae:89: 168:100.45, Ds 1: 80, Dst Por tes): #17(1460 ne5) //title>\n nginx\n	aptured (4864 bif T6), Dst: Wistron t: 192,168,100,22 t: 52857, Seq: 17), #18(1460), #19	:s) on interfac DC_a5:de:60 (30 3/521, Ack: 407, 3(1460), #20(14	ce 0 9:65:ec:a5:de , Len: 554 460), #21(146	:60) 0), #22(1460)	, #1
	[truncated]	<img style:<="" th=""/> <th>="position: ab</th> <th>solute; margin: a</th> <th>uto; top: 0; 1</th> <th>left: 0; right</th> <th>:: θ; bottom:</th> <th>0; •</th>	="position: ab	solute; margin: a	uto; top: 0; 1	left: 0; right	:: θ; bottom:	0; •
t								•
8001 8021 8021 8031 8031 8031 8031 8051 8051 8061 8061 8081	3c 21 44 47 43 3c 68 74 6d 6c 3e 6a 20 20 20 3e 52 26 65 61 68 74 74 26 5 61 68 74 74 78 3c 55 64 79 3e 0a 28 3c 55 66 65 20 76 85 64 79 3e 0a 28 3c 76 3c 3c	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	6d 6c 3e 8a 68 65 61 64 69 74 6c 65 6e 73 65 20 9a 29 20 20 9a 20 3c 62 6f 9c 3c 68 32 3e 78 65 6d 70 78 3c 2f 68	<	d e so so p n			21
Fra	me (608 bytes) R	eassembled TCP (18074	bytes) De-c	hunked entity body i	(17911 hutor)	 Best to an investigation to be a second and 	and the second	
	the second s			inannea energy boay	(1) oll bytes)	Uncompresse	d entity body (2	23719

<u>Conclusion :</u> Le protocole http est en clair, pour conserver la confidentialité de vos données utilisez le protocole https qui est sécurisé (un peu comme le ssh).

<u>Question</u> : Dans le protocole https quelle est l'utilité du cadenas en dehors du fait qu'il vous offre une conversation sécurisée (chiffrée) ?



- ARP Address Resolution Protocol -

On vient de voir qu'un analyseur de réseau peut nous fournir beaucoup d'informations sur les trames qui circulent sur un réseau. On peut aussi débusquer des périphériques en utilisant un scanner ARP (Protocole de Résolution d'Adresses).

Dans le TPA précédent nous avions utilisé un scanner IP (nmap) qui travaille principalement avec les adresses IP mais la réponse des périphériques peut être masquée par un firewall ou un dispositif de filtrage.

L'intérêt de travailler avec le protocole ARP c'est que toute machine voulant parler sur le LAN est obligée d'utiliser les adresses MAC et de répondre au protocole ARP (en IPV4). On retrouve les adresses physiques (MAC) dans la trame Ethernet et Wireshark nous les affiche dans la partie hexadécimale, il traduit la partie constructeur de l'adresse et recopie les trois autres octets de numéro de série, dans la fenêtre du milieu concernant le protocole Ethernet.



Les adresses MAC sont importantes pour le fonctionnement des trames Ethernet (couche 2). Chaque périphérique va donc les mémoriser pour une utilisation ultérieure.

Un routeur ou un périphérique de type PC va mémoriser le couple (adresse IP, adresse MAC) dans une table. Ils gagneront du temps pour envoyer une trame à un destinataire si le couple (ad IP dest, ad MAC dest) existe.

Un switch (couche 2) va mémoriser le couple (ad MAC, numéro du port du switch d'où provient cette adresse), de manière à trouver rapidement vers quel port envoyer une trame ayant l'adresse MAC associée.

— Scanner un réseau en utilisant la couche 2 —

Pour scanner un réseau en couche 2, donc en utilisant les adresses physiques (MAC), on va dans un premier temps utiliser l'application arp-scan.

Dans une fenêtre de commande après avoir mis à jour la base de connaissances,on télécharge/installe arp-scan.

sudo apt-get update sudo apt-get install arp-scan

pi@raspberrypi:~ \$	sudo arp-scan 192.168.	100.0/24				
Interface: eth0, datalink type: EN10MB (Ethernet)						
Starting arp-scan 1.	9.5 with 256 hosts (https	://github.com/royhills/arp-scan)				
192.168.100.53	30:65:ec:a5:02:60	Wistron (ChongQing)				
192.168.100.31	e8:6a:64:f1:d7:2d	(Unknown)				
192.168.100.65	14:0c:76:4b:04:17	FREEBOX SAS				
192.168.100.21	b8:27:eb:d0:12:53	Raspberry Pi Foundation				
192.168.100.27	00:21:b7:3d:a5:e7	Lexmark International Inc.				
192.168.100.78	00:01:e6:2f:c8:05	Hewlett Packard				
192.168.100.25	14:0c:76:af:51:c4	FREEBOX SAS				
192.168.100.30	88:29:9c:5a:bd:61	(Unknown)				

Pourquoi scanner en couche 2 ? A priori les appareils voulant communiquer doivent se faire connaître au niveau du LAN et donc répondent aux sollicitations en couche 2. Ce type de scan est complémentaire au



scanner de niveau 3 de type nmap. Il faut noter que les natels n'ont pas tendance à répondre à ces demandes qu'elles arrivent en couche 2 ou en couche 3.

Pour connaître le fonctionnement de ce type de scan on peut regarder les échanges avec Wireshark, avec un filtrage "arp".

Frame 725: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

- ▼Ethernet II, Src: Raspberr_08:df:38 (b8:27:eb:08:df:38), Dst: Broadcast (ff:ff:ff:ff:ff:ff:ff:ff:ff)
 - ► Destination: Broadcast (ff:ff:ff:ff:ff:ff:ff:ff) Source: Raspberr_08:df:38 (b8:27:eb:08:df:38)
 - Type: ARP (0x0806)

▼Address Resolution Protocol (request)

Hardware type: Ethernet (1) Protocol type: IPv4 (0x0800) Hardware size: 6 Protocol size: 4 Opcode: request (1) Sender MAC address: Raspberr_08:df:38 (b8:27:eb:08:df:38) Sender IP address: 192.168.100.22 Target MAC address: 00:00:00 00:00:00 (00:00:00:00:00:00) Target IP address: 192.168.100.1

En résumé, en couche 2 on met en destination une adresse physique de broadcast (ff:ff:ff:ff:ff:ff:ff) et notre adresse MAC comme source. En couche 3 on utilise le protocole arp avec une demande "arp request" dans laquelle le couple (ad MAC source/ad IP source) c'est vous (Rpi), et le couple de destination (ad MAC/ad IP) possède une ad MAC inconnue (00:00:00:00:00:00).

La réponse comportera le couple de destination (ad MAC/ad IP) renseigné.

— Scanner "fait maison" avec SCAPY —

Il faut installer le bibliothèque Scapy en admin car beaucoup d'actions liées au réseau demandent à être administrateur.

sudo pip3 install scapy

ou sudo -s puis pip3 install scapy

Pour tester les éléments dont on a besoin dans la bibliothèque scapy, dans un premier temps on va simplement utiliser le shell de python3

sudo python3

La bibliothèque scapy doit être importée pour pouvoir l'utiliser

>>> from scapy.all import *

Chaque protocole possède une classe qui lui est dédiée. En regardant la sortie Wireshark du paragraphe précédent, nous aurons besoin de la classe qui gère la couche 2 (Ethernet) et celle qui gère le protocole ARP. On commence par Ethernet, la classe est Ether() et pour voir les propriètés de l'objet on utilise la méthode show()

nethoue c		
>>> c2 =	Ether()	
>>> c2.sł	now()	
###[Eth	ernet]###	
dst	= ff:ff:ff:ff:ff	f:ff
src	= b8:27:el	o:08:df:38
type	$= 0 \times 9000$	

On remarque que l'adresse MAC de destination est déjà une adresse de boadcast, l'adresse source est celle du Raspberry Pi. Le type "Ethertype" à 0x9000 correspond à la description de la loopback (https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml). Pour mettre la valeur on peut modifier l'abiat una faia aráá i c2.type=0x0806

robjet une	lois ciee.	$\sim \sim $
ou lors de l	la création :	>> c2 = Ether(type=0x0806)
On procède	e de la même façon	pour la couche ARP :
>>> arp =	ARP()	
>>> arp.sł	now()	
###[ARP]###	
hwtype	= 0x1	
ptype	= IPv4	
hwlen	= None	
plen	= None	
ор	= who-has	



hwsrc	= b8:27:eb:08:df:38
psrc	= 192.168.100.22
hwdst	= 00:00:00:00:00:00
pdst	= 0.0.0.0

On remarque que l'on a déjà une requête ARP de découverte (who-has dans la propriété op). L'adresse MAC et l'adresse IP source sont déjà remplies, l'adresse MAC de destination aussi (inconnue donc 00:00:00:00:00:00). L'adresse IP de destination devra être complétée, les autres propriétés seront complétées automatiquement.

On construit notre trame en utilisant l'opérateur / qui encapsule les objets: >>> trame = Ether()/ARP(pdst="192.168.100.0/24")

>>> trame =	Ether()/ARP(pdst="192.168
>>> trame.s	how()
###[Etherne	et]###
dst =	ff:ff:ff:ff:ff:ff
src =	= b8:27:eb:08:df:38
type =	ARP
###[ARP]#	##
hwtype	= 0x1
ptype	= IPv4
hwlen	= None
plen	= None
ор	= who-has
hwsrc	= b8:27:eb:08:df:38
psrc	= 192.168.100.22
hwdst	= 00:00:00:00:00:00
pdst	= Net('192.168.100.0/24')

Au lieu de lui demander une adresse IP de destination spécifique on lui donne la plage à scanner avec le CIDR (/24), avec le show() on voit que l'objet à bien compris qu'il s'agit d'un réseau (pdst = Net...). La propriété EtherType a été modifiée automatiquement (type = ARP).

Il ne reste plus qu'à envoyer la trame et à récupérer les trames en retour. Plusieurs méthodes existent :

envoi/réception de couche 3

La méthode sr() qui retourne le couple (paquets avec réponse / paquets

sans réponse).

La méthode sr1() qui est identique à sr() mais pour un seul paquet envoyé.

> envoi/réception de couche 2

La méthode srp() qui retourne le couple (paquets avec réponse / paquets sans réponse) Dans les paquets sans réponse il y a None quand le délai (timeout) est atteint.

L'utilisation de scp() s'impose. Le timeout impose aussi le temps total du scan, plus il sera long plus le scan durera. J'ai choisi un timeout de 4 secondes. On se limitera aux trames reçues srp(...) [0] Dans un premier temps pour voir les objets retournés, on va envoyer une trame uniquement à la passerelle 192.168.100.254 (pour moi, indiquez la votre)

>>> trame = Ether()/ARP(pdst="192.168.100.254")

>>> result = srp(trame, timeout=4,verbose=0)[0]

On ne regarde le premier élément du tableau (qui n'en comporte qu'un)

>>> result[0]

C'est aussi un tableau, on fait afficher le premier élément puis le second >>> result[0][0]

<Ether type=ARP |<ARP pdst=192.168.100.254 |>>

>>> result[0][1]



Le premier élément est la trame d'envoi, le deuxième élément est la réponse que l'on peut afficher plus clairement avec la méthode show()

>>> result[0][1].show()	
###[Ethernet]###	
dst = b8:27:eb:08:df:38	
src = 14:0c:76:a3:51:c1	
type = ARP	
###[ARP]###	
hwtype = 0x1	
ptype = IP_{V4}	
hwlen = 6	
plen = 4	
op = is-at	
hwsrc = 14:0c:76:a3:51:c1	< adresse MAC de réponse
psrc = 192.168.100.254	< adresse IP de réponse
hwdst = b8:27:eb:08:df:38	
pdst = 192.168.100.22	
###[Padding]###	
load = 'x00x00x00x00x00x00	0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x

On regroupe tout cela dans un fichier monscan.py en important seulement ce dont on a besoin et en remplaçant par votre adresse de réseau.

from scapy.all import ARP, Ether, srp

trame = Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(pdst="192.168.100.0/24") resultat = srp(trame, timeout=4,verbose=0)[0]

for envoi, reponse in resultat:
 print("{:16} {}".format(reponse.psrc, reponse.hwsrc))

Pour exécuter le script : sudo python3 monscan.py

— Pour aller plus loin —

Utiliser les objets IP() et ICMP() pour faire un scan de niveau 3 en pinguant votre cible.

Si un firewall protège votre cible quel scan serait possible en niveau 3 qui pourrait le démasquer (ce type de scan existe avec nmap). Après avoir trouvé la réponse vous pouvez la programmer avec scapy. (la réponse sera donnée au prochain TPA)

Petit complément pour clôturer ce TPA —

On a apprendre, depuis un terminal, à charger un fichier à partir d'une adresse web. Depuis un terminal du Raspberry Pi exécuter la commande suivante : wget https://www.idreso.org/TPA/p3 Il faut rendre le fichier exécutable : chmod +x ./p3

▶⊠:

Afin de nous donner un petit signe d'approbation de ce TPA, vous pouvez vous signaler en exécutant ce programme (//p3).

Au plaisir de vous retrouver pour le prochain TPA la semaine prochaine...

