ΚΕΦΑΛΑΙΟ 6

ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ SCRIPT ΓΙΑ ΤΗΝ ΑΝΤΙΜΕΤΩΠΙΣΗ ΤΟΥ SASSER

Εισαγωγή

Από το 1988 και το worm που έφτιαξε ο ερευνητής Robert Morris για να δει αν η εξάπλωση ενός ιου είναι εφικτή, κανείς δεν μπορούσε να φανταστεί το μέγεθος του κινδύνου που θα δημιουργούσαν οι δικτυακοί ιοί και τα worms. Τα τελευταία χρόνια τα διάφορα worms που κυκλοφορούν στο internet -msblaster, slammer, code red, nimda, sasser, mydoom κ.α.- έχουν προκαλέσει τεράστιες ζημιές και προβλήματα. Η ταχύτητα με την οποία είναι σε θέση να μεταδίδονται και να προσβάλλουν καινούργιους χρήστες αυξάνεται εκθετικά, καταλήγοντας σε πλήρη κατάρρευση πολλών δικτύων[1,5]. Οι ερευνητές της ασφάλειας δικτύων συνεχώς καταδεικνύουν τους κινδύνους από την ύπαρξη μηχανημάτων με προβλήματα ασφάλειας και το ενδεχόμενο κάποια στιγμή να εμφανιστεί ένα worm με κακόβουλο payload. Είναι πολύ πιθανό ότι στο μέλλον τα worms θα περιέχουν πιο καταστροφικά payloads και θα κάνουν πράγματα όπως να σβήνουν αρχεία, να εγκαθιστούν sniffers και να κλέβουν εμπιστευτικά αρχεία. Ωστόσο, υπάρχει μια ισορροπία στο να είναι ένα worm απόλυτα καταστροφικό και στο να είναι γρήγορη η μετάδοση του, γιατί όπως και στην φύση, ένας ιός που καταστρέφει τον ξενιστή πολύ γρήγορα δεν καταφέρνει να μεταδοθεί αποτελεσματικά.

Τα worms

Ένα δικτυακό worm είναι στην ουσία κώδικας ο οποίος διαδίδεται μόνος του[2]. Για να εξαπλωθούν τα worms κάνουν scan στο internet σε τυχαίες -συνήθως-1διευθύνσεις ip's και προσπαθούν να εκμεταλλευτούν κάποια αδυναμία στην ασφάλεια που είναι συγκεκριμένη για αυτό το worm, ώστε να μολύνουν τον στόχο. Έπειτα εκτελούν ενέργειες που εξαρτώνται από το payload του worm, κάποια worms για παράδειγμα μετατρέπουν τον μολυσμένο host σε μηχάνημα αποστολής spam. Τέλος ο μολυσμένος πια host γίνεται ένα μέσο για να συνεχίζει να εξαπλώνεται το worm.

Όπως έχει περιγράφει στο μοντέλο του Edward Amoroso[3] ένα worm εκτελεί 3 ενέργειες:

Μόλυνση : εκμεταλλεύεται μια αδυναμία συστήματος και μολύνει έναν στόχο Payload : εκτελεί κακόβουλες ενέργειες στον μολυσμένο στόχο ή σε άλλους απομακρυσμένους hosts. Μετάδοση : χρησιμοποιεί τον μολυσμένο στόχο για να μεταδοθεί και σε άλλους hosts.

Αντιμετώπιση worms

Για να περιοριστεί η εξάπλωση των worms, θεωρητικά αρκούν οι παρακάτω 2 ενέργειες :
1) το λογισμικό που παράγεται να είναι ασφαλές.
2) Οι χρήστες να εγκαθιστούν τα patches και να προσέχουν την ασφάλεια των συστημάτων τους.

Όσον αφορά το λογισμικό, αν και η επιστήμη του προγραμματισμού έχει εξελιχθεί και οι προγραμματιστές γνωρίζουν πολύ περισσότερα απ' ότι στο παρελθόν, ο αριθμός των αδυναμιών που δημοσιεύονται καθημερινά σε sites όπως τα packetstormsecurity.com και securityfocus.com αυξάνεται συνεχώς. Μπορεί οι προγραμματιστές να παράγουν κώδικα που λύνει προβλήματα που σε καμία περίπτωση δεν μπορούσαν να λυθούν στο παρελθόν, η εκπαίδευση τους όσον αφορά την ασφάλεια του κώδικα που γράφουν παραμένει χαμηλή. Αν και έχει γίνει καλή δουλειά από τους ερευνητές στο πώς να γράφεται ασφαλής κώδικας [11] πολλοί προγραμματιστές χρησιμοποιούν συναρτήσεις και τεχνικές οι οποίες προκαλούν παραβίαση του προγράμματος. Για αυτό οφείλεται και το γεγονός ότι πολλά από τα προγράμματα που γράφονται είναι υπερβολικά πολύπλοκα ή έχουν σχεδιαστεί ώστε να επιλύουν ιδιαίτερα δύσκολα προβλήματα και ο κίνδυνος να υπάρχει προγραμματιστικό λάθος αυξάνεται όσο και το μέγεθος του προγράμματος.
Ανεξάρτητα όμως από αυτά, μερικές εταιρίες φαίνεται ότι σχεδόν συνειδητά αποφεύγουν να ασχοληθούν με την ασφάλεια στο λογισμικό τους! Τα ολοένα και αυξανόμενα προβλήματα στο λογισμικό της Microsoft οφείλονται για το συντριπτικά μεγαλύτερο μέρος της κακόβουλης κίνησης στο internet [4] αποδεικνύουν ότι η ασφάλεια στο λογισμικό της εταιρίας είναι χαμηλής προτεραιότητας [13,12] .
Όσον αφορά το δεύτερο, αν και οι περισσότεροι χρήστες του internet χρησιμοποιούν ολοένα και πιο πολύπλοκες και προχωρημένες υπηρεσίες και προγράμματα, δεν προκύπτει από κάπου ότι η συνολική ευαισθητοποίηση για θέματα ασφάλειας αυξάνεται. Οι χρήστες συνηθίζουν να μην δίνουν σημασία στην ασφάλεια του συστήματος τους, καθώς οι εταιρείες λογισμικού και οι περισσότερες ιστοσελίδες τους εθίζουν στην ψευδαίσθηση ότι ασχολούνται εκείνοι με την ασφάλεια, αποφεύγοντας έτσι να τους ενημερώνουν και να τους εκπαιδεύουν.

Tarpit για αντιμετώπιση του code red

Το 2000 κυκλοφόρησε το worm code red. Σε μια προσπάθεια να σταματήσουν την εξάπλωση του, ερευνητές δημιούργησαν το εργαλείο labrea, το οποίο απαντάει σε εισερχόμενες αιτήσεις στην πόρτα 135 με συγκεκριμένα πακέτα, δίχως να ολοκληρώνεται η

σύνδεση, ώστε να κρατάει ανοικτή για πολλή ώρα τη σύνδεση με το μολυσμένο host που προσπαθεί να μεταδώσει το worm [6,7]. Στην ουσία το tcp παράθυρο μένει συνέχεια μηδέν ώστε να μην μπορεί ο source host να στείλει δεδομένα. Με αυτό τον τρόπο μπορεί να καθυστερήσει κάπως η εξάπλωση του worm. Το ίδιο γίνεται και με το tarpit module που διαθέτει το iptables, με την εξής σύνταξη:

iptables -A INPUT -p tcp -m tcp –dport 135 -j TARPIT

Το honeyd διαθέτει επίσης tarpit δυνατότητες, που μπορούν να χρησιμοποιηθούν για να αποθαρρύνουν κάποιον από το να συνεχίσει τις επιθέσεις του στο δίκτυο ή για να καθυστερήσουν τη μετάδοση ενός worm.

Active defense

Μια από τις πιο ενδιαφέρουσες εφαρμογές για το honeyd είναι η ενεργή προστασία ενάντια στους ιούς και τα worms. Ανεξάρτητα από ενέργειες όπως το tarpit για να περιοριστεί ο χρόνος μετάδοσης ενός worm, αξίζει να δουμε την πιθανότητα ένα honeyd host να εκτελεί ένα script το οποίο θα καθαρίζει από το worm, μόλις δέχεται επίθεση απο κάποιο μολυσμένο host. Αυτή η ενέργεια λέγεται active defense ή και strike-back. Αντί απλά να περιμένει την επίθεση από κάποιο worm και να καταγράφει το payload του ή να το καθυστερεί, στην περίπτωση αυτή προσπαθεί να τον καθαρίσει από τον worm!

Η θεωρία έχει ως εξής: ο host A είναι μολυσμένος από κάποιο worm και προσπαθεί να τον μεταδώσει στον host B, ο οποίος είναι κάποιο virtual host που δημιουργήσαμε με τη βοήθεια του honeyd. Εφόσον ο B δεν έχει κάποια πραγματική χρήση, με το που δέχεται την σύνδεση από τον A ξέρουμε ότι πρόκειται για επίθεση. Ο B εκτελεί επίθεση στον A, προσπαθώντας να τον παραβιάσει, εκμεταλλεύοντας το vulnerability με το οποίο είναι μολυσμένος και υποθέτοντας ότι το worm δεν έχει καθαρίσει το vunerability απο το σύστημα. Μερικά worms αφού παραβιάσουν έναν host, στη συνέχεια τον καθαρίζουν από το vulnerability, ώστε να μην μπορεί να παραβιαστεί.

Αν ο B πετύχει να παραβιάσει τον A, σταματάει την διεργασία του worm, σβήνει τον ιο και προσπαθεί να τον προστατεύσει ώστε να μην ξαναπαραβιαστεί.

Στην όλη διαδικασία πάντως μπορεί να υπάρχουν νομικές επιπλοκές, καθώς μπορεί το honeyd host να δέχτηκε επίθεση, αλλά δεν επιτρέπεται να επέμβει στον remote host, ακόμα κι αν αυτός είναι μολυσμένος και του επιτίθεται! Πάντως σε ένα τοπικό δίκτυο το σενάριο αυτό μπορεί να χρησιμοποιηθεί χωρίς κάποιο ιδιαίτερο πρόβλημα.

Honeyd vs msblast

Με το να προσομοιώνουμε hosts και υπηρεσίες θα καταφέρουμε να ξεγελάσουμε έναν μολυσμένο host μόλις επικοινωνήσει με κάποιο από αυτά και θα επιχειρήσει να παραβιάσει το σύστημα μας. Η διαδικασία που ακολουθεί εφαρμόστηκε από τον ερευνητή ασφάλειας Laurent Oudot[10] για την αντιμετώπιση του worm msblast[8] που κυκλοφόρησε τον Αύγουστο του 2003 και εκμεταλλεύεται ένα κενό στην ασφάλεια του microsoft dcom.

Το worm msblast αφού μόλυνε έναν host, στη συνέχεια έκανε dos -denial of service επίθεση στο site του windows update ώστε να μην μπορέσουν οι χρήστες να κάνουν update! Αν και ο ιός καθαρίζεται χειρωνακτικά με πολύ απλά βήματα, για πολύ κόσμο που δουλεύει μόνο με τα patches δημιούργησε σημαντικά προβλήματα.

Αρχικά δημιουργούμε έναν virtual host Microsoft Windows με ανοικτή την πόρτα 135, ώστε να δεχτεί σύνδεση απο μολυσμένους hosts και με την πόρτα 4444 να τρέχει το strike-back script.

```
create default
set default personality "Windows XP Pro"
add default tcp port 135 open
add default tcp port 4444 "/bin/sh scripts/strikeback.sh $ipsrc"
set default tcp action block
set default udp action block
```

Ο msblast βλέπει πρώτα αν η 135 πόρτα είναι ανοικτή. Αν είναι επιχειρεί να στείλει το payload που προκαλεί την παραβίαση. Μετά προσπαθεί να συνδεθεί με την 4444 στην οποία τρέχει το remote shell απο το οποίο εκτελεί τις εντολές του.

Σύμφωνα με το παραπάνω όταν ο host δεχτεί σύνδεση στην 4444 θα εκτελέσει το script strikeback.sh στην εισερχόμενη ip. Το strike back.sh είναι το εξής:

```
!#/bin/sh
# Launches a DCOM exploit toward the infected attacking host
# and then run cleaning commands in the remote DOS shell obtained
./dcom_exploit -d $1 << EOF
REM Executes the following orders on the host :
REM 1) Kill the running process MSBlast.exe
taskkill /f /im msblast.exe /t
REM 2) Eliminate the binary of the worm
del /f %SystemRoot%\system32\msblast.exe
REM 3) Clean the registry
echo Regedit4 > c: \cleanerMSB.reg
```

```
echo [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run]
>> c:\cleanerMSB.reg
echo "auto windows update" = "REM msblast.exe" >> c: \cleanerMSB.reg regedit /s c:
\cleanerMSB.reg del /f c:\cleanerMSB.reg

REM N) Specific actions to update the Windows host could be added here
REM N+1) Reboot the host
shutdown -r -f -t 0 exit
EOF
```

Αρχικά το script τρέχει το exploit που εκμεταλλεύεται το vulnerability στο dcom ενάντια στον μολυσμένο host που μόλις επικοινώνησε. Αν πετύχει, σταματάει τη διεργασία του worm, αναιρεί τον ιο, σβήνει την καταχώριση από τη registry και κάνει reboot. Το script θα μπορούσε να βελτιωθεί με το να περιέχει κώδικα που θα ασφαλίζει το σύστημα ώστε να μην ξαναπαραβιαστεί απο το worm.

Ο msblast καθαρίζεται εύκολα από το σύστημα καθώς αποτελείται από ένα εκτελέσιμο το οποίο μπορεί να σβηστεί, μια διεργασία που το τρέχει και την καταχώριση στη registry. Δυστυχώς τα περισσότερα worms καθαρίζονται με πολύ πιο πολύπλοκους τρόπους, καθώς δημιουργούν πολλαπλές διεργασίες, με πολλά ονόματα, ή αποτελούνται από περισσότερα εκτελέσιμα.

Στο δικό μας πείραμα δεν δεχτήκαμε καμιά επίθεση από host μολυσμένο από τον msblast, για να μπορέσουμε να επαληθεύσουμε το σενάριο αυτό. Ίσως για το ότι ο msblast δεν κυκλοφορεί πια να οφείλεται το γεγονός ότι ο ιός προκαλεί reboot μετά από 60 δευτερόλεπτα, οπότε σε καμία περίπτωση δεν περνάει απαρατήρητος, όπως άλλοι ιοί και ο χρήστης πρέπει να τον αφαιρέσει αν θέλει να συνεχίσει να δουλεύει.

Το worm sasser

Το worm sasser εμφανίστηκε στις 30 Απριλίου 2004 και εκμεταλλεύεται το MS04011 LSASS exploit [9]. Μόλις εκτελεστεί αντιγράφεται στο %WINDIR% σαν avserve.exe , προσθέτει μια καταχώρηση στη Registry, ανοίγει έναν μίνι ftp-server στην 5554 για να μεταφέρει το εκτελέσιμο του ιου και σε άλλα συστήματα, ξεκινάει 128 threads τα οποία ψάχνουν να εκμεταλλευτούν άλλα συστήματα και τέλος καλεί την API method AbortSystemShutdown ώστε να μην μπορεί το σύστημα να κάνει reboot.

Η διεργασία που κάνει το scanning για άλλα συστήματα λειτουργεί ως εξής: Αρχικά προσπαθεί να δει την τοπική διεύθυνση ip. Αν βρει μια δημόσια δρομολογήσημη διεύθυνση θα τη χρησιμοποιήσει, αλλιώς χρησιμοποιεί ιδιωτικές διευθύνσεις, σύμφωνα με το πρότυπο RFC1918. Το 50% του χρόνου προσπαθεί να παραβιάσει εντελώς τυχαίες ip διεύθυνσης και

το άλλο 50% τυχαίες ip διευθύνσεις από το τοπικό δίκτυο. Όταν τα καταφέρνει, ανοίγει ένα shell στο απομακρυσμένο σύστημα στην πόρτα 9996. Έπειτα συνδέεται στην πόρτα αυτή και κατεβάζει το εκτελέσιμο του worm, το οποίο και τρέχει και έπειτα συνεχίζει μετά από 250miliseconds, επαναλαμβάνοντας την ίδια διαδικασία.

Το vulnerability που εκμεταλλεύεται ο sasser υπάρχει στα περισσότερα από τα πρόσφατα λειτουργικά της microsoft και σε αυτό οφείλεται το γεγονός ότι το worm απλώθηκε πολύ γρήγορα και σε πολλά συστήματα.

Ο συγγραφέας του sasser φαίνεται πως έχει γράψει και το worm netsky. Αμέσως μετά την κυκλοφορία του sasser, βγήκαν αρκετές διαφορετικές εκδοχές του worm . Μια από αυτές αυξάνει τις διεργασίες από 128 σε 1024. Κάποια άλλη επιταχύνει το χρόνο μετάδοσης του worm.

Honeyd vs sasser

Για την αντιμετώπιση του worm sasser δημιουργήσαμε το script lsass4.sh που θα τρέχει όταν κάποιο απο τα virtual hosts δέχεται σύνδεση στην πόρτα 445. Το script θα προσπαθεί να καθαρίσει το worm sasser και τις διάφορες παραλλαγές του, πχ sasser b, c, d, e, f, g. Οι παραλλαγές του sasser έχουν διαφορετικά ονόματα στο εκτελέσιμο που αντιγράφουν, στην καταχώριση στη registry και στη διεργασία του ιου. Ο sasser ονομάζει τη διεργασία και το εκτελέσιμο του avserve.exe. Ο sasser.b avserve2.exe, οι άλλες παραλλαγές χρησιμοποιούν τα ονόματα avserve3.exe, skynetave.exe, napatch.exe, package.exe, lsasss.exe, προσπαθώντας να μπερδέψουν το χρήστη που ψάχνει για μια συγκεκριμένη διεργασία να σταματήσει. Όταν μεταδίδεται ο sasser σε κάποιο υπολογιστή, προκαλεί κατάρρευση στην υπηρεσία lsass.exe και ο υπολογιστής κάνει reboot. Φυσικά την επόμενη φορά που ξεκινάει ο υπολογιστής, ο sasser θα τρέξει αυτόματα, καθώς το εκτελέσιμο του υπάρχει στον υπολογιστή και επίσης η καταχώρηση στη registry. Από τις παραλλαγές του sasser μόνο ο sasser.g δεν προκαλεί κατάρρευση στο σύστημα. Πριν
προλάβει ο υπολογιστής να καταρρεύσει, ο ιός δημιουργεί πολλές εκατοντάδες scans ταυτόχρονα προσπαθώντας να εντοπίσει υπολογιστές για να τους μολύνει. Τις συνδέσεις αυτές μπορούμε να δούμε σε ένα windows σύστημα εκτελώντας την εντολή

netstat -an

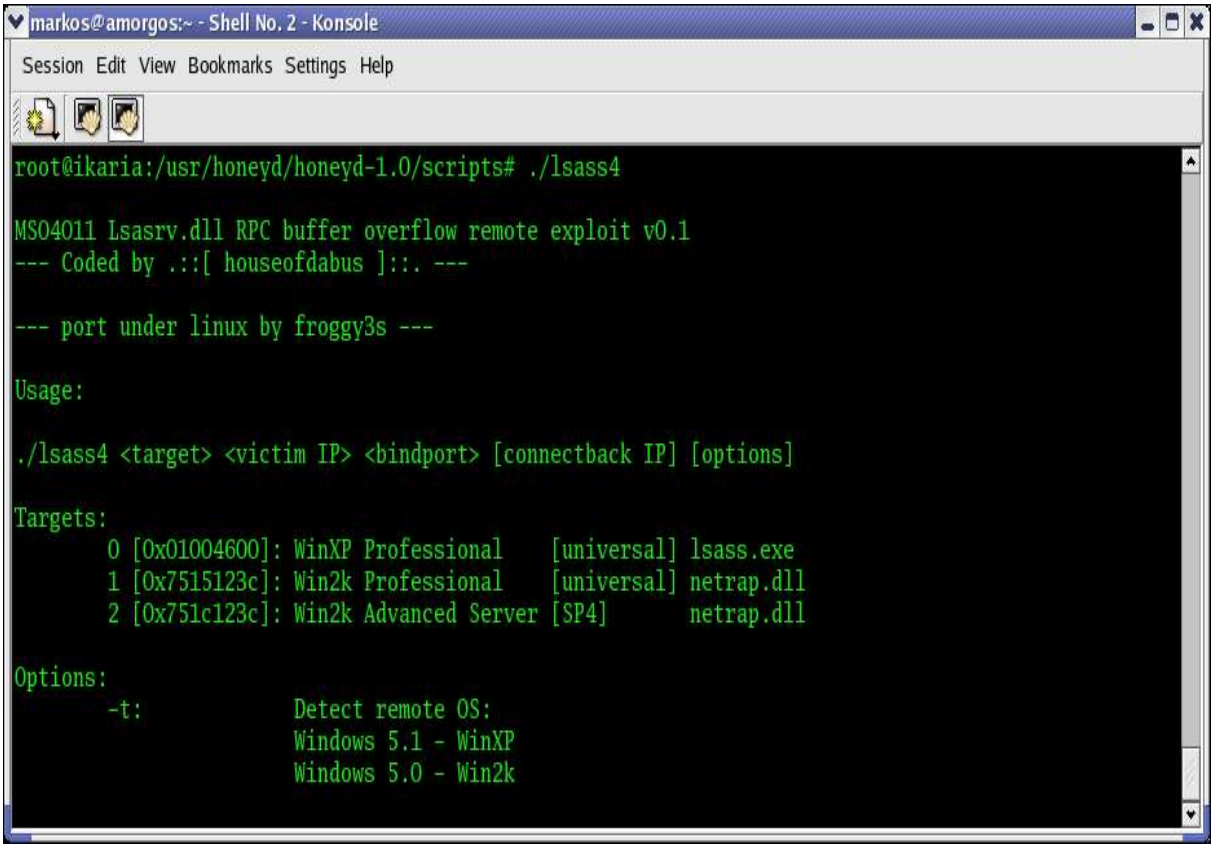
Επίσης αν κοιτάξουμε τις διεργασίες στο σύστημα (ctrl+alt+del) θα δούμε ότι υπάρχει η διεργασία avserve.exe, η οποία μεγαλώνει, καθώς επίσης και η διεργασία του συστήματος

lsass.exe.

1 Για το πείραμα μας δημιουργήσαμε ένα εικονικό δίκτυο σε ένα vmware workstation, που έτρεχε την έκδοση 4 του vmware. Ο ένας υπολογιστής έτρεχε slackware linux 9 (ο honeyd host) και είχε την ip 1.1.1.1, ενώ οι virtual hosts που δημιουργούσε είχαν ip's 1.1.1.2 μέχρι 1.1.1.250. Ο windows υπολογιστής έτρεχε το λειτουργικό σύστημα windows xp professional χωρίς patches και κάποιο firewall, οπότε ήταν ευάλωτος στην αδυναμία lsass. Διαθέτοντας ένα εκτελέσιμο του sasser, τον sasser.b, το εκτελέσαμε ώστε να μολύνει το windows σύστημα και ο στόχος μας ήταν να καθαριστεί το σύστημα αυτόματα μόλις θα προσπαθήσει να μεταδώσει τον ιο σε κάποιο από τα virtual hosts μας, το οποίο θα εκτελέσει το strike back script μας.

Το script αρχικά εκτελεί το lsass4 ενάντια στην εισερχόμενη ip από την οποία δέχτηκε τη σύνδεση στην πόρτα 445. Το lsass4 είναι το exploit που προσπαθεί να εκμεταλλευτεί την αδυναμία στο MS04022 Lsarv.dll και του προκαλεί buffer overflow. Το exploit το κατεβάσαμε από το site packetstormsecurity.com. Το vulnerability για το lsarv.dll εκμεταλλεύεται και το worm sasser. Αφού δεχόμαστε επίθεση από host που προσπαθεί να μας μολύνει με τον sasser, υποθέτουμε ότι και ο ίδιος ο host είναι μολυσμένος από το worm. Του κάνουμε επίθεση χρησιμοποιώντας το exploit αυτό και αφού παραβιάσουμε το σύστημα του προσπαθούμε να τον καθαρίσουμε. Μετά την επιτυχημένη παραβίαση ενός συστήματος με το exploit αυτό, τα προνόμια που δίνει είναι system priviledges. Εφόσον στο σύστημα δεν υπάρχει χρήστης system, τα προνόμια του είναι κάπως περιορισμένα, ενώ δεν έχει καν path και άλλες environment variables. Το βασικότερο είναι οτι με τα προνόμια αυτά δεν μπορούμε να σταματήσουμε τη διεργασία του sasser που τρέχει και σαν συνέπεια δεν μπορούμε να σβήσουμε το εκτελέσιμο και την καταχώριση στη registry, δεν μπορούμε δηλαδή να καθαρίσουμε το σύστημα απο τον sasser. Για το λόγο αυτό ακολουθήσαμε άλλο τρόπο, αφού πάρουμε πρόσβαση στο σύστημα να δημιουργήσουμε ένα αρχείο στο startup κατάλογο των windows που θα περιέχει τις εξής εντολές:

1. Θα τερματίζει τη διεργασία του worm.
2. Θα σβήνει το εκτελέσιμο του worm.
3. Θα καθαρίζει την καταχώρηση στη registry
4. Θα προσπαθεί να προστατέψει τον υπολογιστή απο μελλοντικές εμφανίσεις του sasser.



εικόνα 6.1-το exploit για το lsarv.dll

προσθήκη στο honeyd.conf :

```
add default tcp port 445 "/bin/sh scripts/lsass4.sh $ipsrc"
```

Το script για λόγους ευκολίας της παρουσίασης είναι μόνο για την αντιμετώπιση του sasser.b. Με μικρές προσθήκες μπορεί να αντιμετωπίσει και τις άλλες παραλλαγές του sasser:

echo taskkill /f /im avserve2.exe /t >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat

echo attrib -R %SystemRoot%\debug\dcpromo.log >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat
echo del /f %SystemRoot%\debug\dcpromo.log >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat
echo echo dcpromo > %SystemRoot%\debug\dcpromo.log >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat
echo attrib +R %SystemRoot%\debug\dcpromo.log >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat

echo reg delete HKLM\SOFTWARE|Microsoft\Windows|CurrentVersion\Run /r avserve2.exe >>c:\windows and settings\all users\start menu\programs\startup\sasser -

remove.bat
echo del c:\windows\avserve2.exe >>c:\windows and settings\all users\start menu\programs\startup\sasser-remove.bat

Το script αναλυτικά:

- Σταματάει τη διεργασία του worm. Η διεργασία του sasser είναι το avserve.exe, αλλά οι διάφορες εκδοχές του sasser έχουν τα ονόματα package.exe, lsass.exe κτλ.
- Δημιουργεί το αρχείο dcpromo.log στον κατάλογο %SystemRoot%\debug\ το οποίο προστατεύει το σύστημα ώστε να μην μπορεί να ξαναμολυνθεί από τον sasser. Μια γρήγορη και αποδοτική λύση για το patch του συστήματος, που βρήκαμε στο σχετικό post για το vulnerability στο securityfocus.com
- Τρέχει εντολές για την registry ώστε να απομακρυνθούν οι καταχωρίσεις από το worm.
- Σβήνει το εκτελέσιμο του worm.

Το script αυτό θα εκτελείται από οποιονδήποτε χρήστη των windows κάνει login, καθώς βρίσκεται στον κατάλογο c:\windows and settings\all users\start menu\programs\startup\

Προβλήματα – μελλοντική δουλειά
Το script δοκιμάσαμε να το περάσουμε με χειρωνακτικό τρόπο στο windows σύστημα, αφού τρέξαμε το exploit για το lsass και δούλεψε με επιτυχία. Το μόνο πρόβλημα που μένει να λύσουμε είναι το πως θα διοχετεύσουμε το script στο exploit, ώστε να γίνεται η δουλειά αυτόματα. Η δυσκολία που αντιμετωπίζουμε προς το παρόν είναι το πως θα περάσουμε τις εντολές με telnet στην πόρτα που ανοίξαμε (4445). Είναι σίγουρο ότι με τη μελλοντική δουλειά θα προχωρήσουμε από το μικρό αυτό σημείο στο οποίο “κολλήσαμε”.

Σημείωση ομάδας ISLab: *Σκοπός αυτού του κεφαλαίου ήταν η αποτελεσματική αντιμετώπιση του worm Sasser με το εργαλείο Honeyd. Το παραπάνω script όταν εκτελείται πάνω σε σύστημα μολυσμένο από worm Sasser τερματίζει και καθαρίζει το worm, αλλά δεν μπορεί να εκτελεστεί από το Honeyd για την αυτόματη αφαίρεση του Sasser.*

Η δυνατότητα προγραμματισμού του Honeyd για την αυτόματη αφαίρεση του worm είναι αντικείμενο περαιτέρω πειραματισμού στο εργαστήριο ISLab.

Συμπεράσματα

Αν και σε καμία περίπτωση το honeyd δεν αποτελεί την κύρια προστασία στους ιους και τα worms, παρόλαυτα αποτελεί ένα επιπλέον εργαλείο για να καταλάβουμε τη λειτουργία τους και να περιορίσουμε τη δράση τους. Οι δυνατότητες για strike-back διαφαίνονται ιδιαίτερα ελπιδοφόρες για το μέλλον. Επιπλέον, η χρήση του honeyd μπορεί να διευκολύνει έναν διαχειριστή δικτύου να εντοπίζει γρήγορα τους μολυσμένους υπολογιστές εντός του δικτύου του ώστε να τους καθαρίσει.

ΣΗΜΕΙΩΣΕΙΣ - ΠΑΡΑΠΟΜΠΕΣ

[1] www.securityfocus.com/news/6767

[2] Ryan Permeh et Dale Caddington (Eeye), " Decoding and understanding Internet Worms", 21st November 2001, http://www.blackhat.com/presentations/bh-europe01/dale-coddington/bh-europe-01-coddington.ppt

[3] Edward Amoroso, chapter 4.5 from " Fundamentals of computer security technology ", explaining replication of the worms called "Internet viruses" ; see " Typical virus operation "

[4] Infected windows pcs now source of 80% of spam , http://www.sandvine.com/solutions/pdfs/spam_trojan_trend_analysis.pdf

[5] Stuart Staniford, Vern Paxson, and Nicholas Weaver. How to 0wn the Internet in your Spare Time. In Proceedings of the 11th USENIX Secuirty Symposium, August 2002.

[6] Tony Bautts slowing down Internet worms with tarpits www.securityfocus.com/infocus/1723

[7]Tom Liston, " Welcome to my tarpit, the tactical and strategic use of Labrea " http://labrea.sourceforge.net/labrea-info.html

[8] Security advisory for the MSBlast worm that appeared on the 11th august 2003 and that abused a vulnerability announced in July 2003. http://www.microsoft.com/security/incident/blast.asp

[9] Sasser worm analysis by LURHQ Threat Intelligence Group www.lurhq.com/sasser.html

[10] Laurent Oudot, RSTack Team, "Fighting Internet Worms With Honeypots", 23 October 2003, http://www.securityfocus.com/infocus/1740.

[11] Dwheeler, secure programming for linux and unix how to, www.dwheeler.com

[12] Microsoft fails to patch all flaws, www.itnews.com.au/newsstory.aspx?CIaNID=17533&eid=3&edate=20050112

[13] Microsoft scrambling to fix new outlook security hole,
http://archives.cnn.com/2000/TECH/computing/07/21/ms.outlook.bugs.idg/