

The 2006 International Capture The Flag Hacking Competition

Fall 2006

Giovanni Vigna

University of California Santa Barbara

<http://www.cs.ucsb.edu/~vigna>

The UCSB iCTF

- A “Capture The Flag” hacking contest
- Teams from a number of Universities/Institutions spread across the world compete against each other
- Each team has to defend a virtual network it manages and attack the networks managed by other teams
- Teams are connected by a dedicated overlay network
- A real-time scoring system determines who is the best at defense and attack

A Little Bit of History

- Winter 2001: Red Team/Blue Team
 - Two teams, no virtualization
 - Red Team: has to obtain a “flag” file hidden on specific hosts
 - Blue Team: has to protect the flag and detect attacks
- Spring 2002: Capture The Flag
 - Two teams, no virtualization
 - Defend flag and capture other team’s flag
- Fall 2002: Treasure Hunt
 - Two teams, no virtualization
 - Timed progress through scenario in parallel

A Little Bit of History

- Fall 2003: Capture The Flag
 - Thirteen teams from the US
 - Vulnerable host is VMware image
 - Traffic is anonymized
- Fall 2004: International Capture The Flag
 - Nineteen teams from the US and Europe
 - Virtualization, anonymization
 - Glitch in the virtual network spoils some of the fun
- Spring 2005: International Capture The Flag: Rematch!
 - Twelve teams from US and Europe
 - “Blender” technology allows for non-anonymized traffic

A Little Bit of History

- July 2005: ShellPhish team from UCSB wins the DEFCON CTF exercise
 - Eight teams (mostly US?), “jailed” systems
 - Anonymization
- Fall 2005: UCSB’s Intercontinental Capture The Flag!
 - Twenty-two teams from Europe, Australia, North America, South America
 - Largest capture the flag EVER!

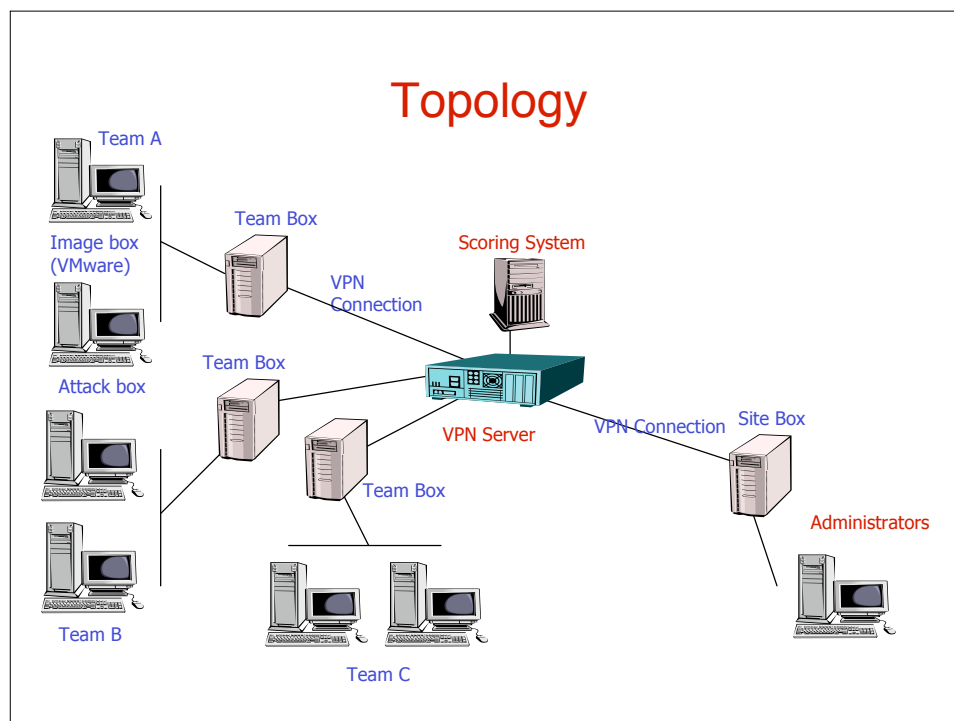


DEFCON’s CTF

- DEFCON (<http://www.defcon.org>) is the largest underground hacker convention in the world
- The associated CTF competition is regarded as the "world championship of hacking"
- First well-organized CTF by the Ghetto Hackers
 - Introduced virtualization and anonymization
 - Introduced scoring bot
 - In 2004, UCSB’s “Enemy Combatants” placed second
- Picked up by Kenshoto and substantially improved
 - “jailed” system with accounts
 - “Service level” weights score
 - “Breakthrough” concept supports creativity

UCSB's CTF vs. DEFCON's

- UCSB's CTF is played by remote teams (DEFCON's teams are physically co-located)
- UCSB's is played by more teams (DEFCON's CTF typically has eight teams)
- UCSB's CTF is an educational exercise
 - “Come and play” approach
 - No “your problem” attitude
 - Actually, no attitude at all
- UCSB's infrastructure does not anonymize traffic
 - More realistic experience
- UCSB's CTF lasts only a few hours (DEFCON's lasts several days... and nights)



The Vulnerable Network

- A virtual network composed of one or more virtual hosts
- OS images configured with a number of services running on VMWare
 - Supported config: vulnerable guest images on Fedora Core hosts
- The OS images are the only hosts (IP addresses) that must be remotely accessible
 - Exception: test image during debugging
- Service examples
 - World Wide Web, FTP, Telnet, SSH, Finger, ...
- Services have a number of exploitable vulnerabilities
- OS images are distributed at the beginning of the day
- Source code for some services is distributed during the actual contest

Flags

- Each service has one or more flags associated with it
- The flags are set by the scorebot as part of the “normal access” to the system
- The flags are changed in a certain time period
- The flags are accessible (in theory) only to a correctly authenticated user (with credentials that are different for each team and that change through time!)
- To capture another team’s flag
 - Access the flag of a service in another team’s server
 - Submit the flag to the scoring system

Monitoring/Scoring

- Each service is equipped with
 - A “get flag” method
 - A “set flag” method
- Getting and setting the flags do not involve exploiting a vulnerability
- Each team is provided with a script to submit a flag
- A service can be in different states
 - Dead
 - Running
 - Functional (running and flags can be retrieved and set)

Monitoring/Scoring

- The scoring systems attempts to read the flag
 - If no connection can be established, the service is considered down
 - No points are assigned
 - If the flags are not accessible, the service is considered non-functional
 - No points are assigned
- The scoring systems attempts to write the flag
 - If no connection can be established, the service is considered down
 - No points are assigned
 - If the flags cannot be written, the service is considered non-functional
 - No points are assigned

Analyzing The Flag

- If the flag is different from the last flag value set by the scoring system, something wrong happened
 - The flag is ignored
 - A new flag is set
 - The team receives no points
- If the flag is identical to the last flag value set
 - If no other team have submitted the flag value to the scoring system, the team is assigned a number of defense points
 - If one or more other teams have submitted that flag, then the other teams receive a number of attack points

Example

- Server implements a web-based discussion service where authenticated users can post messages
- Users can post messages publicly or to a private discussion forum that requires a username and a password to access the forum contents
- The scoring system connects to the service of team B and creates a new forum, called "recipes" protected by a password, say "spaghetti"
- Then it generates the flag and creates a message in the forum that contains the flag (e.g., in the body of a posting to that forum)
- A sample flag: ECA+hWjCb8t8/FgbEg/mSm1hbU231kjg==

Example

- After some time, the scoring system attempts to log into the forum (using forum name “recipes” and password “spaghetti”) and checks if the flag is still there
- If it is, then a new flag is created as the body of another posting
 - Note that the scoring system might also decide to create a complete different forum to store the new flag
- In order to steal the flag, team A has to connect to team B's server and in some way access the contents of the “recipes” forum

Example

- Of course team A does not know the password used to create the forum and therefore it has to find some way to bypass security
- If successful team A will be able to read the flag
- Then, team A will immediately submit the flag using a form on the scoring web site
- The fact that team A stole the flag will be recorded and, at the end of the scoring period, some points will be granted to team A

Scoring Panel

- The scoring panel provides a snapshot of the status of the CTF
- It is accessible through a web page (refreshed every 30 seconds)
- It provides information of team's ping connectivity (ability to answer to ping probes, basic test connections)
- It provides information about the status of services
 - Down
 - Running but non-functional
 - Functional
 - Note: It does not provide information about the ownership of a service

Scoring Panel

- It provides information about the performance of a team in the last scoring period (say, last 10 minutes)
- Note: It does not provide absolute score values
- Penalties are assigned because of improper behavior (e.g., DOS attacks)
- Penalties are assigned because of traffic usage
- The final winner is declared only at the end of the exercise

The Blender

- Problem: how to prevent teams from filtering out the other teams and let the scorebot through
- Anonymization: By using NAT-ing all traffic appears to come from the same IP
 - Problems with fingerprinting (e.g., TTL), traffic not realistic
- The blender records statistics about the traffic exchanged between teams
 - When a scoring round is to be initiated against team A, the blender chooses what team to impersonate based on the traffic received by A so far
 - If Team A received 35% of traffic from team B, the blender will choose to impersonate B with a 0.35 probability

The Blender

- Advantages
 - No static filtering is possible
 - More realistic experience: you know who is attacking you!
- Disadvantages
 - Complexity

Attack Techniques

- Buffer overflows
- Format string attacks
- Shell attacks
- Race conditions
- Misconfigurations
- Authentication attacks
- Web-based attacks
 - Directory traversal
 - Cookie-based services
 - Cross-site scripting
 - Server-side applications
 - Lack of parameter validation (e.g., SQL injection)

Skills

- Scanning
- Firewalling
- Intrusion Detection
- Vulnerability analysis
- For each type of vulnerability
 - How to identify a vulnerability
 - How to exploit a vulnerability
 - How to patch a vulnerability (without disrupting the get/set flag methods)
 - How to detect a vulnerability
- For each service
 - How to monitor the requests to a service
 - How to monitor the execution of a request
 - Protocol security analysis
 - Application security analysis

Lessons Learned and Suggestions

- Have a structured team with clear responsibilities
 - The Perl/Python/PHP group
 - The SQL/database group
 - The flaw-finder group
 - The firewall group
 - The IDS group
 - The C-based exploit group
- Have a leader responsible for coordination and integration
- Have a way to intercept socket connections and apply regexes/substitutions
- Have vulnerability analysis tools handy
- Have a “human IDS”
- Remember: the game lasts only a few hours