Phillips Academy
Capture the Flag

Examples of Problems given to PACTF Competitors

Basic Competition Format. Teams of up to five participants face a series of increasingly challenging problems. Problem material includes websites, executable files, and encrypted pieces of text created by us. We intentionally leave certain security vulnerabilities in the problem material for teams to discover and exploit. Once they do so, they find hidden answer strings known as *flags*, which they can submit to the PACTF website for feedback and points.

Range of Problems. Problems range in topic from cryptographic exploits to binary file disassembly to internet forensics and network cracking. To engage both new teams and experienced ones, we distribute problem difficulty across a smooth learning curve.

Example of a Beginner Cryptography Problem. We kept the first problem of PACTF 2016's cryptography round very easy to make it accessible to completely new teams. Worth ten points, the problem involved a very simple *rotation cipher* (ROT13). Nearly all teams solved this problem.

Examples of a Medium Binary Problem. The 'binary' round included both binary exploitation and reverse engineering. Reverse engineering problems gave participants executables (compiled binary files) from which they needed to extract a flag using dynamic and static binary analysis. They might have to disassemble the source and parse the machine code or debug the executable and manipulate memory contents.

For example, to solve the problem *Sorcery*, participants had to follow pointers in the executable file, eventually leading them to a series of ASCII characters in memory which, when properly reassembled, yielded the flag.

For another problem, participants are given a binary file (available as the file a.out here: goo.gl/0hL10D) and asked what the script would output if it were to finish running. Though executing the binary would eventually work, it would not finish in a reasonable amount of time. To solve the problem, participants must use a disassembler to determine the important function, read the assembly code, and examine registers at different points in time while executing the file. The function inefficiently determines the maximum prime that can fit in the integer size.

Example of a Difficult Tie-Breaker Problem. The following is an example of a tie-breaker problem: *Create the most number of individual sectors in a circle given 70 division lines. Submit your data in a list of pairs of non-repeating, integer angles. For instance, (0, 180), (90, 235), (45, 315) would be similar to [the image available as sectors.png at goo.gl/0hL10D]. This sample would count as 6 sectors.*

If multiple teams solve all the problems, whichever team scores the most number of sectors would win. If they tie in the number of sectors, whichever team submitted their highest-scoring solution first would win.