A Teacher's Guide to Computer Security Education through Capture the Flag (CTF) Competitions

Adam Michlin
Computer Science
Pope John XXIII Regional High School
Sparta, NJ
amichlin@gmail.com

About Adam Michlin

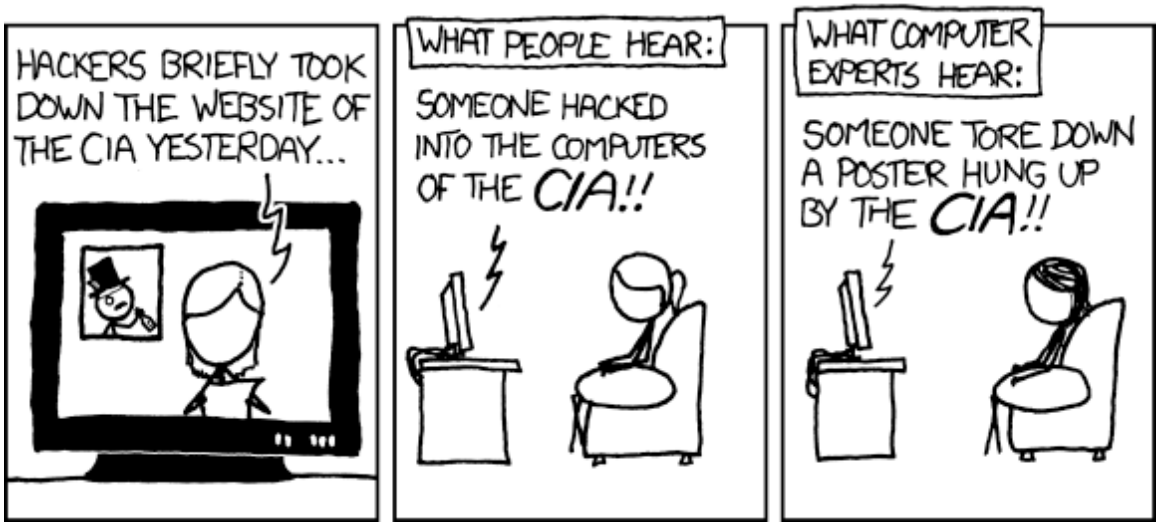
Adam Michlin began his career receiving a BS in Computer Science from the University of California, Santa Cruz in 1997 where he was a teacher assistant in courses including Data Structures, Abstract Data Types and Introduction to Compiler Design I and II.

Subsequent to his degree, he worked for many years as a musician and music educator only to return to computer science education in 2007 at Barron Collier High School in Collier County, Florida. While at Barron Collier, he grew the program from 24 to over 150 students, implementing curriculum which he was asked to reproduce by training teachers throughout Collier County high schools.

In 2014, he made the transition to full time computer science education instruction by taking a position at Pope John XXIII Regional High School in Sparta, NJ where he currently teaches Intermediate Programming, Advanced Computer Security and Web Programming, Advanced Video Game and Mobile Programming and AP Computer Science. In one year at Pope John, he doubled the enrollment in advanced computer science classes and raised the enrollment in the introductory computer science class to over 100 while simultaneously making improvements in the ratio of female to male students through initiatives including a newly founded Girls Who Code club. Pope John XXIII Regional High School now boasts a full five year curriculum for computer science from eighth to twelfth grade.

About the cartoons used in this guide

All cartoons in this guide come from Randall Munroe's webcomic XKCD and are used with permission. Readers are highly encouraged to visit http://www.xkcd.com and support Randall's most wonderful work.



https://xkcd.com/932/

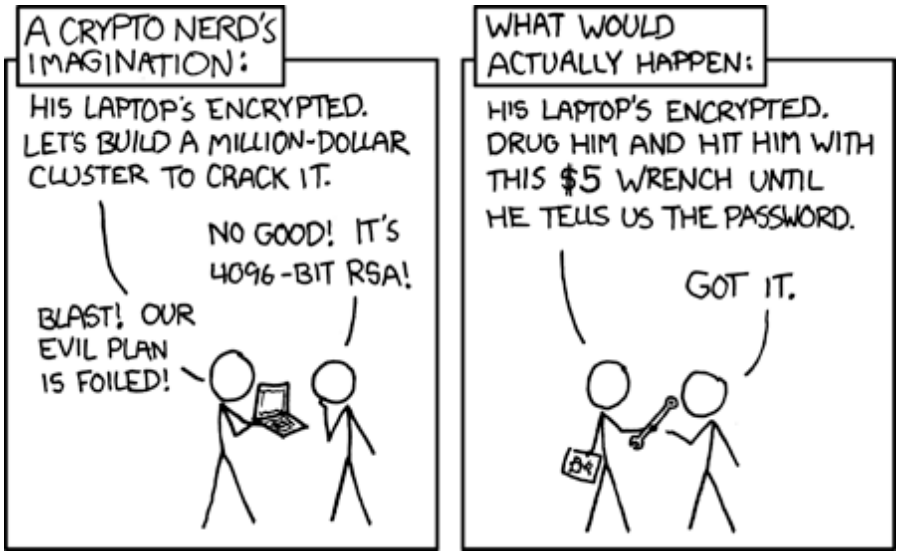
Background

"The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards - and even then I have my doubts"

Eugene Spafford, Computer Science Professor, Purdue, 1989

Pick up any newspaper, read the news on your computer through a web browser, and you cannot avoid the issue of computer security. Personal information on 21 million federal government employees stolen by international hackers, Sony being "hacked" by some mysterious, possibly government sponsored, entity, Microsoft's Xbox gaming servers brought to their knees by a "distributed denial of service attack", any of amount of large companies databases being broken into, computers infected with various malware, credit card numbers shared as quickly as data can travel over the internet, people's private lives opened to public scrutiny ("doxing"), the ubiquitous nature of computers has put computer security to the forefront of the world's consciousness.

As a teacher of computer security, the question I am mostly frequently asked is whether I am teaching students to break laws and wreak havoc. My answer to that is simple: knowledge is neither bad nor good and such bad or good is dictated by how the knowledge is applied. A police officer can well advise the lay public how a would-be burglar would break into a house. Does this make the police offer a bad person? Yet how can the officer educate the public without the knowledge to break in? How is the home owner supposed to protect their property without the knowledge? Breaking into computers is a similar issue since the only way to prevent damage by computer security issues is to understand how those very same computer security issues create vulnerabilities.



https://xkcd.com/538/

Hacking/Computer Security

"I think there is a world market for maybe five computers."
Thomas Watson, President of IBM, 1943

Mirriam's Webster defines the term hacker as follows and this, for better or for worse, has become the popular usage of the word hacker.:

computers : a person who secretly gets access to a computer system in order to get information, cause damage, etc. : a person who hacks into a computer system

In the 1950s, when computers numbered at less than one hundred, the idea of breaking into a system was far out of the minds of computer operators of the day. Computers filled up house sized rooms, were highly protected by their physical surroundings and the idea of accessing a computer remotely (or even having more than one person use a computer) was an impossible absurdity. This was world where the term hacker originated, essentially meaning someone who could make a computer solve a new problem or solve an existing problem in a better (faster, more accurate) way. Hackers prided themselves on repurposing equipment meant for pure calculation to do things like playing the prototypical video games and creating computer music.

"Hacking just means building something quickly or testing the boundaries of what can be done"
Mark Zuckerberg, CEO Facebook

Gradually, computers became smaller, faster, and more generally available. During the 1960s, in response to fears of communication failures in the event of a nuclear attack, the United States Government, through their Advanced Research Projects Agency (ARPA), decided to connect many of these computers at university, government facilities and companies through a network originally

known as the Advanced Research Projects Agency Network (ARPANET) which be the foundation of what known know as the Internet.

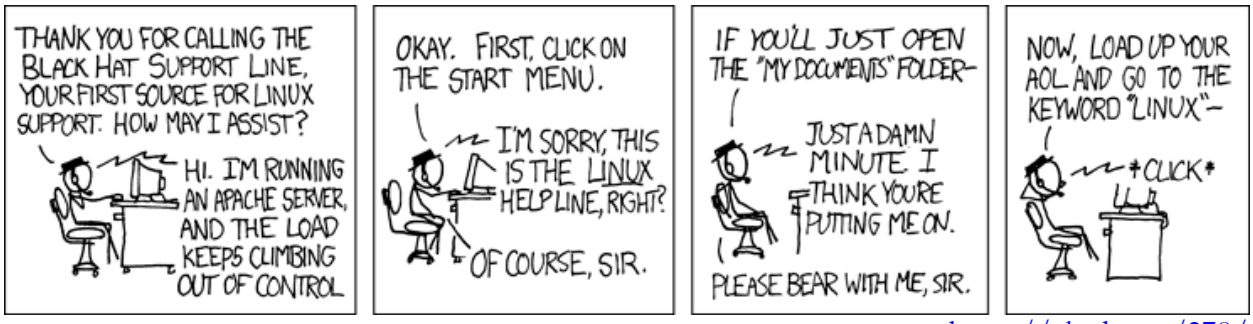
"There is no reason anyone would want a computer in their home."
Ken Olsen, founder of Digital Equipment Corporation, 1977

As computers flourished in the 1970s and 1980s, more and more people had computers at work and eventually on their desktops at home. Computer security became more and more an issue, but was still limited to something corporations and governments worried about. Your average person did not worry about being "hacked" or otherwise having their lives affected by what we now refer to as Computer Security.

This all changed on November 2, 1988 when a student at Cornell University named Robert Tappan Morris (son of Robert Morris, a pioneering computer researcher himself) decided to try to gauge the size of the Internet. He wrote a small computer program that was directed to copy itself from computer to computer using known vulnerabilities in the computers of the day (primarily DEC VAX and Sun-3 systems). Morris' computer spread across the Internet as intended, but as a result of some tweaks to make it more persistent, the program reproduced itself so many times that infected systems were crashing.

In many ways, this was the first episode of what we now call Denial of Service (DOS) attacks, that is, the overwhelming the resources of a computer through an excessive amount of requests for a legitimate service or resource. Consider the real world analogy of sending one thousand customers to the local fast food restraint to purchase a hamburger at exactly 11am on the same date. There is nothing illegal in regards to the purchase of a hamburger, but one thousands customers doing so simultaneously will still overwhelm the resources (parking lot, restaurant space, ability to serve customers) of the restaurant.

Eventually, particularly in the 1990s as home computers became more and more common and home users were increasingly connected to the modern successor of the ARPANET, the Internet, computer security became an issue that affected a good majority of people's everyday lives. With the rise in the 2000s of cell phones and other devices connected to the Internet, computer security became a subject aware to a good portion of the popular culture.



https://xkcd.com/278/

Black, White, and Grey Hats

The terms black, white and grey hats are often used to classify various types of hackers. Taken from the clichés of American Western movies back in the day, here is a breakdown of what each term means:

Black Hat

A black hat hacker is the stereotype most often seen in modern day media. A black hat hacker does what he/she does for the purposes of creating damage, confusion, profiting, or bragging rights.

White Hat

The other side of coin gives us white hat hackers whose reason for being is to protect against black hat hackers.

Grey Hat

As with most things, hacking rarely falls directly into category of black or white so we find most common hacker to find is the grey hat hacker. Think of grey hat hackers are some shade between pure white and pure black, either directly in the middle or leading one way or the other.

A practical scenario

Our hacker finds an undocumented vulnerability in the website owned by AcmeLabs.com. A black hat hacker would tend to either exploit this vulnerability by accessing the AcmeLabs' customer data and exploiting it directly or selling the information of the vulnerability to a third party with an equally nefarious plans. A white hack hacker would immediately inform AcmeLabs of their vulnerability and proceed no further. A grey hat hacker would inform AcmeLabs of their vulnerability but, in the case of AcmeLabs failing to close that vulnerability, they might make the vulnerability public to force AcmeLabs to take action.

What is a Capture the Flag (CTF) Competition?

Traditionally, a CTF competition revolves around two teams defend and invading territory. Each team has a flag (which could be a literal flag or any other object) and the goal is to steal the opposing teams flag and return it your team's base. Doing such is referred to as "capturing the flag".

In computer security, CTF competitions are more about hiding pieces of data in a variety of ways that require understanding of computer and computer security fundamentals. Individuals or groups then compete to find these hidden data items using their knowledge of such fundamentals.

History of CTF

"Capture the Flag is one of the oldest contests at Defcon dating back to Defcon 4. In the past few years, "capturing the flag" has become a popular moniker for all kinds of contests, and the sheer quantity of CTFs has been increasing steadily. Defcon CTF is one of the (if not the) oldest CTF that continues to run today. Here you can find a brief history of the contest and its evolution."

https://www.defcon.org/html/links/dc-ctf-history.html

What is picoCTF?

"picoCTF is a computer security game targeted at middle and high school students. The game consists of a series of challenges centered around a unique storyline where participants must reverse engineer, break, hack, decrypt, or do whatever it takes to solve the challenge. The challenges are all set up with the intent of being hacked, making it an excellent, legal way to get hands-on experience.

https://picoctf.com/about

Why picoCTF?

picoCTF was selected for this guide for very specific reasons. The competition starts with a very slow learning curve that can make the most inexperience high school, even middle school, student feel comfortable. At the same time, the later problems are as difficult as any collegiate for professional level CTF competition.

How Do I Get Started with picoCTF?

Easy! Go to:

http://2013.picoctf.com

Create a team name and password and go for it!

Purpose of this Guide

The purpose of this guide is twofold:

1. To provide a quick way to prepare to integrate a CTF competition into their classroom for an experienced Computer Science teacher
2. To provide an outline for a set of long term projects to allow a less experienced Computer Science teacher to integrate a CTF competition in their curriculum.

This guide **cannot** directly provide all the background knowledge to teach a CTF as such a document would number in the hundreds of pages. What it can do, however, is give you a set of resources to begin this journey.

The sheer amount of topics will tend to be overwhelming, so the reader is discouraged from attempting to learn everything at once. Pick a topic here, then another topics, and then yet another and you will soon find that you can do most if not all the easy to medium level of problems.

Ethical Concerns

Do realize that you are teaching students techniques that can do great damage if used improperly. It is important to constantly remind students that they are given tools that they have to use wisely.

In 10 years of teaching, I have never had a situation where a student has used the tools to cause problems for either my school or the outside world. I attribute this to two reasons:

1. I make very clear that to do so would mean immediate removal from the program.
2. I try to make the program so interesting that they are too busy hacking targets created for them to attack other targets.

I will be blunt for a moment and say that most schools are completely ill equipped to deal with students trained in computer security. I do nothing to hide this from the students (they already know, trust me), but make a point of explaining that attack the schools security is a waste of time because:

1. It is an exercise in triviality (I have been known the use the phrase, "Like stealing candy from a baby").
2. It puts the program itself in danger.

With that said, it is highly recommended that you discuss the discipline ramifications with your administration and be sure to have your IT staff on board.

Technical Concerns

Federal law, in the United States, requires that schools employ filtering software which very often block the very information you and your student's need to complete the tasks in PicoCTF. With hope, your IT people will be able to work with you to open up the necessary search terms (I can almost guarantee you many will be blocked, don't wait until the last minute to check!).

In a perfect world, students will have access to several operating systems. (at least Linux, Mac OSX, and Windows). Linux is highly recommended for reasons explained in a bit.

Also in a perfect world, students should have access to several browsers (at least Chrome, Firefox, Safari). Internet Explorer is solely unsuited for the task.

What Background Should You, the Educator, have?

Ideally, you should be very comfortable as a user of either Mac OS X or Windows and have some programming experience is some language.

What Should You, the Educator, Be Able To Do?

Again ideally, you should be able to handle a majority of the easy to medium level problems before attempting to use this in a group setting. PicoCTF starts with problems that are as simply as using Google and ends with problems that even some computer security professionals struggle with.

What Should Your Students Background Be?

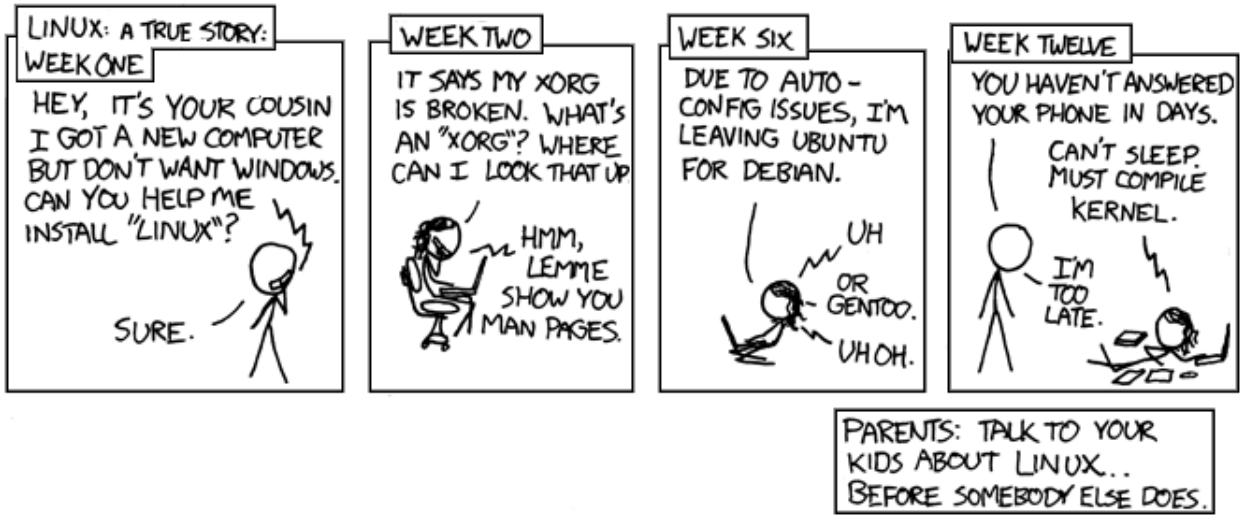
They should be able to Google and should have some experience just about any text based programming language (Scratch and the like will not be so useful).

What Should Your Students Be Able To Do?

PicoCTF was specifically designed to be accessible even by students with little to no computer security. Inexperienced students should be able to score between 500 and 1000 points. Experienced students should be able to score between 1000 and 2000 points.

Why Linux?

Operating systems wars are pretty much the religious argument of the computer science world and the reality is that the best way to compete in the competition is to use many different operating systems (at least Mac OS X, Linux and Windows). With that said, there are many things that are significantly easier on a Linux system. It is highly recommend that you get up to speed in Linux. Especially since the assumption is that you are reasonably proficient on either Mac OS X or Windows.



https://xkcd.com/456/

Linux Distributions

More religious wars! I recommend beginning users start out with a distribution known as Ubuntu. My first choice is to install it on a real computer (it can be significantly old and slow, perhaps something top of the line in 2005). For even slower computers, look into something called Lubuntu.

For the more advanced computer programming teachers, look into a distribution named Kali.

A Brief Introduction to Linux Commandline

A One Page Manual to the Linux Command Line:
http://www.digilife.be/quickreferences/QRC/The%20One%20Page%20Linux%20Manual.pdf

ls
list files to command line

ssh
Secure Shell – used to connect from one computer to another via the command line. Information on setting up ssh on your Linux machine can be found at:

`http://www.tecmint.com/install-openssh-server-in-linux/`

mkdir
Make Directory

cd
Change Directory

Text editors: vi, pico/nano

cat
concatenate – combine two files (or print one file out) to the command line

grep
grep is a utility that allows you to search for text in one or more files.

wget
Web Get – retrieve files from the web via the command line. Think of this like a web browser for the commandline that can generally only download files (i.e. it is not interactive).

unzip
unzip decompress a .zip file into it's compoenent files.

When in doubt:

man <command>
Prints out a manual for any standard Linux command

When in real doubt:

Google!
`https://www.google.com/search?q=grep`

Text Editors

Text editors are yet another source of religious arguments. Some recommended text editors for Linux:

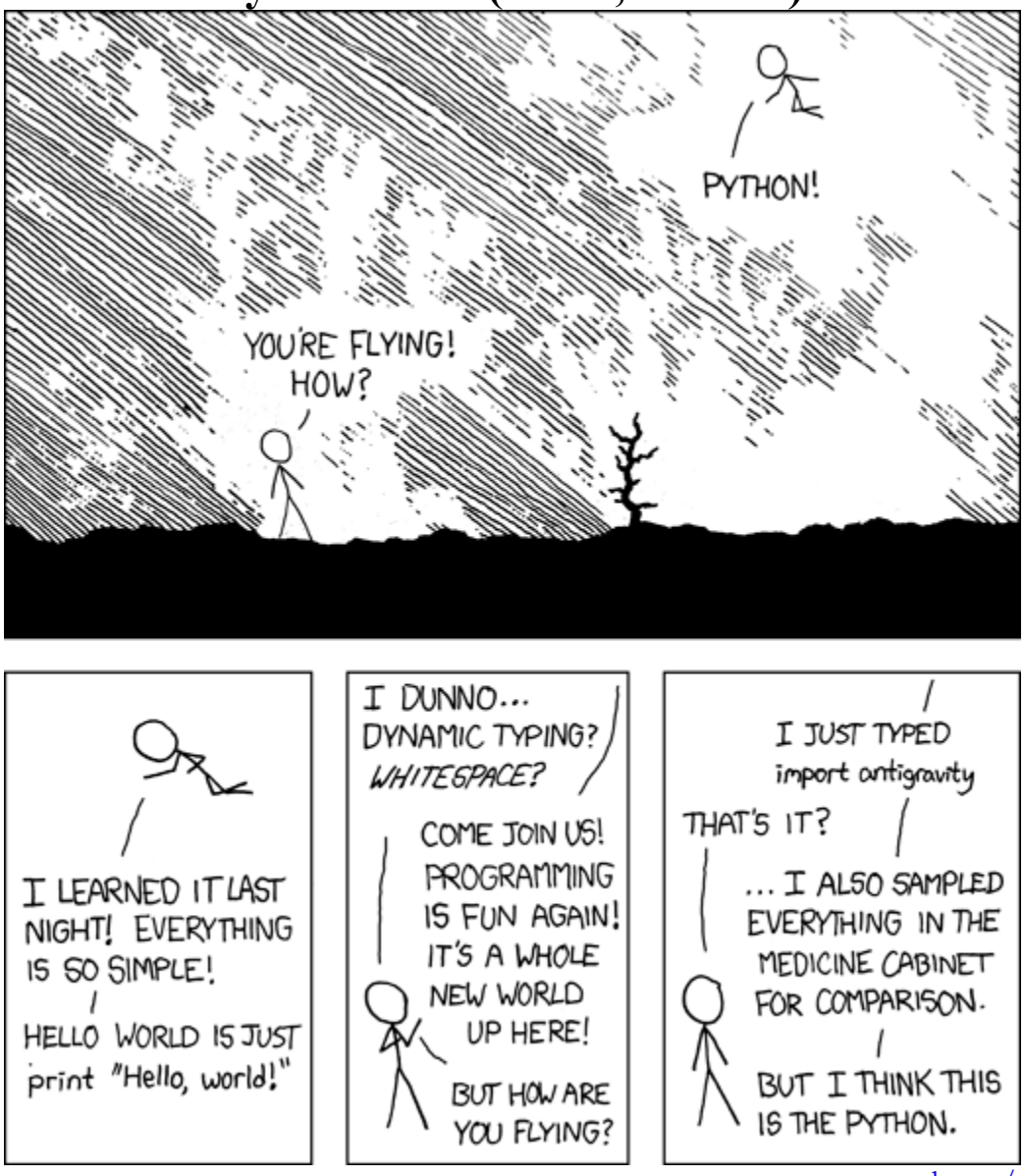
Command line based: vi (intermediate to advanced), nano (beginners)
Graphical: gedit (beginners)

It is very important that you choose a true text editor and not a word processing program. For Windows, this author is a fan of a free product named NotePad++. For Mac OS X, this author is a fan of a free product named TextMate.

It is highly recommend that you and your students learn a command line text editor, as a good portion of the work done by hackers revolves around the command line. Tutorial web pages abound, here is just one:

http://heather.cs.ucdavis.edu/~matloff/UnixAndC/Editors/ViIntro.html

Introductory Activities (Hello, World!)



https://xkcd.com/353/

The Purpose of a "Hello, World" Program

It might seem overwhelming be the sheer amount of programming languages involved in PicoCTF. Please understand that this guide in no way expects you to master all these languages in one sitting. Hello, World problems are not about learning the language but rather about learning how to compile, execute, or otherwise use the language. With hope, each of these Hello, World solutions will put you in a situation to purchase any of a number of books or visit any number of websites on these topics and get to work.

Introductory Python (version 2.7):

Python is considered possibly the easiest text based language for beginners to write a hello world program. Create a file called hello.py with your chosen text editor and put the following code into it:

```
print "Hello World"
```

Run the program by typing:

```
python hello.py
```

Python should be installed on most Linux and Mac OS X computers. Windows computers will generally have to have it installed:

https://www.python.org/downloads/

A great and free resource for learning python is available at:

http://learnpythonthehardway.org/

Introductory HTML:

HTML, Hypertext Markup Language, is exactly as the initials describe: a markup language not a programming language. With that said, an understanding of HTML is integral to success in PicoCTF.

You can start by creating a file named `index.html` with your chosen text editor on any operating system. Then directly open the file from your web browser.

```
<html>
Hello, World<br />
</html>
```

Definitely consider setting up your own webserver with the Linux operating system.

https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu

Once you have followed the above instructions, move your `index.html` file into `/var/www/html`

You should be able to open the file using your Linux browser (likely Firefox) by inputting the URL: http://localhost

Technically, the file is now available for viewing on your local network. Look into the `ifconfig` command to determine what address your Linux machine has on the network.

A great resource for further information on HTML can be found at:

http://www.w3schools.com

Introductory C:

C is probably the oldest and most venerable languages involved in PicoCTF and this author's preferred language as many languages (including PHP, C++, Objective C, Java, and JavaScript) base a good portion of their syntax off of C.

```
#include <stdio.h>
int void main()
{
  printf("Hello, World\n");

  return 0;
}
```

Compile:

```
clang hello.c -o hello
```

Execute:

```
./hello
```

You may have to install the clang compiler (gcc is also a good compiler option). A great place to start in learning C is:

http://cs50.tv

This is Harvard's introductory class which uses the C language for the first half the class. As an added bonus, it covers PHP and JavaScript and many of the more advanced topics in computer science security in the second half of the class.

Introductory PHP:

PHP is a so-called server side language which means the language is processed on the web server before it sent to the browser. It can be run as a standalone (which is almost never done) and is almost always used in conjunction with HTML

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
 </body>
</html>
```

Notes:

Files should be named index.php

Enter: `http://localhost/index.php`

For PHP, a web server installed at `locahost` with supported php installation is absolutely necessary.

A great resource for learning php is:

`https://www.codecademy.com/tracks/php`

Introductory JavaScript:

JavaScript, interestingly enough, has nothing to do with Java and was created by Netscape (at one point makers of the dominant web browser platform) to make web pages more interactive. JavaScript is a client side scripting language, hence the need for a browser that supports JavaScript but not requirement that the server support JavaScript.

```
<html>
<body>

  <p>Header...</p>

  <script>
    alert('Hello, World!')
  </script>

  <p>...Footer</p>

</body>
</html>
```

Notes:
Files should be named index.html (or index.php)

Web browser
Enter: `http://localhost/index.php`
Assumes browser with supported Javascript implementation

A great resource for learning JavaScript is:

`https://www.codecademy.com/tracks/javascript`

Introductory Java:

The Java language was originally conceived in the 1990s to write so called "smart" appliances. Very quickly, the creators discovered there wasn't much interest in smart appliances at the time and Java was repurposed for use in the burgeoning Internet market.

Java is unique in that it is both a compiled and interpreted language. The java compiler turns the java code into a platform independent set of byte codes which are run in a so-called virtual machine. Hence the requirement to first run your program through javac (which generates a java .class file) and the run the .class file through the java virtual machine.

A very important tip for PicoCTF is the knowledge that while decompiling programs back into source code is generally difficult in most language, Java programs can be reconstituted almost entirely with various offline and online web based Java decompilers. Remember this for any problems that give you a .class file.

```
class HelloWorldApp
{
    public static void main(String[] args)
{
        System.out.println("Hello World!");
    }
}
```

Compile:
`javac HelloWorldApp.java –o hello`

Execute:
`java HelloWorldApp`

Encryption/Decryption:

Casear Cipher
Caesar Cipher is the most simplest of encryption schemes where each letter is rotated. Every letter is rotated by the same number so a Caesar Cipher of 1 would turn:

`abcd` into `bcde`

Caesar Cipher can be decoded by brute force, that is trying ever one of the 26 possible rotations until you find a decoding that produces readable text. There are also several website that can automate this decryption, which students are encouraged to search for themselves.

Substitution
Substitution Cipher directly replaces one letter with another. Thus, given the following substitution:
`h = a`
`e = z`
`l = r`
`o = f`

`hello` becomes `azrro`

Frequency Analysis
Substitution cipher is a bit more resistant to brute force methods, but both substitution and Caesar ciphers have an inherent limitation based on the reality that in any given text, the letter E is most likely to be the most common character. It is a simple matter to look up the most likely letters and take a few educated guesses until something understandable results.

Even in the more complicated ciphers, there typically is a web service that will break the code for you. I would encourage you and your students to do a bit of encoding and decoding by hand, but use the services for larger problems.

Other Tools:

Wireshark
Wireshark is a TCP/IP based network monitoring tool. Several of the PicoCTF problems us a related service named cloudshark to allow students to examine a predetermined set of network traffic.

The problems tend to look very complicated, but in the end they almost also are an exercise in looking for a needle in a haystack. Neither you or your students need to have an exhaustive understand of either Wireshark or TCP/IP to be successful with these problems as long as you are able to look through the packets and have the patience to not give up.

Wireshark is a very powerful program that can give students to all sorts of data if your network is unencrypted (it likely is), so tread carefully here.

More about Wireshark at:

https://www.wireshark.org/

hexedit
You will need some sort of hexadecimal editor (hexeditor) which will allow you to look at binary files directly. Any hexeditor will do, Linux users are able to install hexedit with this command:

```
sudo apt-get install hexedit
```

And there are numerous free hexeditors available for both Mac OS X and Windows.

photoshop (or GIMP)
There is at least one problem which requires stopping an animated GIF file. Photoshop and the open source GIMP are both capable and, as of late, there are websites that are able to separate animated GIF files into their component pictures.

strings
`strings` is the unsung hero of PicoCTF and is installed on most Linux systems. `strings` has a simple function, it reads in any binary file and tries to find anything that might be a remotely intelligible string. Here is an example of `strings` running on a C hello world program:

```
username@mylinuxbox:~/dev$ strings hello
/lib64/ld-linux-x86-64.so.2
7Csw/
libc.so.6
puts
__libc_start_main
__gmon_start__
GLIBC_2.2.5
UH-@
UH-@
[]A\A]A^A_
Hello World!
;*3$"
```

zip/unzip
Many times, files will be compressed and can be uncompressed using the linux `unzip` progam. This unto itself does not make it worthy of mention, but it is worthwhile to know that there are ways to repair damaged zip files and sometimes people will even hide a zip file within a zip file.

Some PicoCTF 2013 problems:

XMLOL: 30
This is perhaps the most trivial (other than the Google related) problems. There are at least two ways to solve the problem:

1. Right click and View page source.
2. `wget` the file on a Linux machine and view it with a text editor

In either case, you should quickly see the flag:

```
<super_secret_flag>569256581303012324121093577</super_secret_fl
ag>
```

Grep is Your Friend: 40
This problem is not too difficult if you are using a Linux machine or any computer with the program grep.

The first step is to `wget` the file:

```
wget https://2013.picoctf.com/problems/grep.tar
```

It is recommended that you move the file into its own directory as it will expand to a large amount of files shortly.

A .tar file is a format for combined files into one file for easy transport. The term tar comes from Tape ARchive, but it might be easier remembered as sticking files to each other.

To extract the file:
```
tar -xvf grep.tar
```

Luckily, grep can make short work of this problem. Run:
```
grep -r SECRET
```

And you'll quickly see:

```
fHYYpdrfeOCHyQicfe96xfw==:SECRET AUTH CODES
```

This brings up two common themes in picoCTF:

1. Hints abound in every part of the problem description, especially the title.
2. The flags are not subtle. If you or your students are doubting whether they have found the flag, it probably isn't the flag.

GETKey: 50

Not so secure PHP:
password.php:
```
<?php

  ini_set('display_errors','On');
  error_reporting(E_ALL);

  if ($_GET)
  {
    $username = $_GET['username'];
    $password = $_GET['password'];
    echo 'Your Username: ' . $username . '<br />';
    echo 'Your password: ' . $password . '<br /><br />'
  }
?>

<html>
<body>
  <form action="" method="GET">
Username: <input type="text" name="username"><br /><br />
Password: <input type="password" name="password"><br/><br />
  <input type="submit" value="Login!"><br /><br />
```

```
  </form>

</body>
</html>
```

It might be tempting to say this code is insecure because it prints out the Username and Password of the user and there is some truth to that. However, it is important to know that by using the GET method the username and password are passed as parameters in the URL:

http://myserver.org/password.php?username=MyUsername&password=MyPassword

Such problems are easily rectified by changing to the use of the PUT method:

passwordsecure.php:

```
<?php

  ini_set('display_errors','On');
  error_reporting(E_ALL);

  if ($_GET)
  {
    $username = $_PUT['username'];
    $password = $_PUT['password'];
    echo 'Your Username: ' . $username . '<br />';
    echo 'Your password: ' . $password . '<br /><br />'
  }
?>

<html>
<body>
  <form action="" method="PUT">
Username: <input type="text" name="username"><br /><br />
Password: <input type="password" name="password"><br/><br />
  <input type="submit" value="Login!"><br /><br />
  </form>

</body>
</html>
```

Which results in the following URL:

http://myserver.org/passwordsecure.php

GETKey takes advantage of security issues arising from improper use of the GET method. You will notice the parameters listed and consider what value admin (true) should have and which competition you are in (picoctf).

It isn't necessary that students understand the details of put and get to succeed with this problem,

so the above code is given both for your understanding and for potential use if you wish to get into the details with your students.

Byte Code: 70

The first step is to wget the file:

`wget https://2013.picoctf.com/autoproblems/tmpfIr7XA.zip`

The second step is to unzip the file:

`unzip tmpfIr7XA.zip`

This should give you the following file:

`Authenticator.class`

A quick run through:

http://www.javadecompilers.com/

Gives us:

```
/*
 * Decompiled with CFR 0_101.
 */
import java.io.Console;
import java.io.PrintStream;

class Authenticator {
    public static char[] key;

    public static void main(String[] arrstring) {
        key = new char[10];
        Authenticator.key[0] = 121;
        Authenticator.key[1] = 104;
        Authenticator.key[2] = 109;
        Authenticator.key[3] = 65;
        Authenticator.key[4] = 114;
        Authenticator.key[5] = 109;
        Authenticator.key[6] = 72;
        Authenticator.key[7] = 87;
        Authenticator.key[8] = 120;
        Authenticator.key[9] = 117;
        Console console = System.console();
        String string = "";
        while (!string.equals("ThisIsth3mag1calString4679")) {
            string = console.readLine("Enter password:", new Object[0]);
        }
        for (int i = 0; i < key.length; ++i) {
            System.out.print(key[i]);
        }
```

```
        System.out.println("");
    }
}
```

Run the program:

`java Authenticator`

Enter the magic password (we're not quite at the flag): `ThisIsth3mag1calString4679`

You should receive the final flag as: `yhmArmHWxu`

Where Do I Go From Here

First, don't be afraid to let your students try picoCTF, even if you haven't completely mastered the material. The goal of this guide is to give you any number of jump off points to explore your own interests and anyone who is thoroughly grounded in every aspect tested in picoCTF is likely qualified to make a six figure salary, so you have nothing to apologize for by not knowing everything.

If all else fails, Harvard's CS50 class is a great (and free!) next step:

http://cs50.tv