New Exam Update

Table of Content

Class	Topics
24	• Regulatory and environmental • requirements
	Recovery: • In-place/overwrite • Alternative location

Table of Content (cont'd)

Class	Topics
24	Incident response: • Order of volatility Licensing/digital rights management (DRM)/end -user license agreement (EULA): • Perpetual license agreement Non-disclosure agreement (NDA)/mutual non-disclosure agreement (MNDA): • Acceptable use policy (AUP) • Regulatory and business compliance requirements • Splash screens
	• Shell Scripts • Basic Script Constructs • Windows Scripts • JavaScript and Python • Use Cases for Scripting • Scripting Best Practices and Considerations

Table of Content (cont’d)

Class	Topics
24	Methods/tools: • Simple Protocol for Independent Computing Environments (SPICE) • Windows Remote Management (WinRM)

Summary: What's New

New Objectives	Old Objectives	New Topics	Labs
2.9 Compare and contrast data destruction and disposal methods.	2.8 Given a scenario, use common data destruction and disposal methods.	• Regulatory and environmental • requirements	• Formatting and Cleaning a Disk
4.3 Given a scenario, implement workstation backup and recovery methods.	4.3 Given a scenario, implement workstation backup and recovery methods.	Recovery: • In-place/overwrite • Alternative location	• Restore Data from File History • Create Backups in Windows 09IrV • Create Backups in Linux 09IrU • Configure File History • Create a Restore Point

*Note: New objectives are now in effect for the CompTIA A+ 220-1201 exam.

Summary: What’s New (cont’d)

New Objectives	Old Objectives	New Topics	Labs
4.6 Explain the importance of prohibited content/activity and privacy, licensing, and policy concepts.	4.6 Explain the importance of prohibited content/activity and privacy, licensing, and policy concepts.	Incident response: • Order of volatility Licensing/digital rights management (DRM)/end -user license agreement (EULA): • Perpetual license agreement Non-disclosure agreement (NDA)/mutual non-disclosure agreement (MNDA): • Acceptable use policy (AUP) • Regulatory and business compliance	• Completing the Chain of Custody Form

*Note: New objectives are now in effect for the CompTIA A+ 220-1201 exam.

Summary: What's New (cont'd)

New Objectives	Old Objectives	New Topics	Labs
4.8 Explain the basics of scripting.	4.8 Explain the basics of scripting.	• Shell Scripts • Basic Script Constructs • Windows Scripts • JavaScript and Python • Use Cases for Scripting • Scripting Best Practices and Considerations	• Creating and executing a powershell script • Creating and executing a python script • Creating and executing a shell script
4.9 Given a scenario, use remote access technologies.	4.9 Given a scenario, use remote access technologies.	Methods/tools: • Simple Protocol for Independent Computing Environments (SPICE) • Windows Remote Management (WinRM)	• Configuring an SSH Server

*Note: New objectives are now in effect for the CompTIA A+ 220-1201 exam.

Module 21: Using Data Security

2.9 Compare and contrast data destruction and disposal methods

Regulated Data Classification

- Sensitive data types
  - PII: Identifies individuals (e.g., SSN, email, biometrics)
  - Healthcare: Medical records, anonymized or de-identified
  - Credit card: Governed by PCI DSS (e.g., card numbers, CV2)
- Best practices
  - Train employees on secure data handling
  - Avoid unauthorized access or exposure
  - Prevent careless actions (e.g., unencrypted data)

4.3 Given a scenario, implement workstation backup and recovery methods

Backup Testing and Recovery Best Practices

Testing

- Restore to test directory or VM
- Verify data integrity and file inclusion
- Test regularly (weekly/monthly)

Recovery

- In-place: Restore to original location, may cause downtime
- Alternate: Restore to new hardware/cloud for major failures

4.6 Explain the importance of prohibited content/activity and privacy, licensing, and policy concepts

Incident Responses

- Incidents
  - Malware, data breaches, phishing, DoS, unlicensed software, prohibited content
- Response plan
  - Follow procedures; notify CIRT or law enforcement
  - CIRT: Handles incidents; includes technicians, managers, and decision-makers

Incident Response Role-Playing

- Review the incident details provided by the facilitator.
- Develop a response plan, including containment, investigation, and communication steps.
- Role-play your response in front of the class.

Policy

- AI usage policies in the workplace
  - Implement policies to regulate AI use in workflows
  - Acceptable Use Policy (AUP) sets guidelines and legal compliance
  - Define AI-related plagiarism and attribution rules
  - Clarify AI’s role in writing and restrict AI-generated submissions

Module 22: Implementing Operational Procedures

4.8 Explain the basics of scripting

Shell Scripts

- Coding and scripting
    - Shell: OS-specific commands
    - General: OS-independent, uses an interpreter
    - Programming: Compiled apps
    - Linux Scripts: use .sh, set execute (chmod +x), run via ./script.sh

Basic Script Constructs

- Syntax: Errors prevent execution; use correct format
- Comments: # for ignored notes
- Variables: Store/change values; $1, $2 for input
- Control flow: If (branches), for/while/until (loops)
- Operators: ==, !=, <, >, &&, ||
- Best practice:
  - Avoid infinite loops, handle variables properly

Windows Scripts

- PowerShell
  - Uses cmdlets (Verb-Noun format), .ps1 files, and PowerShell ISE
- VBScript
  - Legacy scripting (.vbs), runs via wscript.exe or cscript.exe
- Batch Files
  - CMD-based scripts (.bat) for simple automation

JavaScript and Python

- JavaScript
  - Web scripting (.js), runs in browsers, servers, Windows, and macOS (JXA)
- Python
  - General-purpose (.py/.pyw), runs via interpreter or as an executable
- Python 3
  - Preferred; Python 2 is obsolete (EOL 2020)

Use Cases for Scripting

Execution

- Automate tasks via commands or APIs

Cross-platform

- Python calls Bash and PowerShell

Tasks

- Restart, map drives, install apps, update, backup, gather data

Best practices

- Use error handling & automation tools

Scripting Challenge

- Choose a scripting language (e.g., Bash, PowerShell, or Python).
- Write a script to:
  - Create directories for three users.
  - Assign read/write permissions to each user for their directory.
- Test the script in a simulated environment.

Scripting Best Practices and Considerations

- Malware risks
  - Restrict access, scan for vulnerabilities, use minimal privileges
- System risks
  - Test in development, update baselines, avoid security gaps
- Resource mishandling
  - Prevent infinite loops, manage storage, monitor execution

4.9 Given a scenario, use remote access technologies

Module 14: Supporting Windows

4.8 Explain the basics of scripting

WinRM

- Allows systems to exchange access management information
- As a Simple Object Access Protocol, it relies on HTTP/HTTPS connections to communicate
- All current Windows already include the WinRM app

Simple Protocol for Independent Computing Environments (SPICE)

- Provides remote display system to monitor and interact with VM environments across Internet
- Allows server to monitor and interact with each client

The Official CompTIA A+ Core 1 Student Guide

CONTENTS

2.9 Compare and contrast data destruction and disposal methods.

Disposal and Recycling Outsourcing Concepts

If a media device is not being repurposed or recycled, physical destruction might be an appropriate disposal method. A disk can be mechanically destroyed in specialist machinery:

- Drilling - A disk can be destroyed using a drill on specific sections of the platters including the landing zone (where read/write heads rest when not in use) and along the data tracks. Safety goggles should always be worn when using this method. While safe for most cases, this method is not appropriate for the most highly confidential data as there is at least some risk of leaving fragments that could be analyzed using specialist tools.
- Shredding - The disk is ground into little pieces. A mechanical shredder works in much the same way as a paper shredder.
- Incinerating - The disk is exposed to high heat to melt its components. This should be performed in a furnace designed for media sanitization. Municipal incinerators may leave remnants.
- Degaussing - A hard disk is exposed to a powerful electromagnet that disrupts the magnetic pattern that stores the data on the disk surface. Note that degaussing does not work with SSDs or optical media.

There are many third-party vendors specializing in outsourced secure disposal. They should provide a certificate of destruction/recycling showing the make, model, and serial number of each drive they have handled plus date of destruction and how it was destroyed. A third-party company might also use overwriting or crypto-erase and issue a certificate of recycling rather than destruction.

These certificates are important to retain, especially when dealing with data that falls under certain regulatory requirements, such as HIPAA. These regulations will have strict requirements on how data should be destroyed. You should also make sure that all environmental regulations are followed as hard drives can contain hazardous materials and should be disposed of properly. Hard drives also contribute to the growing problem of e-waste and should be disposed of properly.

4.3 Given a scenario, implement workstation backup and recovery methods.

Recovery Options

When performing a recovery, you can choose to either perform an in-place (overwrite) recovery or recover to an alternate location.

An in-place recovery restores the data to the original system and location which overwrites the current system. This method is commonly used when recovering from minor issues such as a corrupted file. This method will most likely require downtime while the restoration process takes place.

Restoring to an alternate location restores the data to a different computer or even to an offsite cloud environment. If the data loss was due to a catastrophic hardware failure or major cyberattack leaving the original hardware inaccessible, this method is preferred. This method does require more planning but can lead to reduced disruption of business operations if the backup system is ready to go and the data can be quickly restored.

4.6 Explain the importance of prohibited content/activity and privacy, licensing, and policy concepts.

Incident Response

While performing technical support, you may have to report or respond to security incidents. A security incident could be one of a wide range of different scenarios, such as:

- A computer or network infected with viruses, worms, or Trojans.
- A data breach or data exfiltration where information is seen or copied to another system or network without authorization.
- An attempt to break into a computer system or network through phishing or an evil twin Wi-Fi access point.
- An attempt to damage a network through a denial of service (DoS) attack.
- Users with unlicensed software installed on their PC.
- Finding prohibited material on a PC, such as illegal copies of copyrighted material, obscene content, or confidential documents that the user should not have access to.

An incident response plan sets out procedures and guidelines for dealing with security incidents. Larger organizations will provide a dedicated computer incident response team (CIRT) as a single point-of-contact so that a security incident can be reported through the proper channels. The members of this team should be able to provide the range of decisionmaking and technical skills required to deal with different types of incidents. The team needs managers and technicians who can deal with minor incidents on their own initiative. It also needs senior decision-makers (up to director level) who can authorize actions following the most serious incidents.

The actions of staff immediately following the detection of an incident can have a critical impact on the subsequent investigation. When an incident is detected, it is critical that the appropriate person on the CIRT be notified so that they can act as the first responder, take charge of the situation, and formulate the appropriate response.

If there is no formal CIRT, it might be appropriate to inform law enforcement directly. Involving law enforcement will place many aspects of investigating the incident out of the organization's control. This sort of decision will usually be taken by the business owner.

One exception may be when you act as a whistleblower because you have proof that senior staff in the organization pose an insider threat or are disregarding regulations or legislation.

Order of Volatility

When recovering evidence, it is important to follow the order of volatility. This is the order that data should be collected based on how long it is likely to remain available. The most volatile data should be collected first and then go down the list to the data that is least likely to disappear.

A general order of volatility from most to least volatile is:

1. CPU cache and registers - This data is extremely volatile and changes rapidly.
2. Memory (RAM) - RAM is volatile memory, and its contents are lost when the power is turned off.
3. Temporary file system/swap space - This space is used for temporary storage and can contain valuable evidence.
4. Disk storage - This can include hard drives, SSDs, and other persistent storage devices. While persistent, this data can be overwritten or deleted.
5. Archival media - This includes any backup media such as USB drives, tape drives, etc. This data is the least volatile.

Digital Rights Management

Digital music and video are often subject to copy protection and digital rights management. When you purchase music or video online, the vendor may license the file for use on a restricted number of devices. You generally need to use your account with the vendor to authorize and deauthorize devices when they change. Most DRM systems have been defeated by determined attackers, and consequently, there is plenty of content circulating with DRM security removed. From an enterprise's point of view, this is prohibited content, and it needs monitoring systems to ensure that its computers are not hosting pirated content files.

End-User License Agreements

Prohibited content also extends to the unauthorized installation and use of software. When you install software, you must accept the license governing its use, often called the end-user license agreement. The terms of the license will vary according to the type of software, but the basic restriction is usually that the software may only be installed on one computer or for use by one single person at any one time.

An EULA might distinguish between personal and corporate/business/for-profit use. For example, a program might be made available as freeware for personal use only. If an employee were to install that product on a company-owned device, the company would be infringing the license.

Non-Disclosure Agreements

Non-disclosure agreements (NDA) are agreements designed to protect sensitive information. A NDA is a legally binding contract that obligates all parties to protect sensitive information that is shared between them. An NDA agreement can be either unilateral or mutual:

- In a unilateral NDA, one party shares sensitive data with the other party, and only the receiving party is obligated to keep the information secret. An example would be a company having employees sign an NDA before being allowed to work on a project that contains sensitive information.
- With a mutual NDA, both parties agree to protect each other's secrets. This type of agreement is common when two organizations partner together for a project.

Policy Documentation

An acceptable use policy (AUP) sets out what someone is allowed to use a particular service or resource for. Such a policy might be used in different contexts. For example, an AUP could be enforced by a business to govern how employees use equipment and services such as telephone or Internet access provided to them at work. Another example might be an AUP enforcing a fair use policy governing usage of its Internet access services.

Enforcing an AUP is important to protect the organization from the security and legal implications of employees (or customers) misusing its equipment. Typically, the policy will forbid the use of equipment to defraud, defame, or to obtain illegal material. It is also likely to prohibit the installation of unauthorized hardware or software and to explicitly forbid actual or attempted intrusion (snooping). An organization's acceptable use policy may forbid use of Internet tools outside of work-related duties or restrict such use to break times.

Further to AUPs, it may be necessary to implement regulatory compliance requirements as logical controls or notices. For example, a splash screen might be configured to show at login to remind users of data handling requirements or other regulated use of a workstation or network app.

4.8 Identify the basics of scripting

Shell Scripts

Coding means writing a series of instructions in the syntax of a particular language so that a computer will execute a series of tasks. There are many types of coding language and many ways of categorizing them, but three helpful distinctions are as follows:

- A shell scripting language uses commands that are specific to an operating system.
- A general-purpose scripting language uses statements and modules that are independent of the operating system. This type of script is executed by an interpreter. The interpreter implements the language for a particular OS.
- A programming language is used to compile an executable file that can be installed on an OS and run as an app.

The various types of scripting are often described as glue languages. Rather than implement an independent bit of software (as a programming language would), a

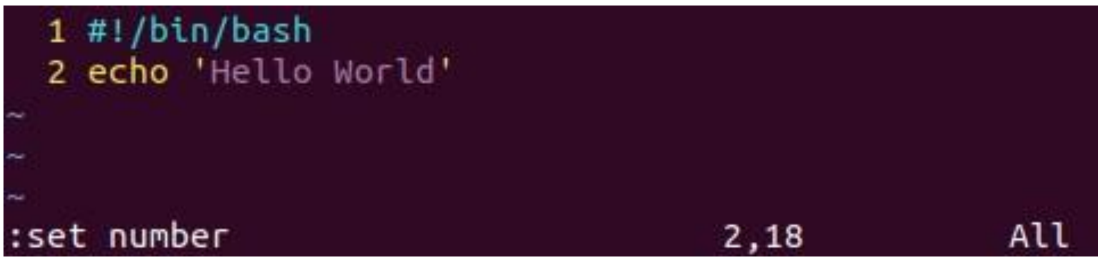
glue language is used to automate and orchestrate functions of multiple different OS and app software.

You can develop a script in any basic text editor, but using an editor with script support is more productive. Script support means the editor can parse the syntax of the script and highlight elements of it appropriately. For complex scripts and programming languages, you might use an integrated development environment (IDE). This will provide autocomplete features to help you write and edit code and debugging tools to help identify whether the script or program is executing correctly.

Linux shell script uses the sh extension by convention. Every shell script starts with a shebang line that designates which interpreter to use, such as Bash or Ksh. Each statement comprising the actions that the script will perform is then typically added on separate lines. For example, the following script instructs the OS to execute in the Bash interpreter and uses the `echo` command to write “Hello World” to the terminal:

```
#!/bin/bash echo 'Hello
World'
```

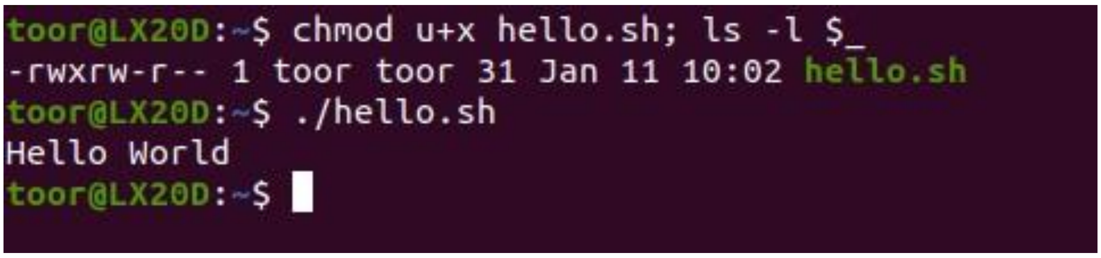
An example of a Linux shell script open in the Vim text editor



```
1 #!/bin/bash
2 echo 'Hello World'
~
~
~
:set number                                    2,18          All
```

Remember that in Linux, the script file must have the execute permission set to run. Execute can be set as a permission for the user, group, or world (everyone). If a PATH variable to the script has not been configured, execute it from the working directory by preceding the filename with `./` (for example, `./hello.sh`), or use the full path.

Setting execute permission for the user and running the script



```
toor@LX20D:~$ chmod u+x hello.sh; ls -l $_
-rwxrw-r-- 1 toor toor 31 Jan 11 10:02 hello.sh
toor@LX20D:~$ ./hello.sh
Hello World
toor@LX20D:~$
```

Basic Script Constructs

To develop a script in a particular language, you must understand the syntax of the language. Most scripting languages share similar constructs, but it is important to use the specific syntax correctly. A syntax error will prevent the script from running, while a logical error could cause it to operate in a way that is different from what was intended.

Comments

It is best practice to add comments in code to assist with maintaining it. A comment line is ignored by the compiler or interpreter. A comment line is indicated by a special delimiter. In Bash and several other languages, the comment delimiter is the hash or pound sign (#).

```
#!/bin/bash # Greet the
world echo 'Hello World'
```

Variables

A variable is a label for some value that can change as the script executes. For example, you might assign the variable `FirstName` to a stored value that contains a user's first name. Variables are usually declared, defined as a particular data type (such as text string or number), and given an initial value at the start of the routine in which they are used.

An argument or parameter is a variable that is passed to the script when it is executed. In Bash, the values `$1`, `$2` , and so on are used to refer to arguments by position (the order in which they are entered when executing the script). Other languages support passing named arguments.

A constant is a label for a value that remains constant throughout the execution of the script. Common constants in scripts may include tax percentages that are not changing during the script.

Branches and Loops

A script contains one or more statements. In the normal scheme of execution, each statement is processed in turn from top to bottom. Many tasks require more complex structures, however. You can change the order in which statements are executed based on logical conditions evaluated within the script. There are two main types of conditional execution: branches and loops.

Branches

A branch is an instruction to execute a different sequence of instructions based on the outcome of some logical test. For example, the following code will display "Hello Bobby" if run as `./ hello.sh Bobby` , executing the statement under "else". If run with no argument, it prints "Hello World":

```
#!/bin/bash # Demonstrate If syntax in
Bash if [ -z "$1" ] then echo 'Hello
World' else echo "Hello $1" fi
```

`-z` tests whether the first positional parameter (`$1`) is unset or empty.

Note: In the condition, the variable is enclosed in double quotes as this is a safer way to treat the input from the user (supplied as the argument). In the second echo statement, double quotes are used because this allows the variable to expand to whatever it represents. Using single quotes would print "Hello $1" to the terminal.

Loops

A loop allows a statement block to be repeated based on some type of condition. A "For" loop can be used when the number of iterations is predictable. The following command executes the `ping` command for each host address in 192.168.1.0/24:

```
#!/bin/bash # Demonstrate For syntax in
Bash for i in {1..254} do ping -c1
"192.168.1.$i" done
```

As well as "For" structures, loops can also be implemented by "While" statements. A "While" or "Until" loop repeats an indeterminate number of times until a logical condition is met. The following script pings the address supplied as an argument until a reply is received:

```
#!/bin/bash # Demonstrate Until syntax in Bash until ping -c1 "$1"
&/dev/null do echo "192.168.1.$1 not up" done

echo "192.168.1.$1 up"
```

The condition executes the ping command and tests the result. When a reply is received, ping returns true. The `&/dev/null` part stops the usual ping output from being written to the terminal by redirecting it to a null device.

Note: Make sure your code does not contain unintended or infinite loops. The loop above will continue until a reply is received, which could never happen.

Operators

Looping and branching structures depend on logical tests to determine which branch to follow or whether to continue the loop. A logical test resolves to a TRUE or FALSE value. You need to be familiar with basic comparison and logical operators:

Symbol Notation	Switch Notation	Usage
==	-eq	Is equal to (returns TRUE if both conditions are the same)
!=	-ne	Is not equal to (returns FALSE if both conditions are the same)
<	-lt	Is less than
>	-gt	Is greater than
<=	-le	Is less than or equal to
>=	-ge	Is greater than or equal to
&&	AND	If both conditions are TRUE, then the whole statement is TRUE
||	OR	If either condition is TRUE, then the whole statement is TRUE

Windows Scripts

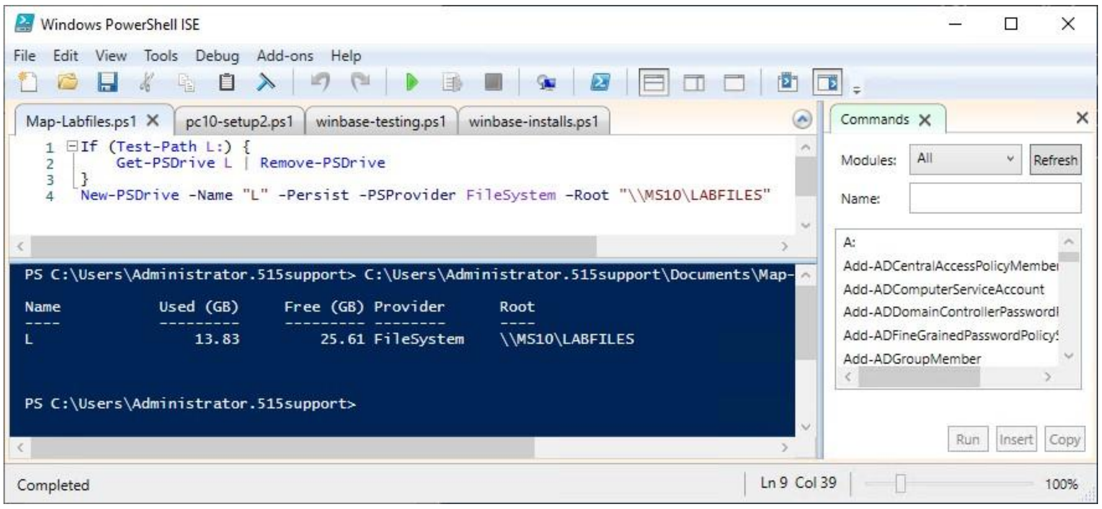
Windows supports several distinct shell coding environments. The three commonly used are PowerShell, Visual Basic Script, and the CMD interpreter.

Windows PowerShell

Windows PowerShell combines a script language with hundreds of prebuilt modules called cmdlets that can access and change most components and features of Windows and Active Directory. Cmdlets use a Verb-Noun naming convention. For example, `Write-Host` sends output to the terminal, while `Read-Host` prompts for user input.

Microsoft provides the Windows PowerShell Integrated Scripting Environment (ISE) for rapid development. PowerShell script files are identified by the .ps1 extension.

Windows PowerShell ISE



```
If (Test-Path L:) {
    Get-PSDrive L | Remove-PSDrive
}
New-PSDrive -Name "L" -Persist -PSProvider FileSystem -Root "\\MS10\LABFILES"
```

```
PS C:\Users\Administrator.515support> C:\Users\Administrator.515support\Documents\Map-

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
L                  13.83         25.61 FileSystem    \\MS10\LABFILES


PS C:\Users\Administrator.515support>
```

Screenshot courtesy of Microsoft.

VBScript

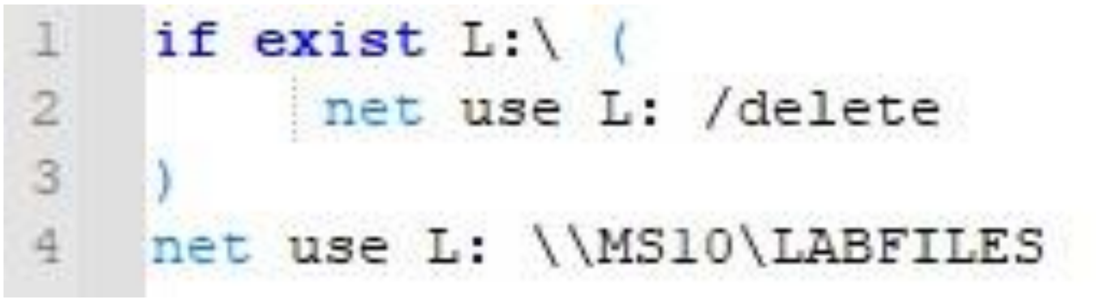
Visual Basic Script is a scripting language based on Microsoft's Visual Basic programming language. VBScript predates PowerShell. VBScript files are identified by the .vbs extension. VBScript is executed by the wscript.exe interpreter by default. Wscript.exe displays any output from the script in a desktop window or dialog. A script can also be run with cscript.exe to show output in a command prompt.

Note: You would now normally use PowerShell for Windows automation tasks. You might need to support legacy VBScripts, though.

Batch Files

A shell script written for the basic Windows CMD interpreter is often described as a batch file. Batch files use the .bat extension.

An example of a Windows batch file



```
if exist L:\ (
    net use L: /delete
)
net use L: \\MS10\LABFILES
```

Screenshot courtesy of Microsoft.

JavaScript and Python

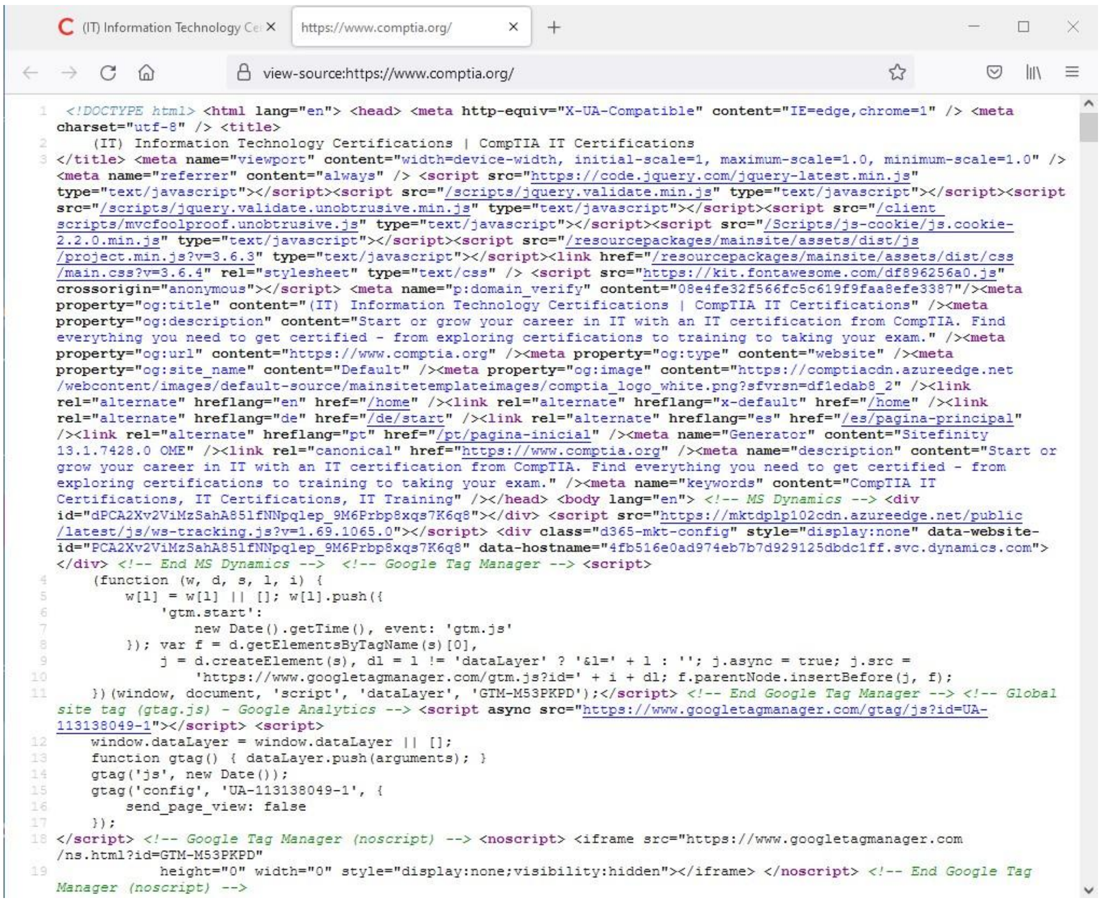
Bash and PowerShell/VBScript are closely tied to the Linux and Windows operating systems respectively. There are many other platform-independent scripting and programming languages.

JavaScript

JavaScript is a scripting language that is designed to implement interactive web-based content and web apps. Most web servers and browsers are configured with a JavaScript interpreter. This means that JavaScript can be executed automatically by placing it in the HTML code for a web page.

If not embedded within another file, JavaScript script files are identified by the .js extension. The Windows Script Host (wscript.exe and cscript.exe) supports JavaScript. JavaScript is also supported on macOS for automation (along with AppleScript). This is referred to as JavaScript for Automation (JXA).

JavaScript code embedded in a web page. Some code is loaded from .JS files from other servers; some code is placed within script tags.


(IT) Information Technology Ce
https://www.comptia.org/
view-source:https://www.comptia.org/

```
 <!DOCTYPE html> <html lang="en"> <head> <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" /> <meta
charset="utf-8" /> <title>
    (IT) Information Technology Certifications | CompTIA IT Certifications
</title> <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1.0, minimum-scale=1.0" />
<meta name="referrer" content="always" /> <script src="https://code.jquery.com/jquery-latest.min.js"
type="text/javascript"></script><script src="/scripts/jquery.validate.min.js" type="text/javascript"></script><script
src="/scripts/jquery.validate.unobtrusive.min.js" type="text/javascript"></script><script src="/client
scripts/mvcfoolproof.unobtrusive.js" type="text/javascript"></script><script src="/Scripts/js-cookie/js.cookie-
2.2.0.min.js" type="text/javascript"></script><script src="/resourcepackages/mainsite/assets/dist/js
/project.min.js?v=3.6.3" type="text/javascript"></script><link href="/resourcepackages/mainsite/assets/dist/css
/main.css?v=3.6.4" rel="stylesheet" type="text/css" /> <script src="https://kit.fontawesome.com/df896256a0.js"
crossorigin="anonymous"></script> <meta name="p:domain_verify" content="08e4fe32f566fc5c619f9faa8efe3387"/><meta
property="og:title" content="(IT) Information Technology Certifications | CompTIA IT Certifications" /><meta
property="og:description" content="Start or grow your career in IT with an IT certification from CompTIA. Find
everything you need to get certified - from exploring certifications to training to taking your exam." /><meta
property="og:url" content="https://www.comptia.org" /><meta property="og:type" content="website" /><meta
property="og:site_name" content="Default" /><meta property="og:image" content="https://comptiacdn.azureedge.net
/webcontent/images/default-source/mainsitetemplateimages/comptia_logo_white.png?sfvrsn=df1edab8_2" /><link
rel="alternate" hreflang="en" href="/home" /><link rel="alternate" hreflang="x-default" href="/home" /><link
rel="alternate" hreflang="de" href="/de/start" /><link rel="alternate" hreflang="es" href="/es/pagina-principal"
/><link rel="alternate" hreflang="pt" href="/pt/pagina-inicial" /><meta name="Generator" content="Sitefinity
13.1.7428.0 OME" /><link rel="canonical" href="https://www.comptia.org" /><meta name="description" content="Start or
grow your career in IT with an IT certification from CompTIA. Find everything you need to get certified - from
exploring certifications to training to taking your exam." /><meta name="keywords" content="CompTIA IT
Certifications, IT Certifications, IT Training" /></head> <body lang="en"> <!-- MS Dynamics --> <div
id="dPCA2Xv2ViMzSahA85lfNNpqlep_9M6Prbp8xqs7K6q8"></div> <script src="https://mktdplp102cdn.azureedge.net/public
/latest/js/ws-tracking.js?v=1.69.1065.0"></script> <div class="d365-mkt-config" style="display:none" data-website-
id="PCA2Xv2ViMzSahA85lfNNpqlep_9M6Prbp8xqs7K6q8" data-hostname="4fb516e0ad974eb7b7d929125dbdc1ff.svc.dynamics.com">
</div> <!-- End MS Dynamics -->  <!-- Google Tag Manager --> <script>
    (function (w, d, s, l, i) {
        w[l] = w[l] || []; w[l].push({
            'gtm.start':
                new Date().getTime(), event: 'gtm.js'
        }); var f = d.getElementsByTagName(s)[0],
            j = d.createElement(s), dl = l != 'dataLayer' ? '&l=' + l : ''; j.async = true; j.src =
                'https://www.googletagmanager.com/gtm.js?id=' + i + dl; f.parentNode.insertBefore(j, f);
    })(window, document, 'script', 'dataLayer', 'GTM-M53PKPD');</script> <!-- End Google Tag Manager --> <!-- Global
site tag (gtag.js) - Google Analytics --> <script async src="https://www.googletagmanager.com/gtag/js?id=UA-
113138049-1"></script> <script>
    window.dataLayer = window.dataLayer || [];
    function gtag() { dataLayer.push(arguments); }
    gtag('js', new Date());
    gtag('config', 'UA-113138049-1', {
        send_page_view: false
    });
</script> <!-- Google Tag Manager (noscript) --> <noscript> <iframe src="https://www.googletagmanager.com
/ns.html?id=GTM-M53PKPD"
            height="0" width="0" style="display:none;visibility:hidden"></iframe> </noscript> <!-- End Google Tag
Manager (noscript) -->
```

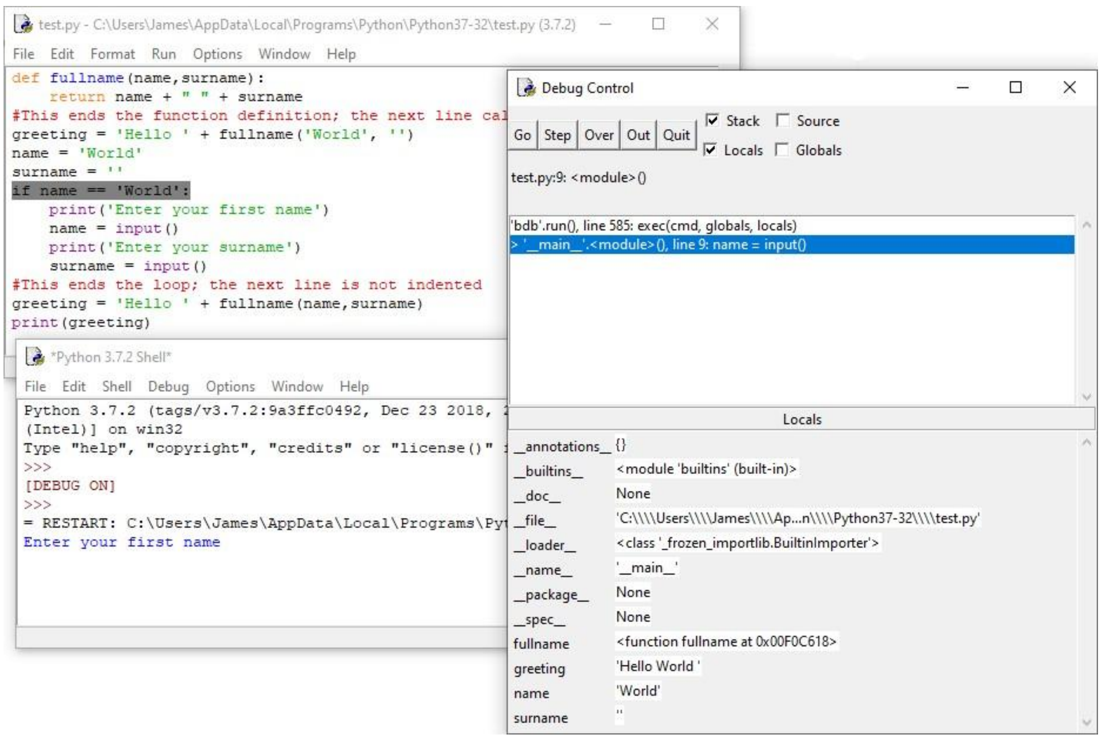
Screenshot courtesy of Mozilla.

Python

Python is a general-purpose scripting and programming language that can be used to develop both automation scripts and software apps. A Python project can either be run via an interpreter or compiled as a binary executable. There are several interpreters, including CPython (python.org) and PyPy (pypy.org). CPython is the simplest environment to set up for Windows.

Python script files are identified by the .py extension. When using CPython in Windows, there is a console interpreter (python.exe) and a windowed interpreter (pythonw.exe). The extension .PYW is associated with pythonw.exe.

Python Integrated Development and Learning Environment (IDLE)


test.py - C:\Users\James\AppData\Local\Programs\Python\Python37-32\test.py (3.7.2)
File Edit Format Run Options Window Help

```
def fullname(name,surname):
    return name + " " + surname
#This ends the function definition; the next line cal
greeting = 'Hello ' + fullname('World', '')
name = 'World'
surname = ''
if name == 'World':
    print('Enter your first name')
    name = input()
    print('Enter your surname')
    surname = input()
#This ends the loop; the next line is not indented
greeting = 'Hello ' + fullname(name,surname)
print(greeting)
```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" 
>>>
[DEBUG ON]
>>>
= RESTART: C:\Users\James\AppData\Local\Programs\Py
Enter your first name
```

Stack ✓ Source ☐ Locals ✓ Globals ☐

```
test.py:9: <module>()

'bdb'.run(), line 585: exec(cmd, globals, locals)
> '__main__'.<module>(), line 9: name = input()
```

Locals

__annotations__	{}
__builtins__	<module 'builtins' (built-in)>
__doc__	None
__file__	'C:\\\\Users\\\\James\\\\Ap...n\\\\Python37-32\\\\test.py'
__loader__	<class '_frozen_importlib.BuiltinImporter'>
__name__	'__main__'
__package__	None
__spec__	None
fullname	<function fullname at 0x00F0C618>
greeting	'Hello World '
name	'World'
surname	''

There are two major versions of Python: version 2 and version 3. Both to be installed at the same time. In Linux, using the keyword `python` executes a script as version 2, while `python3` executes a script in the version 3 interpreter. As of 2020, Python 2 is end of life (EOL), so scripts should be updated to version 3 syntax.

Use Cases for Scripting

One of the primary use cases for scripting is basic automation. Automation means performing some series of tasks that are supported by an OS or by an app via a script rather than manually. When using a local script environment, such as Bash on Linux or PowerShell on Windows, the script can use the built-in command environment.

When using a general-purpose language, such as Python, the script must use the operating system's application programming interface (API) to "call" functions. These API calls must be implemented as modules. Python has many prebuilt modules for automating Windows, Linux, and macOS. For example, the `os` module implements file system, user/permission functions, and process manipulation for whatever environment the interpreter is installed in. You can also use the interpreter in a more specific context. For example, `mod_python` implements a Python interpreter for the Apache web server software.

Another option is to call one script from another. For example, if you have some task that involves both Linux and Windows PCs, you might create a Python script to manage the task but execute Bash and PowerShell scripts from the Python script to implement the task on the different machines.

Restarting Machines

In an ideal world, no OS would ever need restarting. While Windows has made some improvements in this respect, many types of installation or update still require a reboot. In PowerShell, you can use the `Restart-Computer` cmdlet. The `-Force` parameter can be used to ignore any warnings that might be generated.

Linux is famous for its ability to run for any period without requiring a restart. However, should the need arise, the command to restart the host in Bash is `shutdown -r`

Remapping Network Drives

In a Windows batch file, the `net use` command performs drive mapping. The same thing can be done with PowerShell using the `New-PSDrive` cmdlet. This type of script demonstrates the need for error handling. If you try to map a drive using a letter that has been assigned already, the script will return an error. You can anticipate this by using an If condition to remove an existing mapping, if present:

```
If (Test-Path L:) {

    Get-PSdrive L | Remove-PSDrive

}

New-PSDrive -Name "L" -Persist -PSProvider FileSystem -Root "\\MS10\LABFILES"
```

Error handling is an important part of developing robust scripts.

Network drive mapping is a Windows-only concept. In Linux, a file system is made available by mounting it within the root file system, using the mount and umount commands.

Installation of Applications

In Windows, a setup file can be executed in silent mode by using the command switches for its installer. Installers are typically implemented either as .EXE files or as Windows Installer (.MSI) packages. To use an EXE setup in a batch file, just add the path to the installer plus switches:

```
C:\David\Downloads\setup.exe /S /desktopicon=yes
```

To use a Windows Installer, add the `msiexec` command:

```
msiexec C:\David\Downloads\install.msi /qn
```

You can also run these commands directly in a PowerShell script. However, the Start-Process cmdlet gives you more options for controlling the installation and handling errors.

In Linux, scripts are often used to compile apps from source code. You could also use a script to automate APT or YUM package management.

Initiating Updates

In Windows, the wusa.exe process can be called from a batch file to perform typical update tasks. In PowerShell, the `PSWindowsUpdate` module contains numerous cmdlets for managing the update process. Most third-party applications should support update-checking via an API.

In Linux, you can call `apt-get`/ `apt` or `yum` from your Bash script. The `-y` option can be used to suppress confirmation messages.

Automated Backups

At the command prompt, a simple type of backup can be performed by using ordinary file-copy tools, such as `robocopy` in Windows, or the script could call functions of a proper backup utility. The script can be set to run automatically by using Windows Task Scheduler or via cron in Linux.

Gathering of Information/Data

In Windows PowerShell, there are hundreds of Get verb cmdlets that will return configuration and state data from a Windows subsystem. For example, `Get-NetAdapter` returns properties of network adapters and `Get-WinEvent` returns log data. You can pipe the results to the `Where-Object` and `Select-Object` cmdlets to apply filters.

Bash supports numerous commands to manipulate text. You can gather data from the output of a command such as `ps` or `df` , filter it using `grep` , format it using tools like `awk` or `cut` , and then redirect the output to a file.

```
printf "Processes run by $1 on $(date +%F) at $(date +%T) \n"  "ps-$1.log" ps -ef |
grep "$1" | cut "$((${#1}+9))-"  "ps-$1.log"
```

This script reports processes by the username supplied as an argument to a log file, using the argument variable to name the file. The `printf` command appends a header with the date, time, and username. The second line filters `ps` output by the username uses the length of the argument variable plus nine to cut characters from each line and appends the output to the same log file.

Scripting Best Practices and Considerations

Deploying any type of code comes with the risk of introducing vulnerabilities. This means that deployment of scripts must be subject to best practices.

Malware Risks

There are several ways that a custom script could be compromised to allow a threat actor to install malware or perform some type of privilege escalation.

- If the interpreter is not a default feature, enabling it expands the attack surface. Threat actors use environments such as PowerShell to craft fileless malware.
- The threat actor could modify the source code to make it act as malware. In effect, the threat actor is using the script as a Trojan.
- The script could open a network port or expose some type of user form for input. If the script does not handle this input correctly, the threat actor could exploit a vulnerability to return unauthorized data or run arbitrary code.

To mitigate these risks, all script source code should be subject to access and version controls to prevent unauthorized changes. Code should be scanned and tested for vulnerabilities and errors before it can be deployed. Scripts should be configured to run with the minimum privileges necessary for the task.

Inadvertent System-Settings Changes

Another risk is from non-malicious or inadvertent threats where a script performs some unforeseen or unexpected system change. One example is accidental DoS, where a script powers off a system rather than restarting it or locks out remote access, perhaps by changing a firewall configuration. Other examples include weakening the security configuration by enabling the script environment, creating port exceptions, disabling scanning software so that the script executes successfully, and so on. Scripts that can only be made to work by disabling security mechanisms are not safe enough to consider running. Test all code in a development environment, and ensure that any changes to hosts that are required to run the scripts are included and updated/monitored through new configuration baselines.

Browser or System Crashes Due to Mishandling of Resources

Another way for a script to cause accidental DoS is through mishandling of resources. Some programming languages, such as C/C++, require very careful use of coding techniques to avoid creating vulnerabilities in the way the instructions manipulate system RAM. Scripting languages don't suffer from this type of vulnerability (they are considered safe with respect to memory handling), but coding mistakes can still lead to situations where the script mishandles computer or storage resources. Some examples are:

- Creating files that deplete disk storage resources, such as log files or temp files.
- Using a faulty loop code construct that does not terminate and causes the script to hang.
- Making a faulty API call to some other process, such as the host browser, that causes it to crash.

Every script must be tested to try to eliminate these kinds of mistakes before it is deployed, and its execution should be monitored to pick up any bugs that were not found in the test phase.

4.9 Given a scenario, use remote access technologies.

Remote Desktop Tools

With remote desktop, the target PC runs a graphical terminal server to accept connections from clients. This allows a user to work at the desktop of a different computer over the network.

Remote desktop is often configured for laptop users working from home with a slow link. Having gained access to the corporate network (via the Internet using a VPN, for example) they could then establish a remote desktop connection to a PC in the office. A technician can also use a remote desktop access tool to configure or troubleshoot a computer.

When allowing remote access to a host or network, you must assess and resolve security considerations:

- Remote access permissions should be granted to accounts selectively using least privilege principles.
- The connection must use encryption to be made secure against snooping. Users must have a means of confirming that they are connecting to a legitimate server to mitigate the risk of evil twin-type attacks. The server can be installed with a digital certificate to identify it securely.
- The server software supporting the connection must be safe from vulnerabilities, especially when the server port is accessible over the Internet.

Simple Protocol for Independent Computing Environments (SPICE)

The Simple Protocol for Independent Computing Environments provides a remote display system to monitor and interact with virtual machine environments from across the Internet. The server and client relationship is used for the protocol by allowing the server to monitor and interact with each client. Security of the machines is provided through various authentication options such as Kerberos. It also provides for a secure TLS mode which enables encryption of the traffic transmitted between the server and client machines.

WinRM

WinRM is Microsoft's implementation of the WS-Management protocol. It allows systems to exchange and access management information across a network. Being a Simple Object Access Protocol (SOAP) based program it relies on HTTP/HTTPS connections to communicate between the systems. The administrator can use the WinRM console to execute management commands on the system remotely. All current versions of Windows already include the WinRM application. It is used as one method to forward logs to a remote system for log collection and analysis.

Security concerns about the use of WinRM include the ability for remote attackers to execute their own commands against the network device. Additional concerns of traffic being passed in clear text format when using HTTP are unfounded since WinRM utilizes Kerberos to encrypt the traffic before passing it through the HTTP connection.