Networking Concepts

Introduction

A basic understanding of networking is important for anyone managing a server. Not only is it essential for getting your services online and running smoothly, it also gives you the insight to diagnose problems.

This document will provide a basic overview of some common networking concepts. We will discuss basic terminology, common protocols, and the responsibilities and characteristics of the different layers of networking.

This guide is operating system agnostic, but should be very helpful when implementing features and services that utilize networking on your server.

Networking Glossary

Before we begin discussing networking with any depth, we must define some common terms that you will see throughout this guide, and in other guides and documentation regarding networking.

These terms will be expanded upon in the appropriate sections that follow:

* **Connection**: In networking, a connection refers to pieces of related information that are transferred through a network. This generally infers that a connection is built before the data transfer (by following the procedures laid out in a protocol) and then is deconstructed at the at the end of the data transfer.
* **Packet**: A packet is, generally speaking, the most basic unit that is transferred over a network. When communicating over a network, packets are the envelopes that carry your data (in pieces) from one end point to the other.

Packets have a header portion that contains information about the packet including the source and destination, timestamps, network hops, etc. The main portion of a packet contains the actual data being transferred. It is sometimes called the body or the payload.

* **Network Interface**: A network interface can refer to any kind of software interface to networking hardware. For instance, if you have two network cards in your computer, you can control and configure each network interface associated with them individually.

A network interface may be associated with a physical device, or it may be a representation of a virtual interface. The "loopback" device, which is a virtual interface to the local machine, is an example of this.

* **LAN**: LAN stands for "local area network". It refers to a network or a portion of a network that is not publicly accessible to the greater internet. A home or office network is an example of a LAN.
* **WAN**: WAN stands for "wide area network". It means a network that is much more extensive than a LAN. While WAN is the relevant term to use to describe large, dispersed networks in general, it is usually meant to mean the internet, as a whole.

If an interface is said to be connected to the WAN, it is generally assumed that it is reachable through the internet.

* **Protocol**: A protocol is a set of rules and standards that basically define a language that devices can use to communicate. There are a great number of protocols in use extensively in networking, and they are often implemented in different layers.

Some low-level protocols are TCP, UDP, IP, and ICMP. Some familiar examples of application layer protocols, built on these lower protocols, are HTTP (for accessing web content), SSH, TLS/SSL, and FTP.

* **Port**: A port is an address on a single machine that can be tied to a specific piece of software. It is not a physical interface or location, but it allows your server to be able to communicate using more than one application.
* **NAT**: NAT stands for network address translation. It is a way to translate requests that are incoming into a routing server to the relevant devices or servers that it knows about in the LAN. This is usually implemented in physical LANs as a way to route requests through one IP address to the necessary backend servers.

Virtual Private Network (VPN)

A virtual private network (VPN) is a private network that is built over a public infrastructure. Security mechanisms, such as encryption, allow VPN users to securely access a network from different locations via a public telecommunications network, most frequently the Internet.

In some cases, virtual area network (VAN) is a VPN synonym.

VPN data security remains constant through encrypted data and tunneling protocols. The key VPN advantage is that it is less expensive than a private wide area network (WAN) buildout. As with any network, an organization's goal is to provide cost-effective business communication.

In a remote-access VPN, an organization uses an outside enterprise service provider (ESP) to establish a network access server (NAS). Remote users then receive VPN desktop software and connect to the NAS via a toll-free number, which accesses the organization's network. In a site-to-site VPN, many sites use secure data encryption to connect over a network (usually the Internet).

Network Layers

While networking is often discussed in terms of topology in a horizontal way, between hosts, its implementation is layered in a vertical fashion throughout a computer or network.

What this means is that there are multiple technologies and protocols that are built on top of each other in order for communication to function more easily. Each successive, higher layer abstracts the raw data a little bit more, and makes it simpler to use for applications and users.

It also allows you to leverage lower layers in new ways without having to invest the time and energy to develop the protocols and applications that handle those types of traffic.

The language that we use to talk about each of the layering scheme varies significantly depending on which model you use. Regardless of the model used to discuss the layers, the path of data is the same.

As data is sent out of one machine, it begins at the top of the stack and filters downwards. At the lowest level, actual transmission to another machine takes place. At this point, the data travels back up through the layers of the other computer.

Each layer has the ability to add its own "wrapper" around the data that it receives from the adjacent layer, which will help the layers that come after decide what to do with the data when it is passed off.

### The Open System Interconnection Model

The Open System Interconnection (OSI) model specifies how dissimilar computing devices such as Network Interface Cards (NICs), bridges and routers exchange data over a network by offering a networking framework for implementing protocols in seven layers. Beginning at the application layer, control is passed from one layer to the next. The following describes the seven layers as defined by the OSI model, shown in the order they occur whenever a user transmits information.

**Layer 7: Application**

This layer supports the application and end-user processes. Within this layer, user privacy is considered, and communication partners, service and constraints are all identified. File transfers, email, Telnet and FTP applications are all provided within this layer.

**Layer 6: Presentation (Syntax)**

Within this layer, information is translated back and forth between application and network formats.  This translation transforms the information into data the application layer and network recognize regardless of encryption and formatting.

**Layer 5: Session**

Within this layer, connections between applications are made, managed and terminated as needed to allow for data exchanges between applications at each end of a dialogue.

**Layer 4: Transport**

Complete data transfer is ensured as information is transferred transparently between systems in this layer. The transport layer also assures appropriate flow control and end-to-end error recovery.

**Layer 3: Network**

Using switching and routing technologies, this layer is responsible for creating virtual circuits to transmit information from node to node. Other functions include routing, forwarding, addressing, internet working, error and congestion control, and packet sequencing.

**Layer 2: Data Link**

Information in data packets are encoded and decoded into bits within this layer. Errors from the physical layer flow control and frame synchronization are corrected here utilizing transmission protocol knowledge and management. This layer consists of two sub layers: The Media Access Control (MAC) layer, which controls the way networked computers gain access to data and transmit it, and the Logical Link Control (LLC) layer, which controls frame synchronization, flow control and error checking.

**Layer 1: Physical**

This layer enables hardware to send and receive data over a carrier such as cabling, a card or other physical means. It conveys the bitstream through the network at the electrical and mechanical level. Fast Ethernet, RS232, and ATM are all protocols with physical layer components.

This order is then reversed as information is received, so that the physical layer is the first and application layer is the final layer that information passes through.

TCP/IP Model

The TCP/IP model, more commonly known as the Internet protocol suite, is another layering model that is simpler and has been widely adopted. It defines the four separate layers, some of which overlap with the OSI model:

* **Application**: In this model, the application layer is responsible for creating and transmitting user data between applications. The applications can be on remote systems and should appear to operate as if locally to the end user. The communication is said to take place between peers.
* **Transport**: The transport layer is responsible for communication between processes. This level of networking utilizes ports to address different services. It can build up unreliable or reliable connections depending on the type of protocol used.
* **Internet**: The internet layer is used to transport data from node to node in a network. This layer is aware of the endpoints of the connections, but does not worry about the actual connection needed to get from one place to another. IP addresses are defined in this layer as a way of reaching remote systems in an addressable manner.
* **Link**: The link layer implements the actual topology of the local network that allows the internet layer to present an addressable interface. It establishes connections between neighboring nodes to send data.

As you can see, the TCP/IP model, is a bit more abstract and fluid. This made it easier to implement and allowed it to become the dominant way that networking layers are categorized.

Interfaces

Interfaces are networking communication points for your computer. Each interface is associated with a physical or virtual networking device.

Typically, your server will have one configurable network interface for each Ethernet or wireless internet card you have.

In addition, it will define a virtual network interface called the "loopback" or localhost interface. This is used as an interface to connect applications and processes on a single computer to other applications and processes. You can see this referenced as the "lo" interface in many tools.

Many times, administrators configure one interface to service traffic to the internet and another interface for a LAN or private network.

In Digital Ocean, in datacenters with private networking enabled, your VPS will have two networking interfaces (in addition to the local interface). The "eth0" interface will be configured to handle traffic from the internet, while the "eth1" interface will operate to communicate with the private network.

Protocols

Networking works by piggybacking a number of different protocols on top of each other. In this way, one piece of data can be transmitted using multiple protocols encapsulated within one another.

We will talk about some of the more common protocols that you may come across and attempt to explain the difference, as well as give context as to what part of the process they are involved with.

We will start with protocols implemented on the lower networking layers and work our way up to protocols with higher abstraction.

Media Access Control

Media access control is a communications protocol that is used to distinguish specific devices. Each device is supposed to get a unique MAC address during the manufacturing process that differentiates it from every other device on the internet. Addressing hardware by the MAC address allows you to reference a device by a unique value even when the software on top may change the name for that specific device during operation. Media access control is one of the only protocols from the link layer that you are likely to interact with on a regular basis.

IP

The IP protocol is one of the fundamental protocols that allow the internet to work. IP addresses are unique on each network and they allow machines to address each other across a network. It is implemented on the internet layer in the IP/TCP model.

Networks can be linked together, but traffic must be routed when crossing network boundaries. This protocol assumes an unreliable network and multiple paths to the same destination that it can dynamically change between.

There are a number of different implementations of the protocol. The most common implementation today is IPv4, although IPv6 is growing in popularity as an alternative due to the scarcity of IPv4 addresses available and improvements in the protocols capabilities.

ICMP

ICMP stands for internet control message protocol. It is used to send messages between devices to indicate the availability or error conditions. These packets are used in a variety of network diagnostic tools, such as ping and traceroute.

Usually ICMP packets are transmitted when a packet of a different kind meets some kind of a problem. Basically, they are used as a feedback mechanism for network communications.

TCP

TCP stands for transmission control protocol. It is implemented in the transport layer of the IP/TCP model and is used to establish reliable connections.

TCP is one of the protocols that encapsulates data into packets. It then transfers these to the remote end of the connection using the methods available on the lower layers. On the other end, it can check for errors, request certain pieces to be resent, and reassemble the information into one logical piece to send to the application layer.

The protocol builds up a connection prior to data transfer using a system called a three-way handshake. This is a way for the two ends of the communication to acknowledge the request and agree upon a method of ensuring data reliability.

After the data has been sent, the connection is torn down using a similar four-way handshake.

TCP is the protocol of choice for many of the most popular uses for the internet, including WWW, FTP, SSH, and email. It is safe to say that the internet we know today would not be here without TCP.

UDP

UDP stands for user datagram protocol. It is a popular companion protocol to TCP and is also implemented in the transport layer.

The fundamental difference between UDP and TCP is that UDP offers unreliable data transfer. It does not verify that data has been received on the other end of the connection. This might sound like a bad thing, and for many purposes, it is. However, it is also extremely important for some functions.

Because it is not required to wait for confirmation that the data was received and forced to resend data, UDP is much faster than TCP. It does not establish a connection with the remote host, it simply fires off the data to that host and doesn't care if it is accepted or not.

Because it is a simple transaction, it is useful for simple communications like querying for network resources. It also doesn't maintain a state, which makes it great for transmitting data from one machine to many real-time clients. This makes it ideal for VOIP, games, and other applications that cannot afford delays.

HTTP

**Definition:**

HTTP stands for hypertext transfer protocol. It is a protocol defined in the application layer that forms the basis for communication on the web. HTTP works as a request-response protocol between a client and server. A web browser may be the client, and an application on a computer that hosts a web site may be the server.

Example: A client (browser) submits an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

**Explanation:**

In the beginning, network administrators had to figure out how to share the information they put out on the Internet. They agreed on a procedure for exchanging information and called it Hyper Text Transfer Protocol (HTTP).

Once everyone knew how to exchange information, intercepting on the Internet was not difficult. So knowledgeable administrators agreed upon a procedure to protect the information they exchanged. The protection relies on SSL Certificate to encrypt the online data. Encryption means that the sender and recipient agree upon a "code" and translate their documents into random-looking character strings. The procedure for encrypting information and then exchanging it is called Hyper Text Transfer Protocol Secure (HTTPS).

With HTTPS if anyone in between the sender and the recipient could open the message, they still could not understand it. Only the sender and the recipient, who know the "code," can decipher the message.

Humans could encode their own documents, but computers do it faster and more efficiently. To do this, the computer at each end uses a document called an "SSL Certificate" containing character strings that are the keys to their secret "codes."

SSL certificates contain the computer owner's "public key."

The owner shares the public key with anyone who needs it. Other users need the public key to encrypt messages to the owner. The owner sends those users the SSL certificate, which contains the public key. The owner does not share the private key with anyone.

The security during the transfer is called the Secure Sockets Layer (SSL) and Transport Layer Security (TLS).

The procedure for exchanging public keys using [SSL Certificate](https://www.instantssl.com/ssl-certificate.html) to enable HTTPS, SSL and TLS is called Public Key Infrastructure (PKI).

FTP

FTP stands for file transfer protocol. It is also in the application layer and provides a way of transferring complete files from one host to another.

It is inherently insecure, so it is not recommended for any externally facing network unless it is implemented as a public, download-only resource.

DNS

DNS stands for domain name system. It is an application layer protocol used to provide a human-friendly naming mechanism for internet resources. It is what ties a domain name to an IP address and allows you to access sites by name in your browser.

SSH

SSH stands for secure shell. It is an encrypted protocol implemented in the application layer that can be used to communicate with a remote server in a secure way. Many additional technologies are built around this protocol because of its end-to-end encryption and ubiquity.

URL

**Uniform Resource Locators**—**URLs**— are the web browser addresses of internet pages and files. With a URL, you can locate and bookmark specific pages and files for your web browser. URLs can be found all around us. They may be listed at the bottom of business cards, on TV screens during commercial breaks, linked in documents you read on the internet or delivered by one of the internet search engines. The format of a URL resembles this:

* http://www.examplewebsite.com/mypage

which is frequently shortened to this:

* www.examplewebsite.com/mypage

Sometimes they are longer and more complicated, but they all follow acknowledged rules for naming URLs.

URLs consist of three parts to address a page or file:

* The **protocol** is the portion ending in //: Most web pages use the protocol http or https, but there are other protocols.
* The **host or top-level domain**, which frequently ends in .com, .net, .edu or .org but can also end in one of many others that have been officially recognized.
* The **filename** or page name itself.

Browser

Beginning and advanced internet users all access the web through web browser software, which is included on computers and mobile devices at the time of purchase. Other browsers can be downloaded from the internet.

A browser is a free software package or mobile app that lets you view web pages, graphics, and most online content. The most popular web browsers include Chrome, Firefox, Internet Explorer, and Safari, but there are many others.

Browser software is specifically designed to convert HTML and XML computer code into human-readable documents.

Browsers display web pages. Each webpage has a unique address called a URL.

ISP

You need an Internet Service Provider to get to the internet. You may access a free ISP at school, a library or work, or you may pay a private ISP at home. An ISP is the company or government organization that plugs you into the vast internet.

An ISP offers a variety of services for a variety of prices: web page access, email, web page hosting and so on. Most ISPs offer various internet connection speeds for a monthly fee. You may choose to pay more for high-speed internet connection if you like to stream movies or select a less expensive package if you use the internet mostly for light browsing and email.

Encryption and Authentication

Encryption is the mathematical scrambling of data so that it is hidden from eavesdroppers. [Encryption](https://www.lifewire.com/introduction-to-network-encryption-817993) uses complex math formulas to turn private data into meaningless gobbledygook that only trusted readers can unscramble.

Encryption is the basis for how we use the internet as a pipeline to conduct trusted business, like online banking and online credit card purchasing. When reliable encryption is in place, your banking information and credit card numbers are kept private.

Authentication is directly related to encryption. Authentication is the complex way that computer systems verify that you are who you say you are.

Malware

[Malware](https://www.lifewire.com/what-is-malware-2625933) is the broad term to describe any malicious software designed by hackers. Malware includes viruses, trojans, keyloggers, zombie programs and any other software that seeks to do one of four things:

* Vandalize your computer in some way
* Steal your private information
* Take remote control of your computer (zombie your computer) for other ends
* Manipulate you into purchasing something

Malware programs are the time bombs and wicked minions of dishonest programmers. Protect yourself with a firewall and knowledge of how to prevent these programs from reaching your computer.

Trojan

A trojan is a special kind of hacker program that relies on the user to welcome it and activate it. Named after the famous Trojan horse tale, a trojan program masquerades as a legitimate file or software program.

Sometimes it is an innocent-looking movie file or an installer that pretends to be actual anti-hacker software. The power of the trojan attack comes from users naively downloading and running the trojan file.

Protect yourself by not downloading files that are sent to you in emails or that you see on unfamiliar websites.

Phishing

[Phishing](https://www.lifewire.com/what-phishing-and-email-scams-look-like-4064080) is the use of convincing-looking emails and web pages to lure you into typing your account numbers and passwords/PINs. Often in the form of fake PayPal warning messages or fake bank login screens, [phishing attacks](https://www.lifewire.com/protect-yourself-from-phishing-scams-2487728) can be convincing to anyone who is not trained to watch for the subtle clues. As a rule, smart users—beginners and long-time users alike—should distrust any email link that says "you should log in and confirm this."

Operating System (OS)

An operating system (OS), in its most general sense, is software that allows a user to run other applications on a computing device. While it is possible for a software application to interface directly with hardware, the vast majority of applications are written for an OS, which allows them to take advantage of common libraries and not worry about specific hardware details.

The operating system manages a computer's hardware resources, including:

* Input devices such as a keyboard and mouse
* Output devices such as display monitors, printers and scanners
* Network devices such as modems, routers and network connections
* Storage devices such as internal and external drives

The OS also provides services to facilitate the efficient execution and management of, and memory allocations for, any additional installed software application programs.

Some operating systems were developed in the 1950s, when computers could only execute one program at a time. Later in the decade, computers included many software programs, sometimes called libraries, which were linked together to create the beginning of today's operating systems.

The OS consists of many components and features. Which features are defined as part of the OS vary with each OS. However, the three most easily defined components are:

* Kernel: This provides basic-level control over all of the computer hardware devices. Main roles include reading data from memory and writing data to memory, processing execution orders, determining how data is received and sent by devices such as the monitor, keyboard and mouse, and determining how to interpret data received from networks.
* User Interface: This component allows interaction with the user, which may occur through graphical icons and a desktop or through a command line.
* Application Programming Interfaces: This component allows application developers to write modular code.

Examples for OSs include Android, iOS, Mac OS X, Microsoft Windows and Linux.

Virtual Device

A virtual device, in operating systems like Unix or Linux, refers to a device file that has no associated hardware. This type of file can be created with the *mknod* command, for instance. A virtual device mimics a physical hardware device when, in fact, it exists only in software form. Therefore, it makes the system believe that a particular hardware exists when it really does not. A virtual device is also known as a virtual peripheral.

Virtual devices are generally used to fix an error in the operating system. For example, a bug or virus can be detected by supposing an external device is monitoring it. Initially, the command mknod was used to produce the character and block devices that populate the "/dev/" directory. But now the udev device manager automatically creates and destroys device nodes in the virtual file system. The supposed hardware (virtual device) is detected by the kernel, but, actually, it is only a file/directory.

How Does Internet Work ? (in 5 minutes)

<https://www.youtube.com/watch?v=7_LPdttKXPc>

## What does Hub (Networking) mean?

A hub, in the context of networking, is a hardware device that relays communication data. A hub sends data packets (frames) to all devices on a network, regardless of any MAC addresses contained in the data packet.

A switch is different than a hub in that it keeps a record of all MAC addresses of all connected devices. Thus, it knows which device or system is connected to which port. When a data packet is received, the switch immediately knows which port to send it to. Unlike a hub, a 10/100 Mbps switch will allocate the full 10/100 Mbps to each of its ports, and users always have access to the maximum bandwidth – a huge advantage of a switch over a hub.

* Network Hubs: These are common connection points for network devices, which connect segments of a LAN (local area network) and may contain multiple ports – an interface for connecting network devices such as printers, storage devices, workstations and servers. A data packet arriving at one hub’s port may be copied to other ports allowing all segments of the network to have access to the data packet.
* Passive Hubs: These only serve as paths or conduits for data passing from one device, or network segment, to another.
* Intelligent Hubs: Also known as manageable hubs, these hubs allow system administrators to monitor data passing through and to configure each port, meaning to determine which devices or network segments are plugged into the port. Some ports may even be left open with no connection.
* Switching Hubs: These hubs actually read the attributes of each unit of data. The data is then forwarded to the correct or intended port.

Think of a hub as a four-way intersection where everyone has to stop. If more than one car reaches the intersection at the same time, they have to wait for their turn to proceed.

Now imagine what this would be like with a dozen or even a hundred roads intersecting at a single point. The amount of waiting and the potential for a collision increases significantly. But wouldn't it be amazing if you could take an exit ramp from any one of those roads to the road of your choosing? That is exactly what a switch does for network traffic. A switch is like a cloverleaf intersection -- each car can take an exit ramp to get to its destination without having to stop and wait for other traffic to go by.

A vital difference between a hub and a switch is that all the nodes connected to a hub share the bandwidth among themselves, while a device connected to a switch port has the full bandwidth all to itself. For example, if 10 nodes are communicating using a hub on a 10-Mbps network, then each node may only get a portion of the 10 Mbps if other nodes on the hub want to communicate as well. But with a switch, each node could possibly communicate at the full 10 Mbps. Think about our road analogy. If all of the traffic is coming to a common intersection, then each car it has to share that intersection with every other car. But a cloverleaf allows all of the traffic to continue at full speed from one road to the next.

**What does Routing mean?**

Routing refers to establishing the routes that data packets take on their way to a particular destination. This term can be applied to data traveling on the Internet, over 3G or 4G networks, or over similar networks used for telecom and other digital communications setups. Routing can also take place within proprietary networks.

In general, routing involves the network topology, or the setup of hardware, that can effectively relay data. Standard protocols help to identify the best routes for data and to ensure quality transmission. Individual pieces of hardware such as routers are referred to as "nodes" in the network. Different algorithms and protocols can be used to figure out how to best route data packets, and which nodes should be used. For example, some data packets travel according to a distance vector model that primarily uses distance as a factor, whereas others use Link-State Protocol, which involves other aspects of a "best path" for data.

Data packets are also made to give networks information. Headers on packets provide details about origin and destination. Standards for data packets allow for conventional design, which can help with future routing methodologies. As the world of digital technology evolves, routing will also evolve according to the needs and utility of a particular network.

## What does a Query String mean?

A query string is the portion of a URL where data is passed to a web application and/or back-end database. The reason we need query strings is that the HTTP protocol is stateless by design. For a website to be anything more than a brochure, you need to maintain state (store data). There are a number of ways to do this: On most web servers, you can use something like session state server-side. On the client, you can store via cookies. Or in the URL, you can store data via a query string.

On the world wide web, all URLs can be broken down into the protocol, the location of the file (or program) and the query string. The protocol you see in a browser is almost always HTTP; the location is the typical form of the hostname and filename (for example, www.techopedia.com/somefile.html), and the query string is whatever follows the question mark sign ("?").

For example, in the URL below, the bolded area is the query string that was generated when the term "database" was searched on the Techopedia website.

//www.techopedia.com/search.aspx?**q=database§ion=all**

# **Application Programming Interface (API)**

An application programming interface (API) is a set of protocols, routines, functions and/or commands that programmers use to develop software or facilitate interaction between distinct systems. APIs are available for both desktop and mobile use and are typically useful for programming GUI (graphic user interface) components, as well as allowing a software program to request and accommodate services from another program.

An API can be seen as composed of two fundamental elements: a technical specification that establishes how information can be exchanged between programs (which itself is made up of request for processing and data delivery protocols) and a software interface that somehow publishes that specification.

The basic concept behind the API has existed in some form for the entire history of digital technology, as the interaction between unique programs and digital systems has been a primary objective for much of that technology's existence. But with the rise of the world wide web, and the subsequent turn-of-the-millennium dot-com boom, the incentive for this technology reached an unprecedented level.

The API became especially prominent in the burgeoning commercial sector of the world wide web in early 2000, when Salesforce.com incorporated the technology into its platform in order to help customers share and transmit data over their diverse business applications. Soon after that, eBay began rolling out similar technology, and with the rise of social media a few years later, companies like Flickr, Facebook, Twitter and Instagram began doing the same.

# **RESTful API**

A RESTful API is an API that conforms to the representational state transfer or REST model. RESTful APIs are sometimes easier for developers to use because they have a familiar syntax and set of protocols. As more functionality has been built into the internet, developers have talked a lot about the benefits of RESTful architecture.

A RESTful architecture uses HTTP coding for much of its functionality. It uses the established Secure Sockets Layer (SSL) encryption for security purposes. It is also language-agnostic, in a practical sense, and very compatible with many different environments.

That is not to say that RESTful architecture does not have its limitations. One big example that developers have talked about extensively is the lack of state-based data transfer in RESTful architectures. As a result, applications must be stateless, or be supplemented with some outside resource that adds in the desired state information. Again, the simplicity enabled by RESTful architecture means it still enjoys some popular use in the developer community.