Open Access
Open Journal of Cloud Computing (OJCC)
Volume 3, Issue 1, 2016
www.ronpub.com/ojcc
ISSN 2199-1987

Doing More with the Dew: A New Approach to Cloud-Dew Architecture

David Edward Fisher, Shuhui Yang

Purdue University Northwest, Department of Mathematics, Statistics, and Computer Science,
2200 169th St. Hammond, IN 46323, USA, {fisherdavidedward, yangshuhui}@gmail.com

Abstract

While the popularity of cloud computing is exploding, a new network computing paradigm is just beginning. In this paper, we examine this exciting area of research known as dew computing and propose a new design of cloud-dew architecture. Instead of hosting only one dew server on a user's PC — as adopted in the current dewsite application — our design promotes the hosting of multiple dew servers instead, one for each installed domain. Our design intends to improve upon existing cloud-dew architecture by providing significantly increased freedom in dewsite development, while also automating the chore of managing dewsite content based on the user's interests and browsing habits. Other noteworthy benefits, all at no added cost to dewsite users, are briefly explored as well.

Type of Paper and Keywords

Visionary paper: *Dew computing, cloud-dew architecture, dewsite, dew analytics, dew virtual machine, DVM hypervisor, dew resource registry, dew server, artificial intelligence, evaporation*

1 Introduction

It is hardly a surprise that the cloud has grown to become a prominent element in modern network architecture. However, as the cloud continues to evolve, the precise definition of what it exactly entails continues to be debated. While the National Institute of Standards and Technology (NIST) of U.S. Department of Commerce (www.nist.gov) describes cloud computing as "*a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [16], many other definitions have also been provided [1, 6, 7]. For clarity, in our following discussion we have chosen to use a simplified definition of the cloud — Cloud as the collection of remote servers, which provide the resources necessary to support the Internet's various applications and operations.

The convenience that the cloud offers has certainly swept the world by storm. Not only does the cloud provide the user with near unlimited storage space and processing power [1], it has also redefined how we use our personal computers today. For example, millions of users have already abandoned the traditional way of storing data — that is, on their local devices — and have opted to use cloud services such as Google Drive (drive.google.com) or iCloud (www.icloud.com) instead [7, 14]. By storing their data on remote servers, users can access their photos, messages, and documents wherever and whenever they want and on whatever device they happen to be using at the time.

It seems like the cloud is the winning solution to all of our computer needs, but it is not quite perfect. For instance, it is certainly great to have the cloud, but only if one can access it. Unfortunately, by providing services at a remote location, the cloud introduces a single-point-of-failure in that a stable Internet connection is absolutely required to interact with it [1, 6, 7]. If a cloud-dependent user happens to have her/his connection to the Internet disrupted or if cloud servers go offline (as has happened in the past [1, 7]), the user's PC essentially becomes crippled and data-starved. It must be noted that even if a connection to the cloud is available, network latency must always be considered in cloud operations and any human-perceivable latency is often considered a significant inconvenience to cloud users [18]. Due to the above concerns, rushing toward a future where PCs have a total dependence on the cloud is unadvised.

On the other hand, personal computers are becoming increasingly faster, more powerful, and cheaper as we delve deeper into the digital era. However, the prevalence of the cloud is diminishing the impact of such progress. Nowadays it is common for the cloud to perform more work than a user's own device. As a result, a new trend is emerging where PCs frequently find themselves waiting idle with a significant portion of their storage space going unused [15, 17, 18]. While many see this as a process of technological evolution, we view it as an opportunity for innovation. In particular, we believe a newly conceived computing paradigm, known as "dew computing", appears exceptionally promising.

In this paper, we will discuss the definition of dew computing, examine the current design of the dewsite application, and propose an improved cloud-dew architecture. Detailed implementation issues will also be analyzed. We expect this work to shed some insights on the future development of dew computing.

The remainder of this paper is organized as follows. Section 2 begins by providing a brief overview of dew computing and the dewsite application, followed by a short description of our concerns and proposed improvements to cloud-dew architecture. Section 3 discusses each component of the proposed architecture in detail. Finally, Section 4 features a brief summary.

2 Origins of Dew Computing

The concept of dew computing was first proposed in 2015 [28]. At the time, fog computing was just beginning to gain traction among computer scientists. Although the details of fog computing are outside the scope of this paper, the idea was certainly revolutionary — Fog computing reduces latency in time-critical applications by bringing cloud services physically closer to devices on a network [2]. Unfortunately, fog computing is intended to appeal to a computationally-weak and fully automated audience, such as sensors and Internet of Things (IoT) devices, and not humans casually using their PCs [21, 31]. The dew computing paradigm was devised in order to address this issue and extend the cloud metaphor even further. While the cloud is "up in the sky" far away from our computers and the fog is "hovering just above the ground" near our IoT devices, the dew is "on the ground" and actually part of our PCs [20].

Of course, this begs the question of "What does dew computing actually mean?" Many researchers have attempted to answer such question [20, 28, 29]. However, a consensus has not yet been reached. As such, we provide the initially proposed definition, followed by our interpretation.

> *"Dew Computing is a personal computer software organization paradigm in the age of Cloud Computing. Its goal is to fully realize the potentials of personal computers and cloud services. In this paradigm, software on a personal computer is organized according to the Cloud-dew Architecture; in this paradigm, a local computer provides rich functionality independent of cloud services and also collaborates with cloud services."* [30]

Although the definition is somewhat vague, we interpret it as introducing a tighter coupling between the cloud and a user's PC in order to provide some form of additional functionality. Rather than being two disjoint systems, we expect dew computing to blur the line between cloud and PC, drastically increasing their collaboration with one another.

2.1 The Dewsite Application

A novel application of dew computing has been proposed by Wang in [28]. The main idea of this application is creating locally a special form of web server, a "dew server", by using a portion of a PC's memory. This dew server would essentially host scaled-down variants of websites, full of pre-downloaded content, which the user could access without requiring an Internet connection. The idea of a "cached" Internet is not new and RSS technology [27] has been using it for years. However, Wang takes it a step further by describing that a website stored on a user's dew server, what he dubs as a "dewsite", would be much more interactive and contain a built-in mechanism to automatically synchronize with the cloud [28]. With this dewsite concept, for example, a user could browse Facebook, upload photos, and even post messages all without an Internet connection. Once the user reobtains a connection to the Internet, any actions or changes the user made on the dewsite would be synchronized with the cloud. When the user returns to the Facebook website, the user would find his/her profile updated accordingly.

Indeed, the dewsite application appears promising — not only does it provide a significant incentive for users by allowing offline web browsing while eliminating network latency, but it also appeals to developers since it can be implemented without any additional hardware. In fact, dew servers could be implemented entirely via software using excess, unused memory on a PC.

2.2 Areas of Improvement

The proposed dewsite application is certainly revolutionary, but we believe it can be explored further. We took note that dew servers make excellent use of excess memory that PCs have, but we also noticed that their current definition does not account for utilizing unused CPU cycles. Modern PCs are known for their powerful CPUs in addition to their large hard drives and RAM, after all. In order to maximize PC efficiency, we need to find a way to keep the CPU busy with some kind of meaningful tasks. Such tasks not only need to have purpose, but also must be able to operate in an Internet-deficient environment as well. The current dewsite application as defined today fails to meet this demand, but we expect there to be a way for the dew to harness the power of an idle CPU. Note that when we mention "the dew", we are implying the entire dew computing aspect of a PC as a whole.

Furthermore, in [28], it is stated that "*a reasonable design is to run only one dew server on the local computer and the dew server will provide all services of these dewsites*". However, we do not completely agree with this being the best approach. Websites around the world are built on various platforms, in various operating environments, using various database management systems. Wang's proposed solution is to define a set of platforms and database management systems that are "dew-capable" and instruct dewsite developers to use only those technologies [28], but this may complicate things and hinder progress in dew development. For instance, it could cost several thousands of dollars for a website to completely revamp what it was originally built with in order to be hosted on a dew server. Instead, we expect there to be a way for developers to use whatever technology they deem fit, yet still cohabitate in a single, cohesive dew entity. Indeed this will be a challenge, but the possibility must be explored further.

2.3 The Proposed Improvements

Now that we have addressed a few of our concerns we have with the current design of the dewsite application, we provide our ideas on how to solve them.

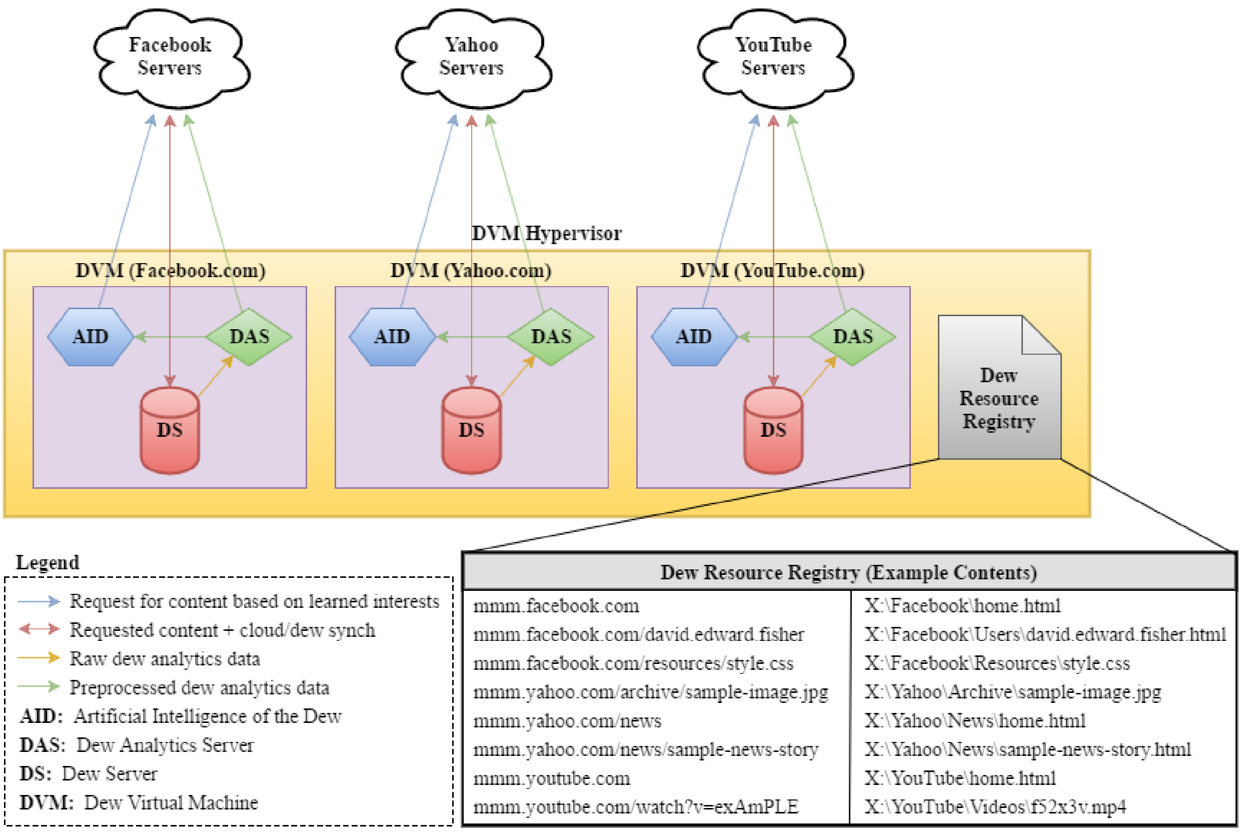
- In order to meet the demand of idle-CPU utilization, we propose a modified web analytics system that is specifically crafted to operate on data generated from activity on dewsites. Big Data is a massive market nowadays [9], and we believe that there is a significant way to involve the dew in Big Data analysis. Not only could the dew aid in performing analytics, but it could also learn from it. Imagine the dew dynamically downloading new content based on the user's interests that it has learned — we believe this is possible.

- In order to provide more freedom in dewsite development, we suggest scrapping the idea of hosting only one dew server and instead provide each domain stored on the dew with its own isolated environment. This could be accomplished by adding another layer, a virtual machine layer, to the cloud-dew architecture proposed in [28].

3 THE PROPOSED CLOUD-DEW ARCHITECTURE

In this section, we discuss our design of cloud-dew architecture. For convenience and readability, we have separated each component into its own subsection. Please note that the proposed design does not require any additional hardware components. In fact, the design is intended to be applicable to any average PC, entirely via software.

A visualization of the proposed cloud-dew architecture as a whole is demonstrated in Figure 1. Notice that each domain is isolated from the others via its dedicated DVM. Each DVM itself contains three components — a dew server, a dew analytics server, and an AID — which collaborate with one another to provide automated dewsite services to the user. Although each DVM operates independently of the others, the DVM hypervisor and dew resource registry provide coordination among installed domains such that the entire system functions as a cohesive entity. The detailed design of each component, as well as their interoperability with one another, will be discussed in the following.



Dew Resource Registry (Example Contents)	
mmm.facebook.com	X:\Facebook\home.html
mmm.facebook.com/david.edward.fisher	X:\Facebook\Users\david.edward.fisher.html
mmm.facebook.com/resources/style.css	X:\Facebook\Resources\style.css
mmm.yahoo.com/archive/sample-image.jpg	X:\Yahoo\Archive\sample-image.jpg
mmm.yahoo.com/news	X:\Yahoo\News\home.html
mmm.yahoo.com/news/sample-news-story	X:\Yahoo\News\sample-news-story.html
mmm.youtube.com	X:\YouTube\home.html
mmm.youtube.com/watch?v=exAmPLE	X:\YouTube\Videos\f52x3v.mp4

Figure 1: The Proposed Cloud-Dew Architecture

3.1 Dew Virtual Machines (DVMs)

As we mentioned in Section 2.2, the original cloud-dew architecture was built on the idea that only one dew server would exist and all downloaded domains would cohabitate within it. In order to allow such cohabitation, dewsites would have to be created using only a small set of "dew-capable" technologies. We also mentioned that a better implementation of cloud-dew architecture is expected to alleviate the significant limitations and added burden on dewsite development.

Instead of having one giant dew server containing every downloaded dewsite on the user's PC, we propose the idea of having multiple small dew servers instead. Not only would this greatly simplify management of the dew by breaking it into smaller, more manageable chunks, it would also allow each domain to have its own dew server built on whatever technologies it deems fit. In other words, the ability to support multiple dew servers running concurrently would mean that a new dew server could be created and designated to each new domain stored on a user's PC. As a result, each dew server could be governed by its respective domain without requiring cooperation from any other domain stored on the machine. For example, if two domains, say Facebook.com and YouTube.com, exist on a user's PC, then there would be two dew servers — one for Facebook.com and one for YouTube.com. In this scenario, if Facebook prefers to implement MySQL as its DBMS, but YouTube prefers to implement Oracle, there would be no issue whatsoever since neither one would require interaction with the other.

In order to support hosting multiple dew servers on a single PC, we believe that dew servers could be operated in designated virtual machines — what we have dubbed "dew virtual machines" (DVMs). Just as standard virtual machines provide an isolated environment for various applications to be executed [17], DVMs would allow dew servers to be hosted in isolated environments as well. Not only would this setup allow for implementation independence amongst the various installed domains, it would also drastically simplify organization and management of the dew in general.

3.1.1 An Approach to Retaining a Fixed-Size DVM: Evaporation

It must be noted that as dew servers pre-download more and more content as time goes on, logic dictates that they will eventually run out of available storage space within their DVMs. In order to solve this issue, we recommend gradually removing the least desirable content in a process we like to call “evaporation”. As a homage to the cloud/fog/dew metaphor, dew evaporation could essentially serve as the primary mechanism to retain a fixed-size DVM in a setting where new content gets added daily.

Perhaps this is best illustrated with a simple example. Imagine a user who has a Yahoo dew server installed on his or her PC, with 1GB of available storage remaining in the Yahoo DVM. Every morning at 6:00 a.m., the latest news articles and images, approximately 100MB of data, are fetched from Yahoo’s remote web server. With some simple math, we can foresee that the DVM will be at its capacity within two weeks if we permanently store all the data — herein lies the problem. However, evaporation could provide a solution to this issue, for example, by removing any content older than a week. If we utilize the one-week evaporation concept in the above example, there would never be more than around 700MB of content at any given time.

Note the similarity between evaporation and the removal process in cache replacement [3]. Caches have a fixed-size memory in much the same way that DVMs do. Consequently, we expect we can adopt and/or extend cache replacement algorithms to suit evaporation as well. It is no coincidence that the above example uses a very primitive time-based algorithm comparable to the LRU algorithm commonly used in replacing cache content [8]. In fact, we believe that other, more complex cache replacement algorithms could also be used when deciding what dewsite content should be periodically deleted. One such algorithm that appears particularly promising was developed by Jarukasemratana and Murata [8], which incorporates web usage mining into the replacement decision.

One might wonder if periodic deletion of old content would trouble users, but we believe that the majority of users will not mind given they have the newest content available. It must be noted that PCs simply do not have the vast storage capacity that the cloud offers, after all. For users who find evaporation undesirable, we recommend a mechanism to flag content to keep permanently. Some dewsites may not be suitable for evaporation at all, but we offer our solution nonetheless. In the end, the majority of dewsites will find it necessary to use evaporation at some point or another in order to continue operating in a fixed-size DVM.

3.2 Dew Virtual Machine (DVM) Hypervisor

As with any architecture involving virtual machines, there must be a component to create, delete, and otherwise manage them [17]. Our system is no exception, thus we introduce the DVM hypervisor. The DVM hypervisor will be primarily responsible for:

- Providing a method of sending dew requests to domains (i.e., asking the domains to install their dew services on the user’s PC);
- Creating new DVMs for domains when dew requests are accepted;
- Managing DVM operation, such that all DVMs can run concurrently without issue; and
- Providing a method of removing or uninstalling unwanted domains.

3.2.1 DVM Instantiation Process

We envision the following steps to take place when installing a new domain (also illustrated in Figure 2):

1) When the user decides he/she wants to install a new domain in his/her dew, he/she accesses a function of the DVM hypervisor, which prompts the user to input a domain name.
2) Once the prompt appears, the user inputs the domain name he/she wishes to install (e.g., facebook.com).
3) After the user submits his/her desired domain name, the DVM hypervisor checks to see if it is already installed on the user’s PC.
4) If the domain does not exist on the user’s PC, the DVM hypervisor sends a request to the domain’s remote (cloud) server and asks it to install its dew services on the user’s PC.
5) If the domain agrees to the request, the DVM hypervisor creates a new (empty) dew virtual machine and labels it with the domain name (e.g., facebook.com).
6) Once the new DVM is created, the domain is given control of it.
7) The new DVM is now dedicated solely to the new domain, so the domain installs all components of its dew system (e.g., operating system, database management system, other dew server files, dew analytics server, AID, etc.) inside its DVM.
8) Once installation is complete, the domain registers all available URLs and their associated file locations with the dew resource registry.

After successfully completing the above steps, the user will be able to access the new domain’s dewsite.

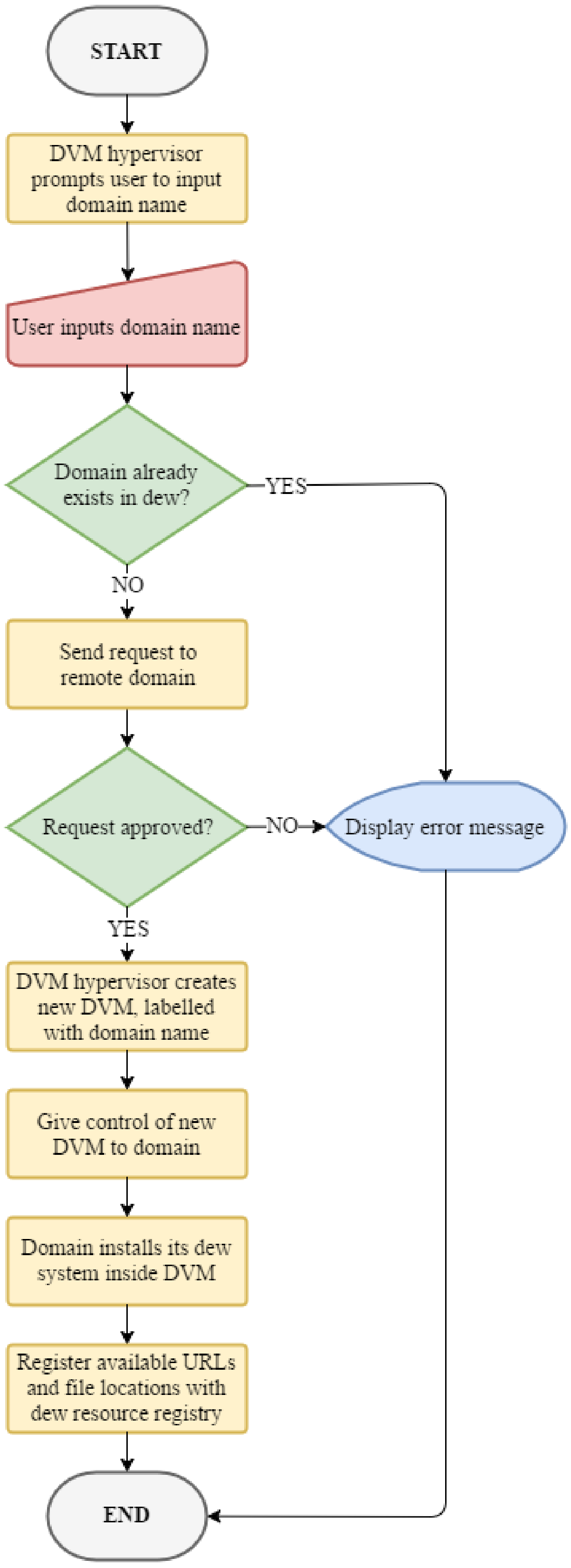

START
DVM hypervisor prompts user to input domain name
User inputs domain name
Domain already exists in dew?
YES
NO
Send request to remote domain
Request approved?
NO
Display error message
YES
DVM hypervisor creates new DVM, labelled with domain name
Give control of new DVM to domain
Domain installs its dew system inside DVM
Register available URLs and file locations with dew resource registry
END

Figure 2: DVM Instantiation Flow

3.3 Dew Resource Registry (DRR)

With potentially thousands of files located throughout numerous dew servers stored on a user's PC, it is essential to create a dedicated component of the dew tasked solely with keeping track of what is available and where to find it. The "dew resource registry" (DRR) is the component responsible for mapping URLs to their associated file locations in much the same way that DNS servers map URLs to their associated IP addresses.

We envision that after a new dew server has been installed on a user's PC, it should be required to communicate with the DRR and specify what URLs are able to be accessed, what file is associated with each URL, and where to find that file on the user's PC. There are two particular benefits to this approach: users, or more specifically browsers, will know what content is available for access and where to quickly find it; and any irrelevant or sensitive files (such as dew server installation files) can be hidden from users via omission from the registry process. The DRR could be implemented as a simple table (such as the example in Figure 1), but tree data structure variants (e.g., B-Trees) and customized hash tables could certainly be applicable as well due to their increased searching/retrieval speed. We leave such topic open for debate.

3.4 Dew Servers

We have frequently mentioned dew servers thus far; however, we have yet to adequately define them outside of Section 2.1. In essence, a dew server is a web server hosted locally but with three important differences. First of all, while web servers are designed to serve numerous clients concurrently, dew servers are intended to serve only one client — the hosting PC itself. Secondly, dew servers have an added responsibility in that they must periodically synchronize with the cloud, and therefore must also handle out-of-synch-related issues. Finally, web servers always have to deal with network-related issues, but dew servers only have to interact with the network during synchronization, not all the time. Other than the above differences, dew servers and web servers are surprisingly similar — both store various content and both must provision content to users when it is requested.

3.4.1 An Approach for Accessing Dewsite Files as Host: Shared Folders

Since our design involves hosting dew servers inside of virtual machines, accessing dewsite files from the host becomes a tad more complicated. Thankfully this topic has been extensively debated by the virtual machine community and one solution already exists — shared folders. A shared folder is a dedicated directory that can be accessed and modified by both the host *and* virtual machines running on the same machine [4, 22]. If a dew server were to place files (in particular, files that are intended for access by the user) into a shared folder, a browser running on the host OS could easily retrieve them without issue. Note that solutions other than shared folders may also exist.

3.4.2 An Approach to Preserving Proprietary Technology: Encryption in Key Areas

When a domain installs its dew system on a user's PC, we expect the domain will have some amount of proprietary technology it will not want to openly disclose — that is simply the business of IT. In an effort to protect the intellectual property of domain owners, we believe that DVMs could support partial encryption, such that areas of concern (for example, proprietary scripts) could be encrypted while the remainder of DVM content remains freely accessible. Encryption would have to be used with extreme caution, however, since it is likely that the majority of dew users will not want a remote entity (e.g., Facebook) using their PCs in unknown ways. A middle-ground of sorts would have to be found, such that intellectual property is adequately protected while dew users still feel comfortable entrusting a portion of their PCs to an outside source. Politics aside, we believe that encryption could have a significant role to play in dew development.

3.4.3 An Approach to Navigating to Dewsite URLs: "mmm" Indicator

As we mentioned in Section 2.1, the term "dewsite" refers to a website hosted locally. In [28], the author describes an important issue regarding how to differentiate a dewsite from its website counterpart. In essence, he asks his readers to consider: if a user has Facebook installed in his/her dew and enters "facebook.com" into his/her browser, how will the browser know if it should retrieve Facebook's website or Facebook's dewsite? The author's solution was simple, yet extremely effective — by using a small indicator attached to the beginning of the URL — "www" indicates the website version while "mmm" indicates the dewsite version. We believe this is a fantastic approach and have chosen to use this idea in our cloud-dew design as well. For example, if a user enters "www.facebook.com" into his/her browser's navigation bar, the browser should load Facebook's website; if a user enters "mmm.facebook.com", the browser should load Facebook's dewsite instead; if a user enters only "facebook.com", the browser could be configured to load either one by default, perhaps depending on if an Internet connection is available or not. All in all, we believe it is best to designate the browser as the entity in charge of examining given URLs and choosing what to load accordingly.

Given a URL, determining where a dewsite file is located is surprisingly similar to determining which IP address to contact. When given a URL corresponding to a website (i.e., beginning with "www"), the browser must refer to a DNS server and discover the proper IP address before loading the page. Comparatively, when given a URL corresponding to a dewsite (i.e., beginning with "mmm"), the browser must refer to the dew resource registry and discover the proper file location before loading the page. Figure 3 features a side-by-side comparison of these two processes.

3.5 Dew Analytics Servers

Few can deny the notion that an idle CPU is wasted potential, so we decided to explore an application where the dew could utilize unused cycles for the benefit of both the user and the cloud. What we ended up conceiving is a new component we have dubbed a "dew analytics server", which is intended to preprocess various forms of analytic data. Analogous to web analytics servers, dew analytics servers are a very similar idea. While web analytics servers operate on data generated when users browse and interact with *websites*, dew analytics servers operate on data generated when users browse and interact with *dewsites*. Although dew analytics and web analytics are more alike than different, there are three significant differences between them:

1) Dew analytics servers receive data generated from a single user; web analytics servers receive data generated from any number of users.

2) Dew analytics data is preprocessed *before* it leaves the user's PC and is sent to the cloud; web analytics data is sent in raw form.

3) In addition to being used by the cloud, dew analytics data is also used by the dew itself in order to better serve the user; web analytics data is used by the cloud only.

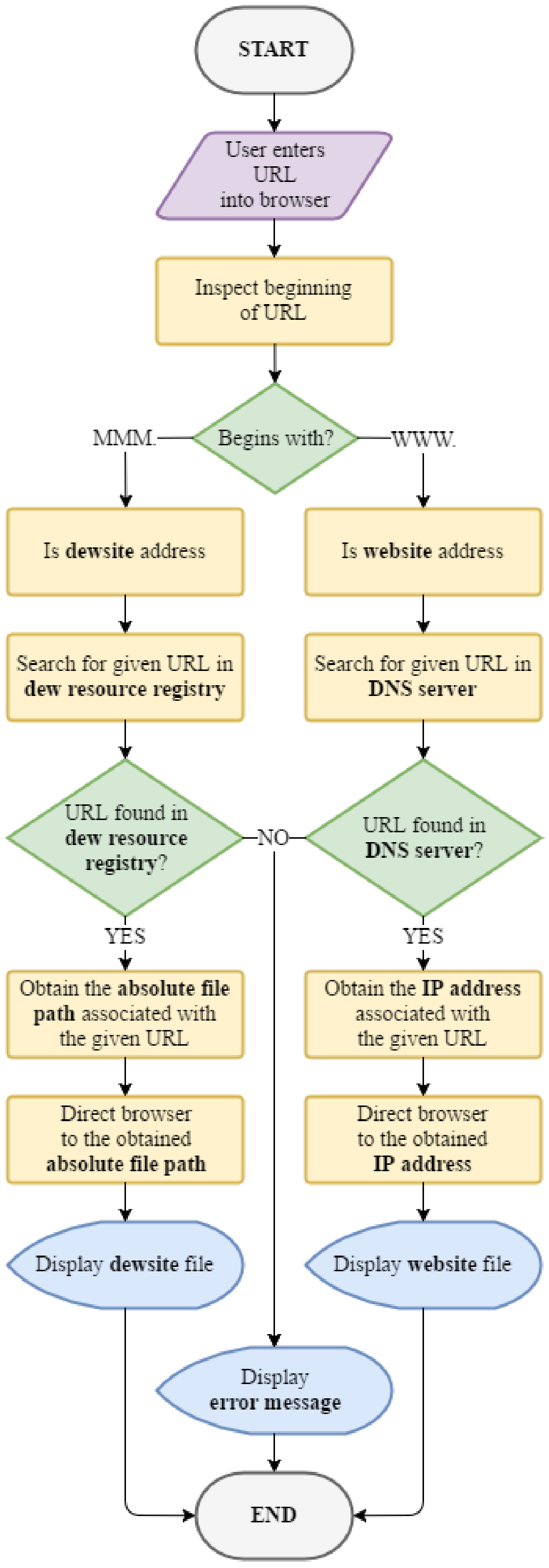

START
User enters URL into browser
Inspect beginning of URL
Begins with?
MMM.
WWW.
Is dewsite address
Is website address
Search for given URL in dew resource registry
Search for given URL in DNS server
URL found in dew resource registry?
URL found in DNS server?
NO
YES
YES
Obtain the absolute file path associated with the given URL
Obtain the IP address associated with the given URL
Direct browser to the obtained absolute file path
Direct browser to the obtained IP address
Display dewsite file
Display website file
Display error message
END

Figure 3: Dew Resource Registry and DNS Server Comparison

3.5.1 Motivation

The motivation for creating dew analytics is as follows:

- PCs frequently find themselves waiting idle [15, 17]. As a result, CPU cycles are essentially wasted by going unused.
- We believe developers will want to implement an analytics system into dewsites for the same reason they implement analytics systems into websites — to learn about their users and enhance their sites to best suit the users' needs.
- Since an Internet connection is not guaranteed to be always available, we cannot assume that raw analytic data can immediately be sent to a remote server, as is done with web analytics [13]. Instead, analytic data generated from dewsites must remain local, where it will continue to "pile up" as time goes on, until the next cloud synchronization occurs.
- Data suitable for analysis often originates as unstructured, and is therefore difficult to work with [9]. In order to be considered usable, the data must first be preprocessed and converted into a structured format; however, this preprocessing step takes considerable time and computational effort [5, 10, 11].

We expect dew analytics servers to effectively utilize unused CPU cycles by preprocessing the user's own analytic data before sending it to the cloud. Please note that we acknowledge that the cloud's immense computational power could quickly preprocess a single user's data. However, if millions of dew PCs preprocessed their analytic data before sending it to the cloud, the cloud's time and processing savings would be extraordinary.

3.5.2 A Briefing on Unstructured Data and the Need to Structure It

For completeness, we briefly review the concept of structured versus unstructured data in the following. Structured data is data which generally resides in databases consisting of a number of columns and rows, where it is grouped into relations or classes based on shared characteristics [19]. Each piece of data typically has one or more associated attributes which allow the data set to be easily queried, sorted, and otherwise manipulated in various ways. The attributes themselves are most commonly composed in a predefined format such as an integer or a fixed-length string [19].

Unstructured data is pretty much everything else. Daniel Senter provides the following definition:

"*Unstructured data is a generic term used to describe data that doesn't sit in databases and is a mixture of textual and non-textual data. Unstructured non-textual data generally relates to media such as images, video and audio files… Slightly less unwieldy are unstructured textual data made up of media files (documents, spreadsheets, presentations), email messages and an array of other [text-based] files.*" [19]

According to IBM, more than eighty percent of all information is unstructured [5]. In fact, the amount of unstructured data is growing so rapidly that companies such as Facebook and Twitter are absolutely drowning in it [10, 11, 19]. Unstructured data is far from useless, but it provides a significant hurdle to analytics systems as the vast majority of them cannot directly work with it [10, 23]. In order to be deemed usable (i.e., analyzable), unstructured data must be put through an extra step in the analysis process — conversion to a structured format [5, 10, 11, 23]. Dew analytics servers are the perfect candidates to perform that step.

Dew analytics servers are not limited to operating only on unstructured data either. Structured data can also be preprocessed, albeit in a more trivial manner. For instance, we expect that other sources of analytic data — such as data generated from page tagging and server logs [13] — will also be prime candidates for valuable information about the user. Dew analytics servers are designed to operate on a wide variety of data types as a result.

3.5.3 A Note on Privacy in Dew Analytics Data

Since dew analytics servers will likely come across sensitive user information such as browsing activity, messages, photos, and other multimedia content, preserving user privacy is of the utmost importance. One method of preserving privacy is to have dew analytics servers remove all personally-identifiable information from analytic data before sending it to the cloud [12]. This way, users can feel comfortable transmitting their data since it will be entirely anonymized. Of course, some users may still feel uncomfortable with anonymous data, therefore domain owners should give users the option to opt out of sending their analytic data to the cloud entirely [12]. In this case, only AID (will be discussed in Section 3.6) will receive the data and it will remain entirely local on the user's PC.

3.6 Artificial Intelligence of the Dew (AID)

A keen reader may note a significant issue with dew analytics servers as defined so far — that they provide a significant benefit for the cloud, but not for the users who own the machines they run on. Why would users want the cloud to use their PCs for something that does not benefit them, after all? We answer such a question by introducing the "artificial intelligence of the dew" (AID). Instead of sending the user's preprocessed dew analytics data only to the cloud, we can directly benefit the user by also sending a copy to AID. Just as its acronym implies, AID's purpose is to aid the user by using data gathered from dew analytics to learn about the user's habits/preferences and to pre-download web content *before* the user needs it.

AID would be a powerful feature that would make the dew appealing to all PC users, even novice individuals. By automating the "chore" of updating and managing dewsite content, owning a dew computer would be entirely hassle-free. AID would also promote more efficient utilization of DVM memory since dewsite content would be much more relevant to the user. In fact, we expect AID and the evaporation process introduced in Section 3.1.1 to form an effective content-management team, such that AID facilitates the intelligent *gathering* of new dewsite content while evaporation facilitates the intelligent *removal* of old dewsite content. To take things a step further, if we establish a communication channel between AID and the evaporation process, the two could actually learn from each other's actions, further improving the relevancy of content stored in a domain's dew server. This collaboration will ultimately allow for the best user experience while occupying the least amount of memory.

Please note that AID cannot simply be "one-size-fits-all"; we expect that each domain will want to design AID to best match the content of their service. For example, social media sites such as Facebook might focus on learning which friends the user most frequently interacts with, while video hosting sites such as YouTube might focus on learning which genre of videos the user prefers to watch. Regardless of what kind of data AID chooses to learn from, it will always have the same goal in mind — to dynamically learn what kinds of content each user desires and to pre-download such content before the user actually requests it him/herself. We imagine that AID will be based on one or more of the many versions of machine learning [26], such as artificial neural networks (ANNs) [24] or deep learning [25]; however, due to the wide variety of potential dewsite services, we leave the actual implementation details for domain developers to decide themselves.

3.6.1. A Mutually-Beneficial Relationship

Now that we have introduced AID, we hope that we have made it evident that dew analytics servers will be just as beneficial to the user as they are to the cloud. Preprocessing analytic data is no small task, but it is an excellent way to take advantage of CPU cycles that would have otherwise gone unused. If dew analytics servers, AID, and evaporation were to be successfully implemented, we believe the following benefits could be a reality — all at no extra cost to the user *or* the cloud.

The *benefits for the user* are:

- Fully automated dewsite content management.
- Personalized content selection tailored to individual browsing habits and preferences.
- Enhanced privacy via the removal of personally-identifiable information from dew analytics data.

The *benefits for the cloud* are:

- Free, yet incredibly valuable, structured dew analytics data.
- Significant time and computational savings via delegating the preprocessing step to end users.

4 Conclusion

Dew computing is a powerful new network paradigm that provides elegant solutions to common issues with the cloud. Not only does it drastically increase accessibility of user data, it also significantly reduces latency when browsing the web. However, we noticed that the current state of cloud-dew architecture has two significant drawbacks: it does not allow for much freedom in dewsite development; and it does not take advantage of an idle CPU. We addressed these issues by proposing a new version of cloud-dew architecture, one that gives domains ultimate development freedom (via dedicated DVMs coordinated by the DVM hypervisor and dew resource registry) and utilizes unused CPU cycles to automate the process of managing dewsite content (via dew analytics servers, AID, and evaporation). Other noteworthy benefits, such as a simplified user experience and cloud time/computational savings, were briefly discussed as well.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “A View of Cloud Computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog Computing and Its Role in the Internet of Things,” in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, pp. 13–16, 2012

[3] B. D. Davison, “A Web Caching Primer,” *IEEE Internet Computing*, vol. 5, no. 4, pp. 38–45, 2001.

[4] A. Everard, “VirtualBox/SharedFolders,” *Ubuntu Documentation*, 26-Sep-2014. [Online]: https://help.ubuntu.com/community/virtualbox/sharedfolders. [Accessed: 04-Jun-2016].

[5] T. Franklin, “The State of Content Analytics,” *EContent*, vol. 38, no. 1, pp. 26–27, 2015.

[6] R. Goldsborough, “Computing in the Cloud,” *Tech Directions*, vol. 70, no. 5, p. 14, Dec. 2010.

[7] Y. Han, “On the Clouds: A New Way of Computing,” *Information Technology & Libraries*, vol. 29, no. 2, pp. 87–92, Jun. 2010.

[8] S. Jarukasemratana and M. Tsuyoshi, “Web Caching Replacement Algorithm Based on Web Usage Data,” *New Generation Computing*, vol. 31, no. 4, pp. 311–329, Oct. 2013.

[9] J. Lamont, “Big data: expediting and validating analyses,” *KM World*, vol. 25, no. 1, pp. 30–32, Jan. 2016.

[10] J. Lamont, “Delving into customer thoughts: Text Analytics Provides Insights,” *KM World*, vol. 23, no. 7, pp. 12–20, 2014.

[11] J. Lamont, “Text analytics: versatile and growing,” *KM World*, vol. 22, no. 7, pp. 8–22, 2013.

[12] K. Marek, “Chapter 1: Web Analytics Overview,” *Library Technology Reports*, vol. 47, no. 5, pp. 5–10, Jul. 2011.

[13] K. Marek, “Chapter 2: Getting to Know Web Analytics,” *Library Technology Reports*, vol. 47, no. 5, pp. 11–16, Jul. 2011.

[14] D. Mohammed, "Security in Cloud Computing: An Analysis of Key Drivers and Constraints," *Information Security Journal: A Global Perspective*, vol. 20, no. 3, pp. 123–127, 2011.

[15] G. Newsham and D. Tiller, "The Energy Consumption of Desktop Computers: Measurement and Savings Potential," *IEEE Transactions on Industry Applications*, vol. 30, no. 4, pp. 1065–1072, 1994.

[16] NIST, "Cloud Computing," *NIST*, 15-Nov-2010. [Online]: http://www.nist.gov/itl/cloud/index.cfm. [Accessed: 04-Jun-2016].

[17] M. Pearce, S. Zeadally, and R. Hunt, "Virtualization: Issues, Security Threats, and Solutions," *ACM Computing Surveys*, vol. 45, no. 2, pp. 17–39, Feb. 2013.

[18] N. K. Sehgal, S. Sohoni, X. Ying, D. Fritz, W. Mulia, and J. M. Acken, "A Cross Section of the Issues and Research Activities Related to Both Information Security and Cloud Computing," *IETE Technical Review (Medknow Publications & Media Pvt. Ltd.)*, vol. 28, no. 4, pp. 279–291, 2011.

[19] D. Senter, "Transforming Unstructured into Structured Data," *Process Excellence Network*, July 15, 2012. [Online]: http://www.processexcellencenetwork.com/innovation/columns/transforming-unstructured-into-structured-data/. [Accessed: 25-May-2016].

[20] K. Skala, D. Davidović, E. Afgan, I. Sović, and Z. Šojat, "Scalable Distributed Computing Hierarchy: Cloud, Fog and Dew Computing," *Open Journal of Cloud Computing (OJCC)*, vol. 2, no. 1, pp. 16–24, 2015. [Online]: http://www.ronpub.com/publications/ojcc/OJCC_2015v2i1n03_Skala.html

[21] I. Stojmenovic and S. Wen, "The Fog Computing Paradigm: Scenarios and Security Issues," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, pp. 1–8, 2014

[22] VMware, "Using Shared Folders," *VMware*, 2016. [Online]. https://www.vmware.com/support/ws5/doc/ws_running_shared_folders.html. [Accessed: 04-Jun-2016].

[23] G. Vohra, "Analytics on Unstructured data – Twitter, Facebook and Social Media," *Data Science Central*, 18-Oct-2012. [Online]: http://www.datasciencecentral.com/profiles/blogs/analytics-on-unstructured-data-twitter-facebook-and-social-media. [Accessed: 25-May-2016].

[24] Wikipedia, "Artificial neural network," *Wikipedia*. [Online]: https://en.wikipedia.org/wiki/artificial_neural_network. [Accessed: 04-Jun-2016].

[25] Wikipedia, "Deep learning," *Wikipedia*. [Online]: https://en.wikipedia.org/wiki/Deep_learning. [Accessed: 04-Jun-2016].

[26] Wikipedia, "Machine learning," *Wikipedia*. [Online]: https://en.wikipedia.org/wiki/Machine_learning. [Accessed: 04-Jun-2016].

[27] Wikipedia, "RSS," *Wikipedia*. [Online]: https://en.wikipedia.org/wiki/RSS. [Accessed: 01-Jun-2016].

[28] Y. Wang, "Cloud-dew architecture," *International Journal of Cloud Computing*, vol. 4, no. 3, pp. 199–210, 2015.

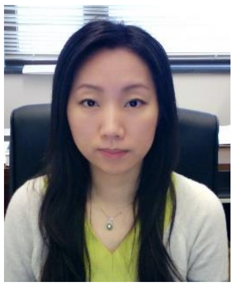
[29] Y. Wang, "Definition and Categorization of Dew Computing," *Open Journal of Cloud Computing*, vol. 3, no. 1, pp. 1–7, 2016. [Online]: http://www.ronpub.com/publications/ojcc/OJCC_2016v3i1n02_YingweiWang.html

[30] Y. Wang, "The Initial Definition of Dew Computing," *Dew Computing Research*, 10-Nov-2015. [Online]: http://www.dewcomputing.org/index.php/2015/11/10/the-initial-definition-of-dew-computing/. [Accessed: 04-Jun-2016].

[31] S. Yi, C. Li, and Q. Li, "A Survey of Fog Computing: Concepts, Applications and Issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*, pp. 37–42, 2015.

AUTHOR BIOGRAPHIES

David Edward Fisher received his BS degree in computer science from Purdue University Northwest in May 2016. His interest in dew computing first emerged during the final semester of his undergraduate career, when he was searching for a topic for his senior design project. For several months, he worked closely with his project mentor, Dr. Yang, and this paper is a result of their ongoing research into cloud-dew architecture. Other research interests of his include information security, networking, communication, and system design. David recently accepted a software engineer position at Cisco Systems, Inc. and is currently enjoying work at the company's San José, California headquarters.



Dr. Shuhui Yang received her BS and MS degrees in 2000 and 2003, respectively, from Jiangsu University, Zhenjiang and Nanjing University, Nanjing, China, and her PhD degree in computer science from Florida Atlantic University in 2007. She is an associate professor in the Department of Mathematics, Statistics, and Computer Science at Purdue University Northwest. Her current research focuses on the design of localized routing algorithms in wireless ad hoc and sensor networks and distributed systems. She is the guest editor for the EURASIP Journal on Wireless Communications and Networking, Special issue on Wireless Network Security. She serves as program committee member for many conferences, such as IEEE INFOCOM and IEEE MASS. Professor Yang received the US National Science Foundation EAGER grant and REU grant in 2009 and 2015, respectively. She is also a member of the IEEE and the IEEE Computer Society.