A Survey on Software-Defined Networking

Wenfeng Xia, Yonggang Wen, *Senior Member, IEEE*, Chuan Heng Foh, *Senior Member, IEEE*, Dusit Niyato, *Member, IEEE*, and Haiyong Xie, *Member, IEEE*

***Abstract*—Emerging mega-trends (e.g., mobile, social, cloud, and big data) in information and communication technologies (ICT) are commanding new challenges to future Internet, for which ubiquitous accessibility, high bandwidth, and dynamic management are crucial. However, traditional approaches based on manual configuration of proprietary devices are cumbersome and error-prone, and they cannot fully utilize the capability of physical network infrastructure. Recently, software-defined networking (SDN) has been touted as one of the most promising solutions for future Internet. SDN is characterized by its two distinguished features, including decoupling the control plane from the data plane and providing programmability for network application development. As a result, SDN is positioned to provide more efficient configuration, better performance, and higher flexibility to accommodate innovative network designs. This paper surveys latest developments in this active research area of SDN. We first present a generally accepted definition for SDN with the aforementioned two characteristic features and potential benefits of SDN. We then dwell on its three-layer architecture, including an infrastructure layer, a control layer, and an application layer, and substantiate each layer with existing research efforts and its related research areas. We follow that with an overview of the de facto SDN implementation (i.e., OpenFlow). Finally, we conclude this survey paper with some suggested open research challenges.**

***Index Terms*—Software-defined networking, SDN, network virtualization, OpenFlow.**

Manuscript received May 31, 2013; revised December 7, 2013 and March 19, 2014; accepted May 15, 2014. Date of publication June 13, 2014; date of current version March 13, 2015. The work of H. Xie was supported in part by the National Natural Science Foundation of China under Grant 61073192, by the Grand Fundamental Research Program of China (973 Program) under Grant 2011CB302905, by the New Century Excellent Talents Program, and by the Fundamental Research Funds for Central Universities under Grant WK0110000014. The work of Y. Wen was supported in part by NTU under a Start-Up Grant, by the Singapore MOE under MOE Tier-1 Grant (RG 31/11), by Singapore EMA under a EIRP02 Grant, and by the Singapore National Research Foundation under its IDM Futures Funding Initiative and administered by the Interactive & Digital Media Programme Office, Media Development Authority. The work of W. Xia was supported by NSFC under Grant 61073192, by 973 Program under Grant 2011CB302905, and by Singapore EMA under an EIRP02 Grant.

W. Xia is with the School of Computer Science, University of Science and Technology of China, Hefei 230026, China, and also with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: wenfeng_xia@ntu.edu.sg).

Y. Wen and D. Niyato are with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ygwen@ntu.edu.sg; dniyato@ntu.edu.sg).

C. H. Foh is with Centre for Communication Systems Research at the University of Surrey, Guildford GU2 7XH, U.K. (e-mail: c.foh@surrey.ac.uk).

H. Xie is with the Cyberspace and Data Science Laboratory, Chinese Academy of Electronics and Information Technology, Beijing 100846, China, and also with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China (e-mail: haiyong.xie@acm.org).

I. Introduction

EMERGING mega trends in the ICT domain [1], in particular, mobile, social, cloud [2] and big data [3], [4], are urging computer networks for high bandwidth, ubiquitous accessibility, and dynamic management. First, the growing popularity of rich multimedia contents and increasing demand for big data analytics of a diverse set of data sources, are demanding higher network connection speed than ever. For example, social TV [5]–[7] and Ultra High Definition (UHD) television bring "north-south" client-server traffic tsunami to data centers, and big data analytic applications, like MapReduce [8], trigger large "east-west" server-to-server traffic in data centers to partition input data and combine output results. Second, a wide penetration of mobile devices and social networks is demanding ubiquitous communications to fulfill the social needs of general population. The number of mobile-connected devices is predicted to exceed the number of people on earth by the end of 2014, and by 2018 there will be nearly 1.4 mobile devices per capita [9]. Social networks have also experienced a dramatic growth in recent years. For instance, Facebook expanded from 1 million users in December 2004 to more than 1 billion active users in October 2012 [10]. Finally, cloud computing has added further demands on the flexibility and agility of computer networks. Specifically, one of the key characteristics for Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) is the self-managed service [2], dictating a high level of automatic configuration in the system. At the same time, with more computing and storage resources placed remotely in the cloud, efficient access to these resources via a network is becoming critical to fulfill today's computing needs. As such, computer networking has become the crucial enabling technology to move forward these emerging ICT mega trends.

In response to the aforementioned requirements for computer networks, one immediate solution would be to make additional investment in the network infrastructure to enhance the capability of existing computer networks, as practiced in reality. It is reported that the worldwide network infrastructure will accommodate nearly three networked devices and 15 gigabytes data per capita in 2016, up from over one networked device and 4 gigabytes data per capita in 2011 [11]. However, such an expansion of network infrastructure would result in an increase in complexity. First, networks are enormous in size. Even the network for a medium size organization, for example, a campus network, could be composed of hundreds or even thousands of devices [12]. Second, networks are highly heterogeneous, especially when equipment, applications, and services are provided by different manufacturers, vendors, and providers.

are reported to be the biggest contributor to network downtime, responsible for 50 to 80 percent of network device outages [13]. This growing complexity further demands novel approaches to future computer networks, in which the complexity can be managed.

Owing to size, heterogeneity, and complexity of current and, possibly, future computer networks, traditional approaches for configuration, optimization, and troubleshooting would become inefficient, and in some cases, insufficient. For example, Autonomous System (AS) based approaches often focus on managing a subset of networks and optimizing performance or quality of user experience for some network services, as in the case of network-oblivious P2P applications [14] and video streaming rate picking [15]. As a result, they often lead to suboptimal performance with a marginal global performance gain. Moreover, implementation of local optimizations in a single domain, without cross-domain coordination, may cause unnecessary conflicting operations with undesirable outcomes. The situation could be made worse as legacy network platforms does not have inbuilt programmability, flexibility and support to implement and test new networking ideas without interrupting ongoing services [16]. Even when new network configuration, optimization, or recovery methods are developed, implementation and testing can take years from design to standardization before a possible deployment. A protocol can take years to be standardized as an RFC [17], [18]. These observations have demanded a novel approach for future networks to support implementation, testing, and deployment of innovative ideas.

Indeed, networking research community and industry have long noticed the aforementioned problems. Previously a few new ideas have been introduced for a better design of future networks [19], including Named Data Networking (NDN) [20], programmable networks [21], "HTTP as the narrow waist" [22] and Software-Defined Networking (SDN) [23]. In particular, SDN is touted as a most promising solution. The key idea of SDN is to decouple the control plane from the data plane and allow flexible and efficient management and operation of the network via software programs. Specifically, devices (e.g., switches and routers) in the data plane perform packet forwarding, based on rules installed by controllers. Controllers in the control plane oversee the underlying network and provide a flexible and efficient platform to implement various network applications and services. Under this new paradigm, innovative solutions for specific purposes (e.g., network security, network virtualization and green networking) can be rapidly implemented in form of software and deployed in networks with real traffic. Moreover, SDN allows logical centralization of feedback control with better decisions based on a global network view and cross-layer information.

In this article, we survey the SDN literature and aim at presenting the definition of SDN and its architectural principle, providing an overview of the recent developments in SDN, and discussing about research issues and approaches for future SDN developments. The rest of this article is organized as follows. We first present the definition of SDN and its key benefits and challenges in Section II. The next three sections

which discusses approaches to build SDN-capable switching devices and challenges of utilizing different transmission media. Section IV deals with the control layer, which introduces operations of an SDN controller and performance issues of the controller. Section V addresses issues at the application layer. This section presents some applications developed on SDN platforms, including adaptive routing, boundless mobility, network management, network security, network virtualization, green networking, and a special SDN use case with cloud computing. Section VI covers OpenFlow, which is considered as the de facto implementation of SDN. A brief conclusion with some discussion on current implementations and further developments of SDN is presented in Section VII.

II. SDN: Definition, Benefits, and Challenges

Lately SDN has become one of the most popular subjects in the ICT domain. However, being a new concept, a consensus has not yet been reached on its exact definition. In fact, a lot of different definitions [23]–[28] have surfaced over the last couple of years, each of which has its own merits. In this section, we first present a generally accepted definition of SDN, and then outline a set of key benefits and challenges of SDN, and finally introduce an SDN reference model as the anchor of this survey paper.

A. *Definition of SDN*

The Open Networking Foundation (ONF) [29] is a non-profit consortium dedicated to development, standardization, and commercialization of SDN. ONF has provided the most explicit and well received definition of SDN as follows:

> Software-Defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable [23].

Per this definition, SDN is defined by two characteristics, namely decoupling of control and data planes, and programmability on the control plane. Nevertheless, neither of these two signatures of SDN is totally new in network architecture, as detailed in the following.

First, several previous efforts have been made to promote network programmability. One example is the concept of active networking that attempts to control a network in a real-time manner using software. SwitchWare [30], [31] is an active networking solution, allowing packets flowing through a network to modify operations of the network dynamically. Similarly, software routing suites on conventional PC hardware, such as Click [32], XORP [33], Quagga [34], and BIRD [35], also attempt to create extensible software routers by making network devices programmable. Behavior of these network devices can be modified by loading different or modifying existing routing software.

Second, the spirit of decoupling between control and data planes has been proliferated during the last decade. Caesar *et al.* first presented a Routing Control Platform (RCP) in 2004 [36], in which Border Gateway Protocol (BGP) inter-domain year, IETF released the Forwarding and Control Element Separation (ForCES) framework, which separates control and packet forwarding elements in a ForCES Network [37]–[40]. In 2005, Greenberg *et al.* proposed a 4D approach [41]–[43], introducing a clean slate design of the entire network architecture with four planes. These planes are "decision", "dissemination", "discovery", and "data", respectively, which are organized from top to bottom. In 2006, the Path Computation Element (PCE) architecture was presented to compute label switched paths separately from actual packet forwarding in MPLS and GMPLS networks [44]. In 2007, Casado *et al.* presented Ethane, where simple flow-based Ethernet switches are supplemented with a centralized controller to manage admittance and routing of flows [45]–[48]. In this latest development, the principle of data-control plane separation has been explicitly stated. Commercial networking devices have also adopted the idea of data-control plane separation. For example, in the Cisco ASR 1000 series routers and Nexus 7000 series switches, the control plane is decoupled from the data plane and modularized, allowing coexistence of an active control plane instance and a standby one for high fault tolerance and transparent software upgrade.

In the context of SDN, its uniqueness resides on the fact that it provides programmability through decoupling of control and data planes. Specifically, SDN offers simple programmable network devices rather than making networking devices more complex as in the case of active networking. Moreover, SDN proposes separation of control and data planes in the network architectural design. With this design, network control can be done separately on the control plane without affecting data flows. As such, network intelligence can be taken out of switching devices and placed on controllers. At the same time, switching devices can now be externally controlled by software without onboard intelligence. The decoupling of control plane from data plane offers not only a simpler programmable environment but also a greater freedom for external software to define the behavior of a network.

B. *SDN Benefits*

SDN, with its inherent decoupling of control plane from data plane, offers a greater control of a network through programming. This combined feature would bring potential benefits of enhanced configuration, improved performance, and encouraged innovation in network architecture and operations, as summarized in Table I. For example, the control embraced by SDN may include not only packet forwarding at a switching level but also link tuning at a data link level, breaking the barrier of layering. Moreover, with an ability to acquire instantaneous network status, SDN permits a real-time centralized control of a network based on both instantaneous network status and user defined policies. This further leads to benefits in optimizing network configurations and improving network performance. The potential benefit of SDN is further evidenced by the fact that SDN offers a convenient platform for experimentations of new techniques and encourages new network designs, attributed to its network programmability and the ability to define isolated

1) Enhancing Configuration: In network management, configuration is one of the most important functions. Specifically, when new equipment is added into an existing network, proper configurations are required to achieve coherent network operation as a whole. However, owing to the heterogeneity among network device manufacturers and configuration interfaces, current network configuration typically involves a certain level of manual processing. This manual configuration procedure is tedious and error prone. At the same time, significant effort is also required to troubleshoot a network with configuration errors. It is generally accepted that, with the current network design, automatic and dynamic reconfiguration of a network remains a big challenge. SDN will help to remedy such a situation in network management. In SDN, unification of the control plane over all kinds of network devices [50], including switches, routers, Network Address Translators (NATs), firewalls, and load balancers, renders it possible to configure network devices from a single point, automatically via software controlling. As such, an entire network can be programmatically configured and dynamically optimized based on network status.

2) Improving Performance: In network operations, one of the key objectives is to maximize utilization of the invested network infrastructure. However, owing to coexistence of various technologies and stakeholders in a single network, optimizing performance of the network as a whole has been considered difficult. Current approaches often focus on optimizing performance of a subset of networks or the quality of user experience for some network services. Obviously, these approaches, based on local information without cross-layer consideration, could lead to suboptimal performance, if not conflicting network operations. The introduction of SDN offers an opportunity to improve network performance globally. Specifically, SDN allows for a centralized control with a global network view and a feedback control with information exchanged between different layers in the network architecture. As such, many challenging performance optimization problems would become manageable with properly designed centralized algorithms. It follows that new solutions for classical problems, such as data traffic scheduling [51], end-to-end congestion control [52], load balanced packet routing [53], energy efficient operation [54], and Quality of Service (QoS) support [55], [56], can be developed and easily deployed to verify their effectiveness in improving network performance.

3) Encouraging Innovation: In the presence of continuing evolution of network applications, future network should encourage innovation rather than attempt to precisely predict and perfectly meet requirements of future applications [57]. Unfortunately, any new idea or design immediately faces challenges in implementation, experimentation, and deployment into existing networks. The main hurdle arises from widely used proprietary hardware in conventional network components, preventing modification for experimentation. Besides, even when experimentations are possible, they are often conducted in a separate simplified testbed. These experimentations do not give sufficient confidence for industrial adaptation of these new ideas or network designs. The idea behind community

TABLE I
COMPARISONS BETWEEN SDN AND CONVENTIONAL NETWORKING

	SDN	Conventional Networking
Features	decoupled data and control plane, and programmability	a new protocol per problem, complex network control [49]
Configuration	automated configuration with centralized validation	error prone manual configuration
Performance	dynamic global control with cross layer information	limited information, and relatively static configuration
Innovation	easy software implementation for new ideas, sufficient test environment with isolation, and quick deployment using software upgrade	difficult hardware implementation for new ideas, limited testing environment, long standardization process

In comparison, SDN encourages innovation by providing a programmable network platform to implement [60], experiment [61], and deploy new ideas, new applications, and new revenue earning services conveniently and flexibly. High configurability of SDN offers clear separation among virtual networks permitting experimentation on a real environment. Progressive deployment of new ideas can be performed through a seamless transition from an experimental phase to an operational phase.

C. SDN Challenges

Given the promises of enhanced configuration, improved performance, and encouraged innovation, SDN is still in its infancy. Many fundamental issues still remain not fully solved, among which standardization and adoption are the most urgent ones.

Though the ONF definition of SDN is most received one, OpenFlow sponsored by ONF is by no means the only SDN standard and by no means a mature solution. An open-source OpenFlow driver is still absent for SDN controller development, a standard north-bound API or a high level programming language is still missing for SDN application development. A healthy ecosystem combining network device vendors, SDN application developers, and network device consumers, has yet to appear.

SDN offers a platform for innovative networking techniques, however the shift from traditional networking to SDN can be disruptive and painful. Common concerns include SDN interoperability with legacy network devices, performance and privacy concerns of centralized control, and lack of experts for technical support. Existing deployments of SDN are often limited to small testbed for research prototypes. Prototypes for research purpose remain premature to offer confidence for real world deployment.

D. SDN Reference Model

ONF has also suggested a reference model for SDN, as illustrated in Fig. 1. This model consists of three layers, namely an infrastructure layer, a control layer, and an application layer, stacking over each other.

The infrastructure layer consists of switching devices (e.g., switches, routers, etc.) in the data plane. Functions of these switching devices are mostly two-fold. First, they are responsible for collecting network status, storing them temporally in

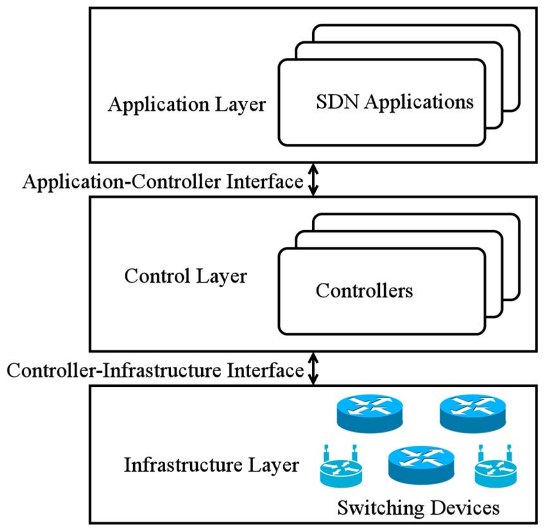

Fig. 1. SDN Reference Model: a three-layer model, ranging from an infrastructure layer to a control layer to an application layer, in a bottom-up manner.

statistics, and network usages. Second, they are responsible for processing packets based on rules provided by a controller.

The control layer bridges the application layer and the infrastructure layer, via its two interfaces. For downward interacting with the infrastructure layer (i.e., the south-bound interface), it specifies functions for controllers to access functions provided by switching devices. The functions may include reporting network status and importing packet forwarding rules. For upward interacting with the application layer (i.e., the north-bound interface), it provides service access points in various forms, for example, an Application Programming Interface (API). SDN applications can access network status information reported from switching devices through this API, make system tuning decisions based on this information, and carry out these decisions by setting packet forwarding rules to switching devices using this API. Since multiple controllers will exist for a large administrative network domain, an "east-west" communication interface among the controllers will also be needed for the controllers to share network information and coordinate their decision-making processes [62], [63].

The application layer contains SDN applications designed to fulfill user requirements. Through the programmable platform provided by the control layer, SDN applications are able to access and control switching devices at the infrastructure layer. Example of SDN applications could include dynamic access control, seamless mobility and migration, server load balancing, and network virtualization.

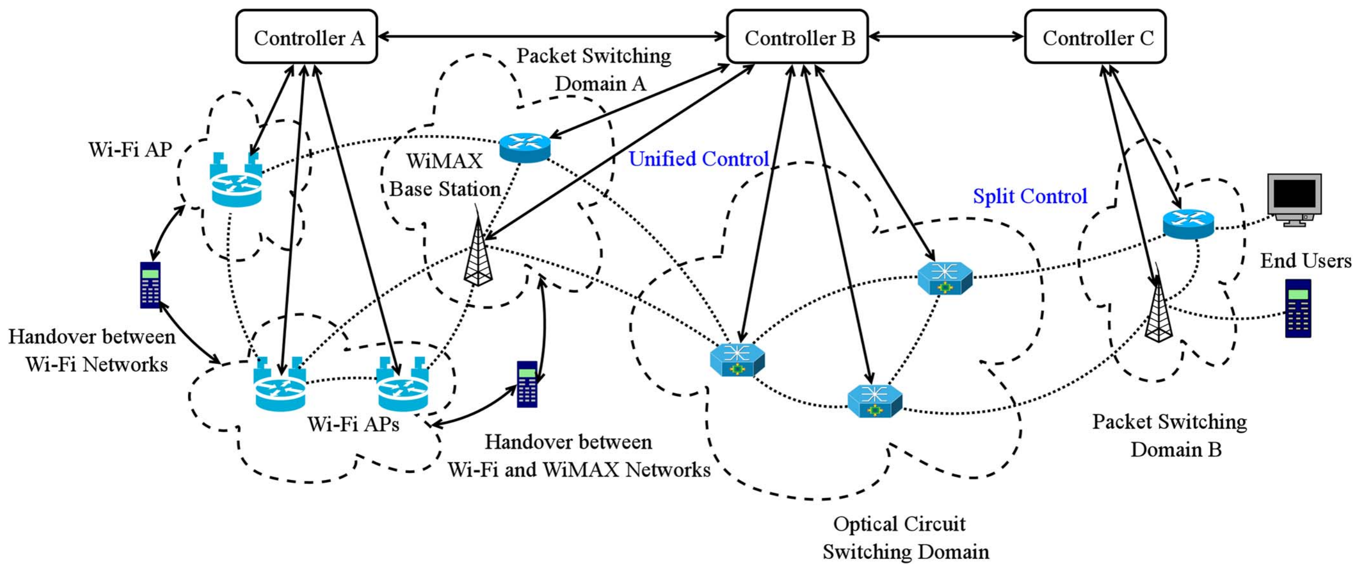
Controller A
Controller B
Controller C
Packet Switching Domain A
Wi-Fi AP
WiMAX Base Station
Unified Control
Split Control
End Users
Handover between Wi-Fi Networks
Wi-Fi APs
Handover between Wi-Fi and WiMAX Networks
Packet Switching Domain B
Optical Circuit Switching Domain

Fig. 2. SDN Infrastructure Architecture: switching devices are connected to formulate a mesh topology via various transmission media, including copper wires, wireless radio and optical fibre.

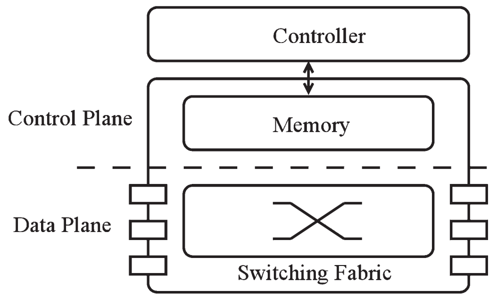

Fig. 3. Switching Device Model in SDN: a two-layer logical model consisting of a processor for data forwarding and onboard memory for control information.

III. Infrastructure Layer

At the lowest layer in the SDN reference model, the infrastructure layer consists of switching devices (e.g., switches, routers, etc.), which are interconnected to formulate a single network. The connections among switching devices are through different transmission media, including copper wires, wireless radio, and also optical fibers. In Fig. 2, we illustrate an SDN-enabled reference network. In particular, the main research concerns associated with the infrastructure layer include both efficient operations of switching devices and utilization of transmission media, as detailed in the next two subsections.

A. Switching Devices

In Fig. 3, we illustrate the architectural design of an SDN switching device, consisting of two logical components for the data plane and the control plane. In the data plane, the switching device, in particular, through its processor, performs packet forwarding, based on the forwarding rules imposed by the control layer. Examples of network processors include XLP processor family (MIPS64 architecture) from Broadcom, XScale processor (ARM architecture) from Intel, NP-x NPUs from architecture) from Netronome, Xelerated HX family from Marvell, OCTEON series processors (MIPS64 architecture) form Cavium and general purpose CPUs from Intel and AMD. In the control plane, the switching device communicates with controllers at the control layer to receive rules, including packet forwarding rules at a switching level and link tuning rules at a data-link level, and stores the rules in its local memory. Examples of memory include TCAM and SRAM.

This new architectural principle lends SDN competitive advantages. Unlike conventional switching devices that also run routing protocols to decide how to forward packets, routing decision makings are stripped from switching devices in SDN. As a result, the switching devices are simply responsible for gathering and reporting network status as well as processing packets based on imposed forwarding rules. It follows that the SDN switching devices are simpler and will be easier to manufacture. The reduced complexity in turn leads to a low cost solution.

This new architecture, however, requires new hardware design for SDN-enabled switching devices. In this subsection, we describe recent research progresses in switching device hardware design, focusing on both the control plane and the data plane. We will also classify the most popular switching device platforms, and discuss testing and evaluation of these switching devices.

1) Control Plane: In the control plane of SDN switching devices, one of the main design challenges resides on the efficient use of onboard memory. Fundamentally, memory usage in a switching device depends on the network scale. Specifically, switching devices in a larger scale network would need a larger memory space; otherwise, they may need constant hardware upgrades to avoid memory exhaustion. In case of insufficient memory space, packets would be dropped or directed to controllers for further decisions on how to process them, resulting in a degraded network performance [64].

Memory management techniques in traditional switch design

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase® Smarter legal research.