*Proceedings of the 8th Conference on Architectural Support for Programming Languages and Operating Systems, 1998.*

# A Cost-Effective, High-Bandwidth Storage Architecture

Garth A. Gibson*, David F. Nagle†, Khalil Amiri†, Jeff Butler†, Fay W. Chang*,
Howard Gobioff*, Charles Hardin†, Erik Riedel†, David Rochberg*, Jim Zelenka*

School of Computer Science*
Department of Electrical and Computer Engineering†
Carnegie Mellon University, Pittsburgh, PA 15213
garth+asplos98@cs.cmu.edu

## ABSTRACT

*This paper describes the Network-Attached Secure Disk (NASD) storage architecture, prototype implementations of NASD drives, array management for our architecture, and three filesystems built on our prototype. NASD provides scalable storage bandwidth without the cost of servers used primarily for transferring data from peripheral networks (e.g. SCSI) to client networks (e.g. ethernet). Increasing dataset sizes, new attachment technologies, the convergence of peripheral and interprocessor switched networks, and the increased availability of on-drive transistors motivate and enable this new architecture. NASD is based on four main principles: direct transfer to clients, secure interfaces via cryptographic support, asynchronous non-critical-path oversight, and variably-sized data objects. Measurements of our prototype system show that these services can be cost-effectively integrated into a next generation disk drive ASIC. End-to-end measurements of our prototype drive and filesystems suggest that NASD can support conventional distributed filesystems without performance degradation. More importantly, we show scalable bandwidth for NASD-specialized filesystems. Using a parallel data mining application, NASD drives deliver a linear scaling of 6.2 MB/s per client-drive pair, tested with up to eight pairs in our lab.*

## Keywords

D.4.3 File systems management, D.4.7 Distributed systems, B.4 Input/Output and Data Communications.

## 1. INTRODUCTION

Demands for storage bandwidth continue to grow due to rapidly increasing client performance, richer data types such as video, and data-intensive applications such as data mining. For storage subsystems to deliver scalable bandwidth, that is, linearly increasing application bandwidth with increasing numbers of storage devices and client processors, the data must be striped over many disks and network links [Patterson88]. With 1998 technology, most office, engineering, and data processing shops have sufficient numbers of disks and scalable switched networking, but they access storage through storage controller and distributed fileserver bottlenecks. These bottlenecks arise because a single "server" computer receives data from the storage (peripheral) network and forwards it to the client (local area) network while adding functions such as concurrency control and metadata consistency. A variety of research projects have explored techniques for scaling the number of machines used to enforce the semantics of such controllers or fileservers [Cabrera91, Hartman93, Cao93, Drapeau94, Anderson96, Lee96, Thekkath97]. As Section 3 shows, scaling the number of machines devoted to store-and-forward copying of data from storage to client networks is expensive.

This paper makes a case for a new scalable bandwidth storage architecture, Network-Attached Secure Disks (NASD), which separates management and filesystem semantics from store-and-forward copying. By evolving the interface for commodity storage devices (SCSI-4 perhaps), we eliminate the server resources required solely for data movement. As with earlier generations of SCSI, the NASD interface is simple, efficient and flexible enough to support a wide range of filesystem semantics across multiple generations of technology. To demonstrate how a NASD architecture can deliver scalable bandwidth, we describe a prototype implementation of NASD, a storage manager for NASD arrays, and a simple parallel filesystem that delivers scalable bandwidth to a parallel data-mining application. Figure 1 illustrates the components of a NASD system and indicates the sections describing each.

We continue in Section 2 with a discussion of storage system architectures and related research. Section 3 presents enabling technologies for NASD. Section 4 presents an overview of our NASD interface, its implementation and performance. Section 5 discusses ports of NFS and AFS filesystems to a NASD-based system, our implementation of a NASD array management system, a simple parallel filesystem, and an I/O-intensive data mining application that exploits the bandwidth of our prototype. This section reports the scalability of our prototype compared to the performance of a fast single NFS server. Section 6 discusses active disks, the logical extension of NASD to execute application code. Section 7 concludes with a discussion of ongoing research.

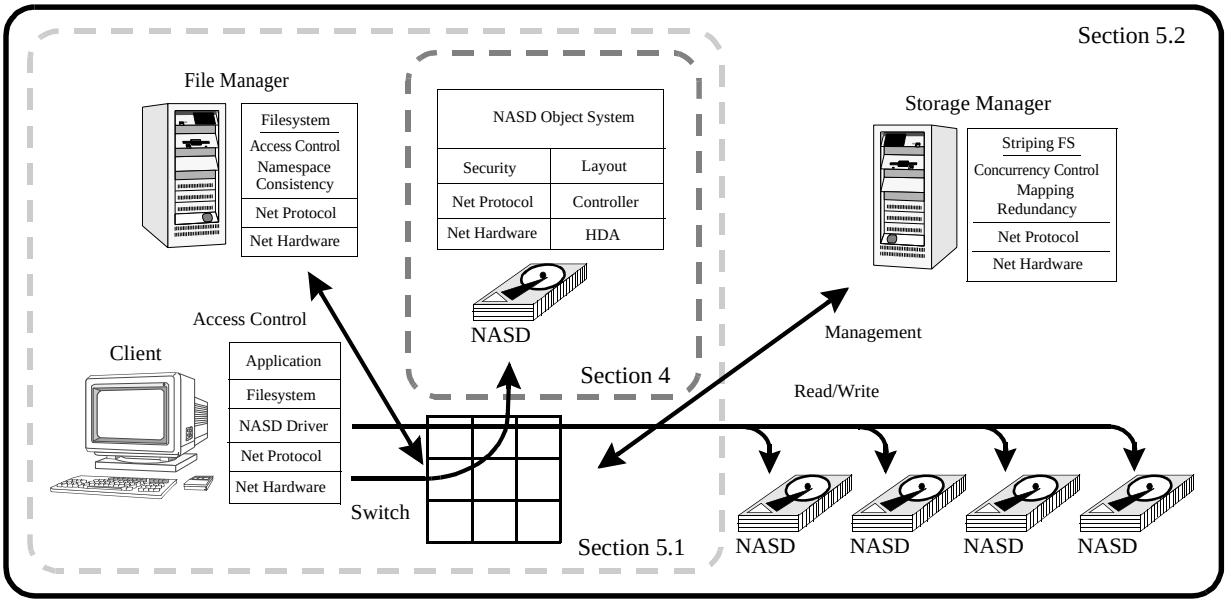



Figure 1: An overview of a scalable bandwidth NASD system. The major components are annotated with the layering of their logical components. The innermost box shows a basic NASD drive as described in Section 4. The larger box contains the essentials for a NASD-based filesystem, which adds a file manager and client as detailed in Section 5.1. Finally, the outer box adds a storage manager to coordinate drives on which parallel filesystem is built as discussed in Section 5.2.

## 2. BACKGROUND AND RELATED WORK

Figure 2 illustrates the principal alternative storage architectures: (1) a local filesystem, (2) a distributed filesystem (DFS) built directly on disks, (3) a distributed filesystem built on a storage subsystem, (4) a network-DMA distributed filesystem, (5) a distributed filesystem using smart object-based disks (NASD) and (6) a distributed filesystem using a second level of objects for storage management.

The simplest organization (1) aggregates the application, file management (naming, directories, access control, concurrency control) and low-level storage management. Disk data makes one trip over a simple peripheral area network such as SCSI or Fibrechannel and disks offer a fixed-size block abstraction. Stand-alone computer systems use this widely understood organization.

To share data more effectively among many computers, an intermediate server machine is introduced (2). If the server offers a simple file access interface to clients, the organization is known as a distributed filesystem. If the server processes data on behalf of the clients, this organization is a distributed database. In organization (2), data makes a second network trip to the client and the server machine can become a bottleneck, particularly since it usually serves large numbers of disks to better amortize its cost.

The limitations of using a single central fileserver are widely recognized. Companies such as Auspex and Network Appliance have attempted to improve file server performance, specifically the number of clients supported, through the use of special purpose server hardware and highly optimized software [Hitz90, Hitz94]. Although not the topic of this paper, the NASD architecture can improve the client-load-bearing capability of traditional filesystems by off-loading simple data-intensive processing to NASD drives [Gibson97a].

To transparently improve storage bandwidth and reliability many systems interpose another computer, such as a RAID controller [Patterson88]. This organization (3) adds another peripheral network transfer and store-and-forward stage for data to traverse.

Provided that the distributed filesystem is reorganized to logically “DMA” data rather than copy it through its server, a fourth organization (4) reduces the number of network transits for data to two. This organization has been examined extensively [Drapeau94, Long94] and is in use in the HPSS implementation of the Mass Storage Reference Model [Watson95, Miller88]. Organization (4) also applies to systems where clients are trusted to maintain filesystem metadata integrity and implement disk striping and redundancy [Hartman93, Anderson96]. In this case, client caching of metadata can reduce the number of network transfers for control messages and data to two. Moreover, disks can be attached to client machines which are presumed to be independently paid for and generally idle. This eliminates additional store-and-forward cost, if clients are idle, without eliminating the copy itself.

As described in Section 4, the NASD architecture (5) embeds the disk management functions into the device and offers a variable-length object storage interface. In this organization, file managers enable repeated client accesses to specific storage objects by granting a cachable capability.

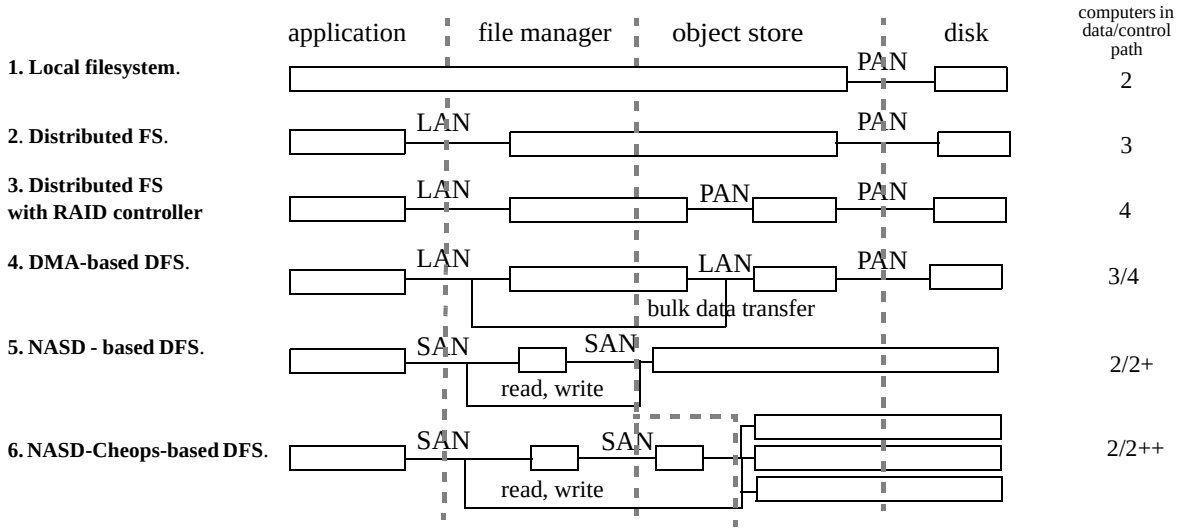



Figure 2: Evolution of storage architectures for untrusted networks and clients. Boxes are computers, horizontal lines are communication paths and vertical lines are internal and external interfaces. *LAN* is a local area network such as Ethernet or FDDI. *PAN* is a peripheral area network such as SCSI, Fibrechannel or IBM's ESCON. *SAN* is an emerging system area network such as ServerNet, Myrinet or perhaps Fibrechannel or Ethernet that is common across clients, servers and devices. On the far right, a *disk* is capable of functions such as seek, read, write, readahead, and simple caching. The *object store* binds blocks into variable-length objects and manages the layout of these objects in the storage space offered by the device(s). The *file manager* provides naming, directory hierarchies, consistency, access control, and concurrency control. In NASD, storage management is done by recursion on the object interface on the SAN.

Hence, all data and most control travels across the network once and there is no expensive store-and-forward computer.

The idea of a simple, disk-like network-attached storage server as a building block for high-level distributed filesystems has been around for a long time. Cambridge's Universal File Server (UFS) used an abstraction similar to NASD along with a directory-like index structure [Birrell80]. The UFS would reclaim any space that was not reachable from a root index. The successor project at Cambridge, CFS, also performed automatic reclamation and added undoable (for a period of time after initiation) transactions into the filesystem interface [Mitchell81]. To minimize coupling of file manager and device implementations, NASD offers less powerful semantics, with no automatic reclamation or transaction rollback.

Using an object interface in storage rather than a fixed-block interface moves data layout management to the disk. In addition, NASD partitions are variable-sized groupings of objects, not physical regions of disk media, enabling the total partition space to be managed easily, in a manner similar to virtual volumes or virtual disks [IEEE95, Lee96]. We also believe that specific implementations can exploit NASD's uninterpreted filesystem-specific attribute fields to respond to higher-level capacity planning and reservation systems such as HP's attribute-managed storage [Golding95]. Object-based storage is also being pursued for quality-of-service at the device, transparent performance optimizations, and drive supported data sharing [Anderson98a].

ISI's Netstation project [VanMeter96] proposes a form of object-based storage called Derived Virtual Devices (DVD) in which the state of an open network connection is augmented with access control policies and object metadata, provided by the file manager using Kerberos [Neuman94] for underlying security guarantees. This is similar to NASD's mechanism except that NASD's access control policies are embedded in unforgeable capabilities separate from communication state, so that their interpretation persists (as objects) when a connection is terminated. Moreover, Netstation's use of DVD as a physical partition server in VISA [VanMeter98] is not similar to our use of NASD as a single-object server in a parallel distributed filesystem.

In contrast to the ISI approach, NASD security is based on capabilities, a well-established concept for regulating access to resources [Dennis66]. In the past, many systems have used capabilities that rely on hardware support or trusted operating system kernels to protect system integrity [Wulf74, Wilkes79, Karger88]. Within NASD, we make no assumptions about the integrity of the client to properly maintain capabilities. Therefore, we utilize cryptographic techniques similar to ISCAP [Gong89] and Amoeba [Tanenbaum86]. In these systems, both the entity issuing a capability and the entity validating a capability must share a large amount of private information about all of the issued capabilities. These systems are generally implemented as single entities issuing and validating capabilities, while in NASD these functions are done in distinct machines and no per-capability state is exchanged between issuer and validator.

To offer disk striping and redundancy for NASD, we layer the NASD interface. In this organization (6), a storage man-

ager replaces the file manager's capability with a set of capabilities for the objects that actually make up the high-level striped object. This costs an additional control message but once equipped with these capabilities, clients again access storage objects directly. Redundancy and striping are done within the objects accessible with the client's set of capabilities, not the physical disk addresses.

Our storage management system, Cheops, differs from other storage subsystems with scalable processing power such as Swift, TickerTAIP and Petal [Long94, Cao93, Lee96] in that Cheops uses client processing power rather than scaling the computational power of the storage subsystem. Cheops is similar to the Zebra and xFS filesystems except that client trust is not required because the client manipulates only objects it can access [Hartman93, Anderson96].

## 3. ENABLING TECHNOLOGY

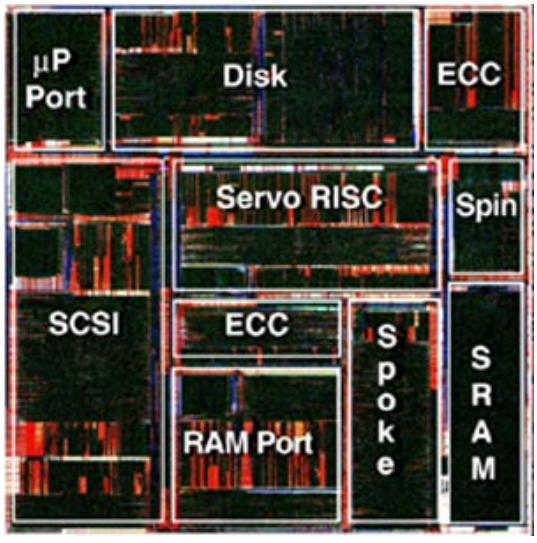
Storage architecture is ready to change as a result of the synergy between five overriding factors: I/O bound applications, new drive attachment technologies, an excess of on-drive transistors, the convergence of peripheral and interprocessor switched networks, and the cost of storage systems.

**I/O-bound applications:** Traditional distributed filesystem workloads are dominated by small random accesses to small files whose sizes are growing with time, though not dramatically [Baker91, TPC98]. In contrast, new workloads are much more I/O-bound, including data types such as video and audio, and applications such as data mining of retail transactions, medical records, or telecommunication call records.

**New drive attachment technology:** The same technology improvements that are increasing disk density by 60% per year are also driving up disk bandwidth at 40% per year [Grochowski96]. High transfer rates have increased pressure on the physical and electrical design of drive busses, dramatically reducing maximum bus length. At the same time, people are building systems of clustered computers with shared storage. For these reasons, the storage industry is moving toward encapsulating drive communication over Fibrechannel [Benner96], a serial, switched, packet-based peripheral network that supports long cable lengths, more ports, and more bandwidth. One impact of NASD is to evolve the SCSI command set that is currently being encapsulated over Fibrechannel to take full advantage of the promises of that switched-network technology for both higher bandwidth and increased flexibility.

**Excess of on-drive transistors:** The increasing transistor density in inexpensive ASIC technology has allowed disk drive designers to lower cost and increase performance by integrating sophisticated special-purpose functional units into a small number of chips. Figure 3 shows the block diagram for the ASIC at the heart of Quantum's Trident drive. When drive ASIC technology advances from 0.68 micron CMOS to 0.35 micron CMOS, they could insert a 200 MHz StrongARM microcontroller, leaving 100,000 gate-equivalent space for functions such as onchip DRAM or cryptographic support. While this may seem like a major jump, Siemen's TriCore integrated microcontroller and ASIC architecture promises to deliver a 100 MHz, 3-way issue,

**(a) Current Trident ASIC (74 mm$^2$ at 0.68 micron)**




**(b) Next-generation ASIC (0.35 micron technology)**

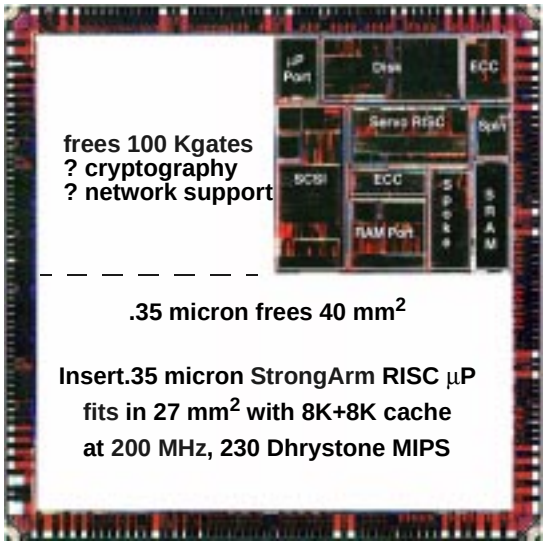



Figure 3: Quantum's Trident disk drive features the ASIC on the left (a). Integrated onto this chip in four independent clock domains are 10 function units with a total of about 110,000 logic gates and a 3 KB SRAM: a disk formatter, a SCSI controller, ECC detection, ECC correction, spindle motor control, a servo signal processor and its SRAM, a servo data formatter (spoke), a DRAM controller, and a microprocessor port connected to a Motorola 68000 class processor. By advancing to the next higher ASIC density, this same die area could also accommodate a 200 MHz StrongARM microcontroller and still have space left over for DRAM or additional functional units such as cryptographic or network accelerators.

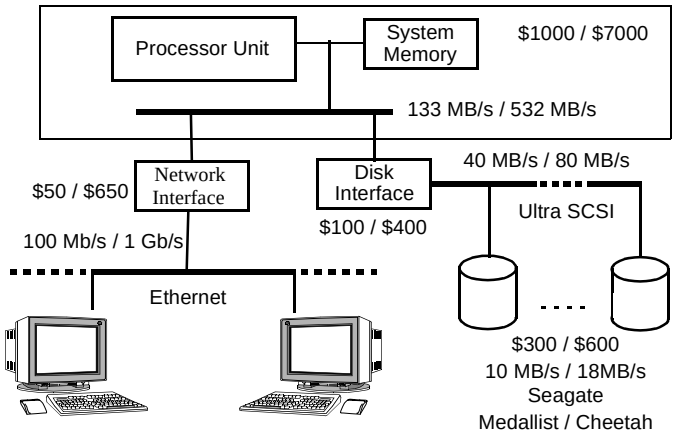



Figure 4: Cost model for the traditional server architecture. In this simple model, a machine serves a set of disks to clients using a set of disk (wide Ultra and Ultra2 SCSI) and network (Fast and Gigabit Ethernet) interfaces. Using peak bandwidths and neglecting host CPU and memory bottlenecks, we estimate the server cost overhead at maximum bandwidth as the sum of the machine cost and the costs of sufficient numbers of interfaces to transfer the disks' aggregate bandwidth divided by the total cost of the disks. While the prices are probably already out of date, the basic problem of a high server overhead is likely to remain. We report pairs of costs and bandwidth estimates. On the left, we show values for a low cost system built from high-volume components. On the right, we show values for a high-performance reliable system built from components recommended for mid-range and enterprise servers [Pricewatch98].

32-bit datapath with up to 2 MB of onchip DRAM and customer defined logic in 1998 [TriCore97].

**Convergence of peripheral and interprocessor networks:** Scalable computing is increasingly based on clusters of workstations. In contrast to the special-purpose, highly reliable, low-latency interconnects of massively parallel processors such as the SP2, Paragon, and Cosmic Cube, clusters typically use Internet protocols over commodity LAN routers and switches. To make clusters effective, low-latency network protocols and user-level access to network adapters have been proposed, and a new adapter card interface, the Virtual Interface Architecture, is being standardized [Maeda93, Wilkes92, Boden95, Horst95, vonEicken95, Intel97]. These developments continue to narrow the gap between the channel properties of peripheral interconnects and the network properties of client interconnects [Sachs94] and make Fibrechannel and Gigabit Ethernet look more alike than different

**Cost-ineffective storage servers:** In high performance distributed filesystems, there is a high cost overhead associated with the server machine that manages filesystem semantics and bridges traffic between the storage network and the client network [Anderson96]. Figure 4 illustrates this problem for bandwidth-intensive applications in terms of maximum storage bandwidth. Based on these cost and peak performance estimates, we can compare the expected overhead cost of a storage server as a fraction of the raw storage cost. Servers built from high-end components have an overhead that starts at 1,300% for one server-attached disk! Assuming dual 64-bit PCI busses that deliver every byte into and out of memory once, the high-end server saturates with 14 disks, 2 network interfaces, and 4 disk interfaces with a 115% overhead cost. The low cost server is more cost effective. One disk suffers a 380% cost overhead and, with a 32-bit PCI bus limit, a six disk system still suffers an 80% cost overhead.

While we can not accurately anticipate the marginal increase in the cost of a NASD over current disks, we estimate that the disk industry would be happy to charge 10% more. This bound would mean a reduction in server overhead costs of at least a factor of 10 and in total storage system cost (neglecting the network infrastructure) of over 50%.

## 4. NETWORK-ATTACHED SECURE DISKS

Network-Attached Secure Disks (NASD) enable cost-effective bandwidth scaling. NASD eliminates the server bandwidth bottleneck by modifying storage devices to transfer data directly to clients and also repartitions traditional file server or database functionality between the drive, client and server as shown in Figure 1. NASD presents a flat name space of variable-length objects that is both simple enough to be implemented efficiently yet flexible enough for a wide variety of applications. Because the highest levels of distributed filesystem functionality—global naming, access control, concurrency control, and cache coherency—vary significantly, we do not advocate that storage devices subsume the file server entirely. Instead, the residual filesystem, the *file manager*, should define and manage these high-level policies while NASD devices should implement simple storage primitives efficiently and operate as independently of the file manager as possible.

Broadly, we define NASD to be storage that exhibits the following four properties:

**Direct transfer:** Data is transferred between drive and client without indirection or store-and-forward through a file server machine.

**Asynchronous oversight:** We define asynchronous oversight as the ability of the client to perform most operations without synchronous appeal to the file manager. Frequently consulted but infrequently changed policy decisions, such as authorization decisions, should be encoded into capabilities by the file manager and subsequently enforced by drives.

**Cryptographic integrity:** By attaching storage to the network, we open drives to direct attack from adversaries. Thus, it is necessary to apply cryptographic techniques to defend against potential attacks. Drives ensure that commands and data have not been tampered with by generating and verifying cryptographic keyed digests. This is essentially the same requirement for security as proposed for IPv6 [Deering95].

DOCKET ALARM

# 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.