Author's accepted version - DOI: https://doi.org/10.1109/CyberSecPODS.2018.8560671
IEEE International Conference on Cyber Security and Protection of Digital Services (Cyber Security 2018)

Reducing the Impact of Network Bottlenecks on Remote Contraband Detection

Sean McKeown
School of Computing
Edinburgh Napier University
Edinburgh, Scotland
s.mckeown@napier.ac.uk

Gordon Russell
School of Computing
Edinburgh Napier University
Edinburgh, Scotland
g.russell@napier.ac.uk

Petra Leimich
School of Computing
Edinburgh Napier University
Edinburgh, Scotland
p.leimich@napier.ac.uk

***Abstract*—Cloud based storage is increasing in popularity, with large volumes of data being stored remotely. Digital forensics investigators examining such systems remotely are limited by bandwidth constraints when accessing this kind of data using traditional tools. This paper explores the potential for sub-file hashing strategies to decrease the time taken to detect contraband on networked storage devices, while maintaining a high degree of accuracy. Results show that sub-file hashing is faster than full file hashing for both LAN and Internet server configurations, with reduced bandwidth heavily favouring sub-file strategies.**

***Index Terms*—networked storage, cloud storage forensics, file server, sub-file signatures, partial-file analysis, hashing strategies, digital forensics, cryptographic hashing, logical acquisition**

I. Introduction

Digital forensics investigators have to cope with increasingly large volumes of data, as the number of devices per case, and capacities of storage media, continue to increase year on year [1]. In particular, digital forensics specialists who deal with cases involving indecent images of children are struggling to cope with the deluge due to time and resource constraints [2]. These circumstances have resulted in substantial law enforcement backlogs, where evidence is securely stored for long periods of time until it can be forensically analysed. In some cases backlogs stretch up to four years from evidence seizure [3], a situation which not only damages the course of justice, but potentially places victims at continued risk.

The huge flood of evidence is no less pronounced in investigations involving corporate assets, with the high profile Megaupload case in 2012 involving the seizure of 150 terabytes of data [4]. The analysis of enterprise data is further complicated in that it is not always possible to seize assets which are required for critical business operations, meaning that evidence must be examined live on running computers, potentially without physical access to the device [5]. Even with physical access and the ability to physically seize servers, enterprise storage is becoming increasingly complex, with virtualised and distributed storage systems that are difficult to reconstruct from raw disk captures. As such, live analysis by accessing servers over a network may be the most feasible option. However, analysis over a network connection is much slower than accessing disk drives directly, with network bottlenecks potentially placing increased pressure on an already constrained investigation timeline.

This paper focuses on the problem of remotely detecting contraband on live networked file servers in a manner that does not require executing programs directly on the server itself. Instead, data is accessed via a network storage protocol, where the primary bottleneck is the throughput of the network connection. Fast processing is achieved by utilising partial-file cryptographic hashing techniques, such that entire files need not be sent over the network, reducing the bottlenecking effect of the network. Two scenarios are explored, with analysis being conducted over a Local Area Network (LAN), and over an Internet connection to a remote file server. Sub-file hashing approaches allow for rapid contraband content detection, taking far less time than traditional full file hashing approaches.

The primary contribution of this paper is a fast and accurate technique for discovering contraband data on networked file storage. Timed benchmark results are provided for both Samba and Network File System (NFS) servers to quantify speed improvements. Experimental data is derived from the publicly accessible Flickr 1 Million [6] and Govdocs [7] datasets.

The remainder of this paper is organised as follows: Section II discusses related work in forensic contraband detection and the acquisition of forensic evidence over network connections. Section III outlines the sub-file hashing strategies and experimental set-up used in this work, with findings presented in Section IV. Finally, concluding comments and suggestions for future work are provided in Section V.

II. Related Work

A. Detecting Contraband

Illegal and copyrighted multimedia files are typically detected by means of cryptographic hashing. Lookup databases of known contraband are created by hashing files when they are encountered, allowing for later automatic detection. When processing a seized piece of evidence, such as a mobile phone or Hard Disk Drive (HDD), each file on the device is hashed separately and checked against this database, with matches determining that contraband has been detected.

Modifications to this hashing process have largely focused on detecting similar, rather than identical, files, either by features of their binary data [8]–[10], or by their semantic, human facing, content [11]. However, such similar file detection is slower to process than traditional cryptographic hashing [12].

Parallel processing models for traditional cryptographic hashing have been explored to reduce the time taken to forensically analyse storage media [13], [14], however they are still fundamentally limited by the speed that data can be acquired from the physical device. An alternative approach is to reduce the amount of data which needs to be read from the storage media while still maintaining high forensic accuracy. Grier and Richard [15] proposed a model which makes use of investigation-specific filters for acquiring evidence from a digital device. By processing file system metadata, key areas of the evidence can be identified and selectively acquired, with analysis taking place on this evidential subset. This approach was shown to reliably capture a large portion of the relevant evidence while greatly reducing the amount of data needing to be captured from a device.

Rather than parsing file system metadata to select data subsets, Penrose et al. [16] uses statistical block sampling and traditional cryptographic hashing to detect contraband files at the block level, ordering disk reads for maximal throughput. By selecting an appropriate sample size for the target media, this approach generates fast results with high levels of statistical confidence. One drawback of this approach is the possibility of common blocks existing between unrelated files, such that mitigation strategies are required to prevent false positives [17].

A different approach is taken by McKeown et al. [18], wherein evidential subsetting is applied at the file level, as opposed to the disk level. The authors describe a method for using PNG header features, and a small chunk of compressed pixel data, to generate discriminative signatures for filtering known contraband. A similar approach for the JPEG file format [19] makes use of optimised Huffman tables for signature generation, which were shown to be unique for over 1 million images. These approaches were shown to reduce forensic processing time substantially, particularly on Solid State Disks (SSDs).

Generalised sub-file signatures, which are not file type specific, were explored in McKeown et al. [20], with an evaluation of signature generation techniques which use both the beginning and end of a file. Reading from the beginning of the file failed to generate unique signatures for two large datasets, even when processing a large number of data blocks. However, reading as few as 4KiB of data from the end of each file generated unique signatures for both datasets. Sub-file signatures were shown to be a fixed cost to calculate, while full file hashing scales linearly with file size. This approach was also shown to have particularly large benefits on SSDs, in line with prior work in this area.

As the quantity of data on storage media continues to grow, reducing the amount of data to process will continue to be an important area of research. In this paper, we further explore the behavioural characteristics of the sub-file hashing strategies introduced in McKeown et al. [20].

B. *Networked Forensic Acquisition*

The most common approach to analysing an electronic device is to isolate the storage media by physically removing it, then connecting it to a write blocker to acquire data from it. However, it may not be practical to seize large volumes of equipment due to physical storage limitations, easily damaged equipment, or conflicting business requirements. In these cases it is possible to collect evidence over a network connection for later analysis.

Scanlon and Kechadi [21] describe the Remote Acquisition Forensics Tool (RAFT), which makes use of a modified Ubuntu Live CD to acquire evidence and send it to a forensic server over the Internet. This approach is intended to expedite the forensics process by reducing the time taken to seize and physically transport evidence, but is limited by the bandwidth of the available Internet connection. There is also the requirement that the analysed machine be booted via the live CD. A similar approach is taken in Koopmans and James [22], which uses a similar live CD to execute automated hash lookups and string searches for the purposes of forensic triage. Collated evidence is transferred to a forensic server using NFS, with similar bandwidth limitations.

A proactive approach to evidence gathering may be taken by a company in order to facilitate fast forensic analysis. Homem et al. [23] describe the Live Evidence Information Aggregation (LEIA), which makes use of a hypervisor on client machines and a peer-to-peer network for distribution, with cloud based storage back-ends. Known hashes and compression are used to reduce the amount of data to acquire and the system is designed for scalable data aggregation and analysis. All systems to be analysed must run the hypervisor and connect to the peer-to-peer network for distributed processing.

In some cases it may not be practical to power off core busniness resources, meaning that evidence has to be acquired on running servers without modification to the infrastructure. Sealey [5] discusses a remote forensic acquisition process on live servers using the Encase Enterprise Edition software from Guidance Software. A servlet application is copied to a device, which then extradites data across the network. This method has the added benefit of not requiring physical access to the machine or to other networked devices.

A key factor with all network based acquisition methods is the bandwidth available for uploading evidence. Existing approaches rely on running software directly on the machine which is being investigated, with data reduction being achieved by pre-processing data locally on the device. In contrast to these approaches, this work examines the case where a common interface is available for data being served from a file server, but does not assume that the examiner has physical access or can execute live programs on the device.

III. Approach

The speed at which forensic evidence can be acquired from a networked device is dependent on the available network throughput, be it the speed of the local network, or the bandwidth of an Internet connection. Reducing the amount of data which is sent over the network will reduce the impact of this bottleneck, potentially resulting in lower processing times. Prior work has used a servlet running on the target machine to pre-process data, which can then send a data subset across the network to the forensics workstation or server. In this work data subsetting is achieved remotely by requesting that only a small fraction of all files is transmitted over the network by the network storage protocol.

A. Sub-file Hashing Strategies

Prior work [20] demonstrates that reading as few as 4096 bytes of data from the end of a file can be unique at the million image scale, while reading a small number of bytes only from the start of the file is less effective. This sub-file approach was also shown to provide considerable speed gains over reading the full file, particularly as file sizes increase. Based on these observations, the work presented below utilises two general sub-file hashing strategies which use data from the end of the file, with data being hashed using SHA256 to produce signatures. The strategies employed are:

[Last n]: Read n bytes from the end of the file. These read blocks are not expected to align with hard disk sectors or SSD pages often (1/4096 of the time, assuming a storage block size of 4096 bytes). Timed benchmarks were conducted for 1–4 data blocks, corresponding to 4KiB, 8KiB, 12KiB and 16KiB, in order to assess the performance degradation caused by reading more data. While previous investigation has shown that 4KiB is sufficient to uniquely identify every unique image in the Flickr 1 Million dataset [20], larger block sizes were tested to demonstrate scalability in scenarios which require more data to generate unique signatures. As this acquisition takes places at the logical level, no slack space is included.

[First n+Last n]: Read n bytes from both the start and the end of the file, resulting in $2n$ bytes of data to hash. The first chunk of data is expected to be block aligned on the storage media, while the last block remains the same as Last n. The value of n used in this work is 4096, meaning that a total of 8KiB of data is required. While being potentially more discriminative than Last n, this technique suffers larger performance penalties, and was not tested with larger block sizes. This technique is referred to as First+Last n for the remainder of this document.

For both techniques, if the file size is smaller than the requested block data the entire file is requested and hashed. Rather than checking file size ahead of time it is faster to catch exceptions when reading files, as this happens infrequently and bypasses the additional overhead for requesting the file size for all files across the network.

B. Datasets

Three datasets were used to evaluate the above sub-file hashing strategies: *i)* The first 25,000 images in the Flickr 1 Million dataset [6] in numerical file name order (0.jpg, 1.jpg, .. 24999.jpg). This dataset is comprised entirely of JPEGs collected from the Flickr image hosting service. *ii)* The Flickr subset converted to the PNG format to create a dataset of larger file sizes, and *iii)* The first 25,000 images of the Govdocs dataset [7] converted to the PNG format, which contains files larger than those in the Flickr PNG set. As Govdocs files contain real file names, and are not numerically ordered, the 25k subset was determined from the file list order provided by the python `os.listdir` function.

Conversion to PNG was facilitated using the Python Pillow library [24]. PNG conversion allowed a variety of file sizes to be explored while still making use of easily obtainable public datasets. The end result is three datasets which roughly double in size for each step up, increasing 2.5× by converting the Flickr set to PNG, with the Govdocs PNG set being 1.8× larger than that. File size statistics for each dataset are given in Table I.

C. Benchmark Set-up

Two experimental scenarios were chosen to reflect typical methods of accessing a file server. *i)* Via a LAN connection, with both a 1Gbps and 100Mbps connections being tested, and *ii)* a VPN connection over the Internet to a remote file server, with a 100Mbps symmetric connection for the client. The LAN server in the first scenario was on an isolated network and connected to the client using a high performance switch, while the Internet server was hosted on a local region Digital Ocean droplet.

Dataset	Size Range	Median	Mean	Total
Flickr (25k Subset)	8KiB–718KiB	112KiB	118KiB	2.81GiB
Flickr PNG (25k subset)	33KiB–846KiB	295KiB	295KiB	7.04GiB
Govdocs PNG (25k Subset)	377B–25MiB	152KiB	535KiB	12.7GiB

TABLE I
File size statistics for the datasets used in this work.

Machine	Specification	Software
Client Workstation	HP EliteDesk 800 G1 i5-4590s, 4GiB RAM	Windows 7 Enterprise 64bit NFS v3 client OpenVPN Client (AES 256/SHA256, no compression, 2048bit key)
LAN Server (Hypervisor)	2× Intel Xeon E5-2697v4 384 GiB RAM RAID 10 10× 840 EVO 1TB SSD	vSphere 6.5.0 ESXi 6.5.0
LAN Virtual Machine	ESXi 6.5+ Virtual Machine 1 Virtual CPU, 2GiB RAM 500GB SSD storage (EXT4)	Ubuntu 17.04 LTS Samba server 4.3.11-ubuntu NFS server 1.2.8-9.2ubuntu2
Internet Server	Digital Ocean $10 Droplet 1 Virtual CPU, 2GiB RAM 50GB SSD storage (EXT4)	Ubuntu 16.04 LTS Samba server 4.5.8-ubuntu NFS server 1:1.2.8-9ubuntu12.1

TABLE II
Specifications of the equipment and software set-up used in this work.

Timed benchmarks were performed for both scenarios using the Samba and NFS file serving technologies in order to assess their potential impact on the performance of sub-file hashing strategies. The impact of file size for each combination was assessed using the three datasets described above (Section III-B). As neither Samba nor NFS provide encrypted transport, a VPN tunnel is required in order to securely access files across the Internet. As such, in order to maintain a realistic access scenario, OpenVPN [25] was used to create a secure connection over the Internet. All software was left with default configurations from a fresh install. Table II provides hardware specifications and software versions for both scenarios. While not all enterprise storage solutions are expected to make use of SSDs, as with the servers in this work, they are expected to have high performance, enabling them to serve many employees simultaneously.

Benchmark code was written in Python 2.7, using the built-in `hashlib` library to generate SHA256 hashes for data blocks. File read orders were determined using Python's `os.listdir` function, and volumes were mounted read only by the client. Reported times do not include file enumeration from `os.listdir`. The mean value of three repeated runs was taken for each set of parameters, with client side memory caches being cleared before each run.

D. *Preserving File Access Times*

As the client device is not interacting with the target file system directly, instead interfacing with a layer of abstraction in the form of a file server, no guarantees can be made that file system metadata will remain unchanged. Despite the Windows client having read only access, the default configurations in this experiment caused file access times to be modified in the EXT4 directories containing the test data. Without access to the server configuration it cannot be verified that modification will not occur. It is therefore recommended that available file metadata be collected prior to remotely hashing files on a file server to prevent the loss of potentially useful forensic information.

IV. Findings

As the sub-file hashing techniques in this paper have previously been shown to produce unique signatures for the full versions of the datasets [20], their discriminating power will not be discussed here. Instead, the focus will be on the performance merits of both approaches. Various aspects of performance are explored in Sections IV-A to IV-C, before summarising the results and relative improvement factors over full file hashing in Section IV-D. The impact of possible follow-up full file confirmation hashing on detected files is then assessed in Section IV-E.

A. *Scaling With Number of Threads*

Multi-threaded file requests can maximise the throughput from an IO device, such that it is always kept busy with concurrent requests. The thread scaling performance of file serving protocols will be limited by the software implementations on both client and server, as well as the physical limitations of the network and underlying physical storage configuration.

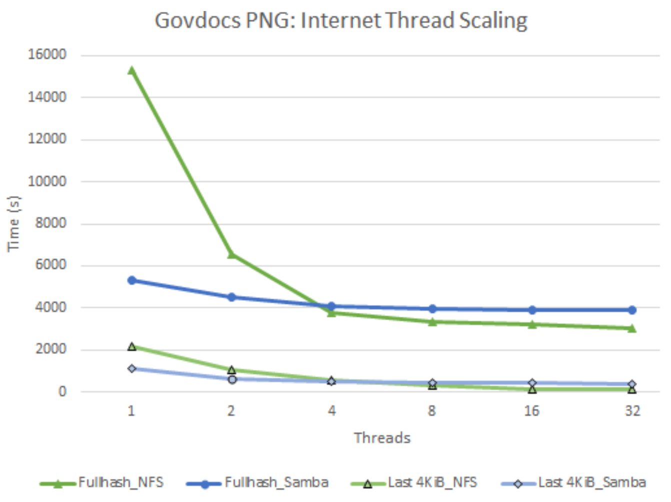

Fig. 1. **The impact of thread count on performance of the Last 4KiB and Fullhash techniques for the Internet File Server. Relative values are representative for all three datasets and LAN configurations.**

Figure 1 shows the thread scaling performance of both NFS and Samba for the Govdocs PNG dataset accessed across the Internet. This graph is representative of the behaviour for all tested connection set-ups and datasets, with Last 4KiB being chosen to represent all sub-file approaches for clarity.

The scaling of NFS is much more pronounced than Samba, with relatively poor performance for 1–2 threads, and surpassing Samba thereafter. Samba does benefit from scaling to 32 threads, however the single threaded nature of the Samba implementation is likely a limiting factor when increasing the number of concurrent file requests, causing relatively small gains. In both cases, the performance scaling of the sub-file approach mimics that of the full file hashing approach, but with less of a performance penalty for low thread counts on NFS.

B. *Performance Scaling With File Size*

One of the benefits of the sub-file signature approach in this work is that their processing costs should be independent of file size. That is, reading a 4KiB chunk of a file should take the same length of time regardless of the size of the file. Figure 2 shows this to be the case for both of the network file storage protocols. This observation also holds for the First+Last 4KiB approach.

Full file hashing appears to scale linearly with file size on networked file systems, which is in line with prior observations for local SSDs [20]. This linear scaling means that the gap in performance between full file hashing and sub-file approaches increases with file size, with the time difference already being substantial at the mean file size of 535KiB for these experiments. For larger files, such as high resolution photos, or even video files, sub-file approaches would tend towards a tiny fraction of the processing time of full file hashing.

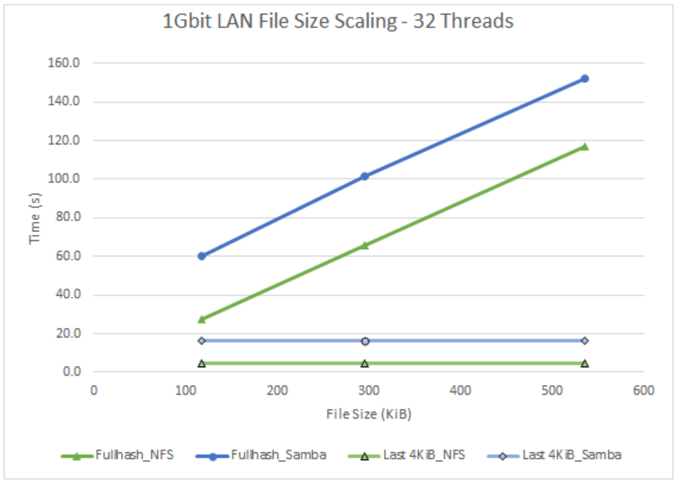

Fig. 2. **The scaling of the 32 thread performance of Last 4KiB and Fullhash across each dataset for the Internet file server. Relative values are representative for LAN configurations.**

This file size scaling effect is amplified as the total throughput of the network is reduced. Going from 1Gbit to 100Mbps on the LAN connection increased full file hashing times by approximately 10×, while Last 4KiB only suffered a 2–3× performance penalty, with a 3–6× penalty for Last 16KiB. This is because the primary limiting factor for the sub-file approaches in this case is the file server itself, rather than the bandwidth of the connection. However, increasing the block size used in the sub-file approaches also has a corresponding bandwidth penalty.

C. Sub-file Hashing Techniques Compared

Figure 3 depicts the Samba processing times for sub-file approaches across each connection type. When bandwidth is not the primary bottleneck, the performance of all sub-file approaches converges at 32 threads, as in Figure 3a. When bandwidth is a limiting factor, as with Figure 3b, the volume of data to be transferred becomes a bottleneck, with each additional 4KiB block in the Last n technique resulting in a small performance penalty. The 8KiB of data hashed by First+Last 4KiB places its performance between Last 8KiB and Last 12KiB, as the Last n data block benefits from contiguity, and therefore an increased sequential read performance.

However, total bandwidth is not the only property of the network connection which seems to have an effect. Rather, the total round trip cost of each transaction, i.e. from the client request to the client receiving the data for each file, appears to impact the overall performance. Figure 3c for the 100Mbit Internet connection shows behaviour more similar to the 1Gbit LAN (3a) scenario than to the equivalent 100Mbit of the LAN connection (3b). A single transaction may involve multiple round trip requests at the file system level, as file handles are opened, and file seeks are performed. As a result, the fraction of the transaction which is attributed to transferring the small data blocks is proportionally less. This difference is not attributable to the underlying storage media as both the LAN and Internet servers were theoretically able to saturate their network connections, with 4KiB random read performance from the SSD storage measured at 1.1Gbit/s and 175.7Mbit/s respectively, using the *iops* tool [26].

The cost of each transaction also explains why the Internet connection is an order of magnitude slower in acquiring data than the 100Mbit LAN scenario, despite having the same theoretical bandwidth.

D. Sub-file Hashing vs. Full File Hashing

A performance summary of the tested sub-file techniques is provided in Table III for the 32 thread configurations. A performance factor of 10 indicates that the technique is 10× faster than the equivalent full file hashing benchmark.

All sub-file approaches prove viable in all scenarios, with gigabit LAN and Internet connection performance being roughly equivalent for all sub-file methods, approximately 3–10× depending on the dataset for Samba, and 5–25× for NFS. However Last 4KiB has the clear advantage when bandwidth is the limiting factor as with the 100Mbit LAN connection, reaching up to 82× on NFS, and 41× on Samba. Increasing the data block size to 16KiB results in roughly half the performance in this scenario, while First+Last 4KiB is typically around 2/3 of the speed.

Sub-file hashing strategies are capable of substantial reductions in the time taken to detect contraband on a networked file server. As the underlying storage performance increases, the network becomes the bottleneck for full file hashing, such that these results should only improve when used with higher performance corporate storage solutions.

E. Impact of Full File Confirmation Hashing

Sub-file approaches are designed for fast contraband detection when dealing with large data volumes, while providing a high degree of accuracy. However, in some cases it may be prudent to perform full file hashing on detected contraband as a verification step. As this step is to eliminate false positives, it is not required for non-contraband files.

Table IV provides projected worst case scenario performance values when performing confirmation hashing with detection rates of 10% and 33% on NFS over a gigabit LAN. These suggested detection rates are likely much higher than the base rate of detection in real cases, but help to offer an insight into the scaling of confirmation hashing. This total time was calculated by adding the appropriate percentage of the Fullhash benchmark time to the processing time of the sub-file technique, and therefore ignores the benefit of caching effects.

The performance impact of confirmation hashing increases with file size, however even with a high detection rate of 33% sub-file processing is still approximately twice as fast as full file hashing. This confirmation step can be done separately from the initial triage phase, allowing for the fast delivery of initial results.

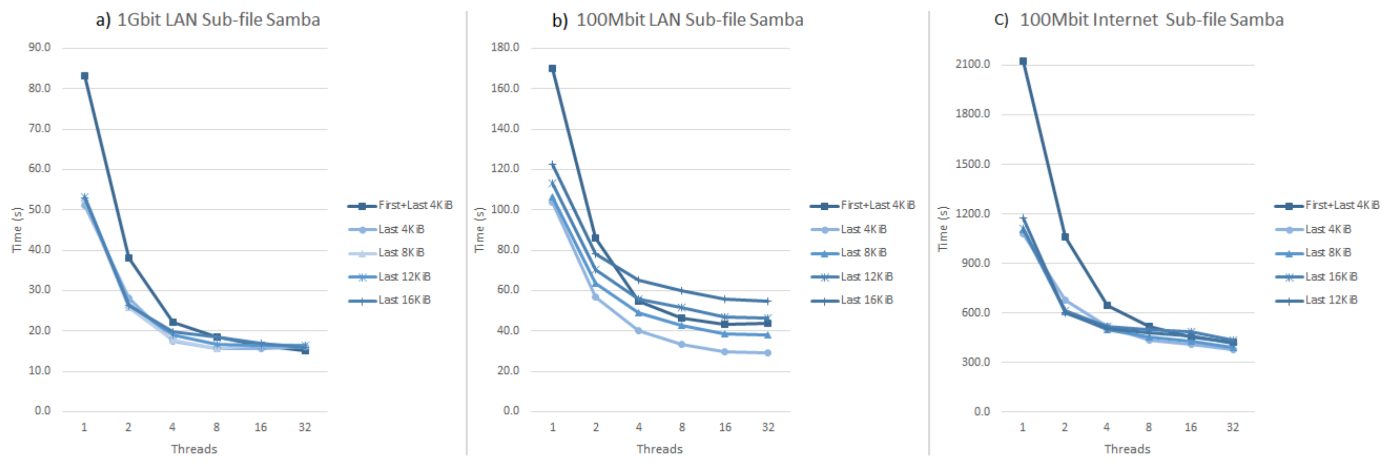

Fig. 3. **Sub-file techniques compared for the Samba protocol across connection types. Behaviour is representative for NFS.**

Dataset	Technique	Samba Performance Factor			NFS Performance Factor		
		1Gbit LAN	100Mbit LAN	100Mbit Internet	1Gbit LAN	100Mbit LAN	100Mbit Internet
Flickr	Last 4KiB	3.72	10.20	3.13	6.18	18.29	6.77
	Last 16KiB	3.75	5.42	2.75	4.23	6.77	5.07
	First+Last 4KiB	3.95	6.81	2.79	4.94	13.06	7.55
Flickr PNG	Last 4KiB	6.31	23.45	6.13	14.84	45.54	14.75
	Last 16KiB	6.37	12.46	5.25	10.17	16.84	10.30
	First+Last 4KiB	6.70	15.64	5.49	11.87	32.51	16.39
Govdocs PNG	Last 4KiB	9.44	41.62	10.22	26.43	82.37	24.27
	Last 16KiB	9.52	22.10	8.99	18.10	30.46	18.37
	First+Last 4KiB	10.01	27.76	9.14	21.13	58.81	27.79

TABLE III
A TABLE OF RELATIVE PERFORMANCE FACTORS TO FULLHASH FOR EACH TECHNIQUE. VALUES PRESENTED ARE FOR 32 THREADS.

1Gbit LAN NFS		Performance Factor		
Technique	**Hit Rate**	Flickr	Flickr PNG	Govdoccs PNG
Last 4KiB	0%	6.18	14.84	26.43
	10%	3.82	5.97	7.25
	33.3%	2.02	2.50	2.70
Last 16KiB	0%	4.23	10.17	18.10
	10%	2.97	5.04	6.44
	33.3%	1.76	2.32	2.58
First+Last 4KiB	0%	4.94	11.87	21.13
	10%	3.31	5.43	6.79
	33.3%	1.87	2.40	2.63

TABLE IV
THE IMPACT OF CONFIRMING DETECTED CONTRABAND WITH FULL FILE HASHING AT DETECTION RATES OF 10% AND 33.3%. VALUES ARE ONLY SHOWN FOR NFS ON THE 1GBIT LAN CONNECTION.

V. CONCLUSIONS AND FUTURE WORK

This work demonstrates that sub-file hashing strategies can be used to rapidly investigate remote networked storage. Experiments were performed on Samba and NFS servers over the Internet and two LAN configurations, showing up to an 82× performance increase over full file hashing on standard resolution image files. Sub-file techniques were shown to perform better on NFS than Samba, with limited bandwidth over a 100Mbit LAN creating a dramatic gulf in performance between sub-file and full file hashing approaches.

The physical storage media is typically the performance bottleneck for forensic processing. However, in a world where large quantities of data are being stored remotely on cloud services, network performance may present itself as a frequent bottleneck in the forensic process. Sub-file hashing can be used to greatly decrease the time taken to perform an investigation over a network, which is particularly important when dealing with hundreds of terabytes of data on large scale storage networks.

Future work can investigate the possibility of applying sub-file approaches to commercial cloud storage providers, such as Dropbox and One Drive. Sub-file hashing approaches offer a great deal of potential to reduce forensics processing times over a network, with many possible network and server configurations to explore. This approach may also have the potential to be applied to large scale file de-duplication, which has similar characteristics to contraband detection.

ACKNOWLEDGEMENT

This research was supported by a scholarship provided by Peter KK Lee. We are also grateful to Peter Aaby for access provided to the LAN server used in this work.

References

[1] D. Quick and K.-K. R. Choo, "Impacts of increasing volume of digital forensic data: A survey and future research challenges," *Digital Investigation*, vol. 11, no. 4, pp. 273–294, Dec. 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287614001066

[2] V. N. Franqueira, J. Bryce, N. Al Mutawa, and A. Marrington, "Investigation of Indecent Images of Children cases: Challenges and suggestions collected from the trenches," *Digital Investigation*, Dec. 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287617302669

[3] D. Lillis, B. Becker, T. O'Sullivan, and M. Scanlon, "Current Challenges and Future Research Areas for Digital Forensic Investigation," *arXiv:1604.03850 [cs]*, Apr. 2016, arXiv: 1604.03850. [Online]. Available: http://arxiv.org/abs/1604.03850

[4] J. Saarinen, "FBI Ordered to Copy 150 Terabytes of Data Seized From Megaupload," 2012. [Online]. Available: https://www.wired.com/2012/06/megaupoad-data/

[5] P. Sealey, "Remote forensics," *Digital Investigation*, vol. 1, no. 4, pp. 261–265, Dec. 2004. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287604000854

[6] M. J. Huiskes, B. Thomee, and M. S. Lew, "New trends and ideas in visual concept detection: the MIR flickr retrieval evaluation initiative," in *Proceedings of the international conference on Multimedia information retrieval*. ACM, 2010, pp. 527–536. [Online]. Available: http://dl.acm.org/citation.cfm?id=1743475

[7] S. Garfinkel, P. Farrell, V. Roussev, and G. Dinolt, "Bringing science to digital forensics with standardized forensic corpora," *Digital Investigation*, vol. 6, pp. S2–S11, Sep. 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287609000346

[8] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," *Digital Investigation*, vol. 3, Supplement, pp. 91–97, Sep. 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1742287606000764

[9] V. Roussev, G. G. Richard, and L. Marziale, "Multi-resolution similarity hashing," *Digital Investigation*, vol. 4, pp. 105–113, Sep. 2007. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287607000473

[10] V. Roussev, "Data fingerprinting with similarity digests," in *IFIP International Conference on Digital Forensics*. Springer, 2010, pp. 207–226.

[11] A. Hadmi, A. A. Ouahman, B. A. E. Said, and W. Puech, *Perceptual image hashing*. INTECH Open Access Publisher, 2012. [Online]. Available: http://cdn.intechopen.com/pdfs/36921/InTech-Perceptual_image_hashing.pdf

[12] F. Breitinger, H. Liu, C. Winter, H. Baier, A. Rybalchenko, and M. Steinebach, "Towards a process model for hash functions in digital forensics," in *International Conference on Digital Forensics and Cyber Crime*. Springer, 2013, pp. 170–186.

[13] V. Roussev, C. Quates, and R. Martell, "Real-time digital forensics and triage," *Digital Investigation*, vol. 10, no. 2, pp. 158–167, Sep. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287613000091

[14] S. L. Garfinkel, "Digital media triage with bulk data analysis and bulk_extractor," *Computers & Security*, vol. 32, pp. 56–72, Feb. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167404812001472

[15] J. Grier and G. G. Richard, "Rapid forensic imaging of large disks with sifting collectors," *Digital Investigation*, vol. 14, pp. S34–S44, Aug. 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287615000511

[16] P. Penrose, W. J. Buchanan, and R. Macfarlane, "Fast contraband detection in large capacity disk drives," *Digital Investigation*, vol. 12, Supplement 1, pp. S22–S29, Mar. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1742287615000080

[17] S. Garfinkel, A. Nelson, D. White, and V. Roussev, "Using purpose-built functions and block hashes to enable small block and sub-file forensics," *Digital Investigation*, vol. 7, pp. S13–S23, Aug. 2010. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287610000307

[18] S. McKeown, G. Russell, and P. Leimich, "Fast Filtering of Known PNG Files Using Early File Features," in *Annual ADFSL Conference on Digital Forensics, Security and Law*, Daytona Beach, Florida, USA, 2017. [Online]. Available: https://commons.erau.edu/adfsl/2017/papers/1/

[19] S. McKeown, G. Russell, and P. Leimich, "Fingerprinting JPEGs With Optimised Huffman Tables," *Journal of Digital Forensics, Security and Law*, In Press.

[20] S. McKeown, G. Russell, and P. Leimich, "Sub-file Hashing Strategies for Fast Contraband Detection," in *International Conference on Cyber Security and Protection of Digital Services (Cyber Security 2018)*. Glasgow, UK: IEEE, Jun. 2018.

[21] M. Scanlon and M.-T. Kechadi, "Online Acquisition of Digital Forensic Evidence," in *Digital Forensics and Cyber Crime*, S. Goel, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 31, pp. 122–131, dOI: 10.1007/978-3-642-11534-9_12. [Online]. Available: http://link.springer.com/10.1007/978-3-642-11534-9_12

[22] M. B. Koopmans and J. I. James, "Automated network triage," *Digital Investigation*, vol. 10, no. 2, pp. 129–137, Sep. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1742287613000273

[23] I. Homem, S. Dosis, and O. Popov, "LEIA: The Live Evidence Information Aggregator: Towards efficient cyber-law enforcement," in *Internet Security (WorldCIS), 2013 World Congress on*. IEEE, 2013, pp. 156–161.

[24] A. Clark, "Python-Pillow," 2015. [Online]. Available: https://github.com/python-pillow/Pillow

[25] J. Yonan, "OpenVPN - Open Source VPN," 2002. [Online]. Available: https://openvpn.net/

[26] B. Schweizer, "iops: Benchmark disk IOs," Feb. 2010, original-date: 2011-09-03T13:08:48Z. [Online]. Available: https://github.com/cxcv/iops