Global Journal of Computer Science and Technology: E Network, Web & Security
Volume 20 Issue 4 Version 1.0 Year 2020
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 ISSN Print 0975-4350

Vanguard: A Blockchain-based Solution to Digital Piracy

By Kavinga Yapa Abeywardena, Tharika Munasinghe, Yasiru Jayasinghe, Sumala Mannage, Thisuri Warnasooriya & Gihan Edirisinghe

Sri Lanka Institute of Information Technology

Abstract- Online pirates and Intellectual Property (IP) holders have been in perpetual war over various products like music, movies, software, etc. since the popularity of the Internet. It is estimated that the US entertainment industry loses approximately 29 billion USD every year for pirates. Online piracy has since gone from bad to worse as growing internet users and better broadband connections enable people to share large files freely over the internet. The objective of this research is to investigate the causes and enablers for online piracy in movie industry and to come up with an anti-pirating solution. The primary outcome of the study will consist of a dedicated blockchain based anti-piracy system, 'Vanguard'. This system will provide all-round piracy protection from a built-in streaming service to a component to actively look through the internet for pirated movies and torrents. This system will greatly deter the piracy of movies since the IP holders can ensure their IP rights through this system and quickly act against illegitimate distribution of their media.

Keywords: blockchain, streaming service, intellectual property, image matching, string matching, deep-learning.

GJCST-E Classification: K.6.5

Strictly as per the compliance and regulations of:

Vanguard: A Blockchain-based Solution to Digital Piracy

Kavinga Yapa Abeywardena [α], Tharika Munasinghe [σ], Yasiru Jayasinghe [ρ], Sumala Mannage [ω], Thisuri Warnasooriya [¥] & Gihan Edirisinghe [§]

Abstract- **Online pirates and Intellectual Property (IP) holders have been in perpetual war over various products like music, movies, software, etc. since the popularity of the Internet. It is estimated that the US entertainment industry loses approximately 29 billion USD every year for pirates. Online piracy has since gone from bad to worse as growing internet users and better broadband connections enable people to share large files freely over the internet. The objective of this research is to investigate the causes and enablers for online piracy in movie industry and to come up with an anti-pirating solution. The primary outcome of the study will consist of a dedicated blockchain based anti-piracy system, 'Vanguard'. This system will provide all-round piracy protection from a built-in streaming service to a component to actively look through the internet for pirated movies and torrents. This system will greatly deter the piracy of movies since the IP holders can ensure their IP rights through this system and quickly act against illegitimate distribution of their media.**

Keywords: blockchain, streaming service, intellectual property, image matching, string matching, deep-learning.

I. Introduction

Intellectual property refers to creations of the mind. This ranges from inventions, literary and artistic works, symbols, names and images used in commerce [1]. A motion picture copyright protects the artistic expression in movies, short films, and videos, including the camera work, dialogue, and sounds [2]. It does not protect idea of the movie or characters showed in it.

The first well-known court case on copyright infringement in the movie industry was Nichols v. Universal Pictures Corporation court case. The case was won by the defendant due to the court deciding that copyright protection does not include ordinary characters in a story.

The first ever concept of streaming was brought to light when physicists were able to develop a method to transmit information between two places without the use of wires [3]. This resulted in the possibility of transmitting radio signals over long distances. Now, in the 21st century, streaming majorly involves live broadcasts though the internet. The technology which began in the early 1990s has now been revolutionized by companies such as Netflix, Spotify, YouTube, Hulu and Pandora [3], which are services that offer television shows, music, movies, sports and games directly to your computer.

a) Research Problem

i. IP Protection Problem

According to United States copyright office annual report in fiscal 2018, the Office registered 560,013 claims for registration and recorded nearly 21,700 documents containing titles of more than 757,400 works [5]. At the same time according to the Motion Picture Association of America (MPAA), there are approximately 150,000 movie screens in the world, with about 40,000 of those in the United States alone. About 560 films were released at the cinema in the United States in 2010 – about 11.5 every week. Of those, 419 were independent films [6].

By analyzing these figures, it is clear that ensuring IP rights of each movie is not an easy task. Among all IP rights, copyrights play a major role in movie industry and copyright law demand chain of documentation. Maintaining these required documents to match copyright law, from the beginning till the end while reflecting chain of rights of the title holders is a complex task. Even with every necessary document, individuals are not able to obtain their rights automatically.

ii. The Increase of Digital Piracy

The rapid development of the internet in recent years brought about a large increase in services that provide media streaming for consumers. This also brought new ways for pirates to distribute media through illegal channels, and for users to access those channels more easily and at a much larger scale [4].

About 34% of all recorded music products sold worldwide in 2004 were pirated copies and that piracy costs the industry over $4.6 billion per annum [7]. Research by Digital TV Europe states that revenue losses for the television and film industries as a result of piracy could reach $52 billion by 2022, as shown in Figure 1 below [8].

Author α: Department of Computer Systems Engineering, Sri Lanka Institute of Information Technology Malabe, Sri Lanka.
e-mail: kavinga.y@sliit.lk
Author σ ρ ω: Department of Information Systems Engineering, Sri Lanka Institute of Information Technology Malabe, Sri Lanka.
e-mails: tharika.m@sliit.lk, yasirujayasinghe@gmail.com, desilvasonali4@gmail.com
Author ¥ §: Department of Computer Science and Software Engineering, Sri Lanka Institute of Information Technology, Malabe, Sri Lanka. e-mails: thisuriwarnasooriya@gmail.com, gihanprabash@gmail.com

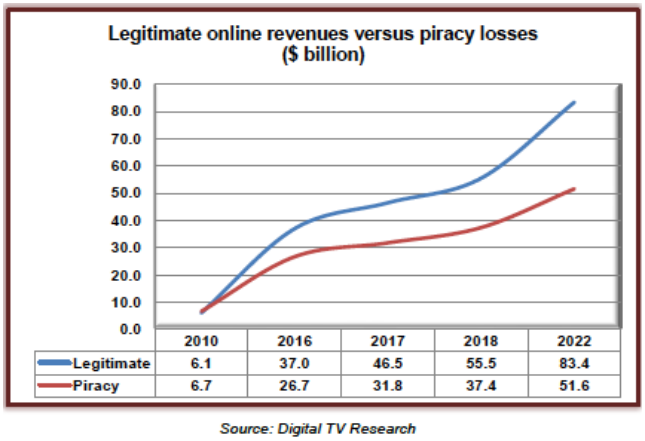


Fig. 1: Estimate of revenue losses due to piracy by year 2022

b) Research Gap

i. *Current Solutions*

Over the years there have been many attempts and researches about more robust IP protection and deterring digital piracy, through means such as DRM (Digital Rights Management) takedowns, advertising models etc. Some such solutions are Bernstein [9][10], which is solution that combines blockchain technology and IPFS, and the Traditional Copyright Process [2], which requires the submission of several documents throughout the movie creation process. After submitting all the documents, the process will go through several steps for around three to ten months, before finally resulting in the copyright certificate.

As solutions against piracy, the Advertising model [11] and other unique approaches such as iTunes' Matching Service [11] act as alternatives to piracy. On the other hand, Custos Screener Copy [12] is a blockchain-based solution that uses forensic watermarking technology to track content.This solution uses the decentralized nature of blockchain and uses peers of the network to hunt down copies of the media.

ii. *Research Gap*

a. *Additional solutions for IP protection*

Although there are many existing technologies and solutions that try to address the management of IP rights, there are a few key areas that can be used to specifically enhance the IP rights of movie industry using blockchain technology.

- IP right enforcement agencies are overloaded with work and the process takes too long [13]
- Difficulties in enforcing Licensing Agreements [14]

In the proposed solution, by using blockchain technology combined with smart contracts, it can fully automate the copyright process through obtaining and verifying documents with each milestone of copyright process during the movie creation process. At the end of the movie production, the producer can obtain their copyright ownership certificate. It can also be used to automatically enforce rights and using pre-define rules for royalties to be properly allocated and distributed.

b. *Additional solutions for deterring piracy*

The Advertising Model and Matching Service solutions [11] mentioned above do not attempt to actively address the digital piracy issue. Instead, they aim to provide alternatives for consumers to make them less likely to consume media through illegitimate channels.

Custos Screener Copy [12] on the other hand, is a product that directly aims to deter piracy of media. There are two ways in which Custos Tech's solution can be improved, they are:

- Creating a more robust forensic watermarking system
- Automating the watermark extraction and comparison process

CustosTech's solution to finding pirated media depends on human miners. This makes the process somewhat unreliable as they will not always be searching for a specific creator's media. Vanguard aims to automate this process, to assure the creator that the system will be continuously searching for illegal copies of their media.

The table below summarizes the additional features that will be introduced by Vanguard.

Table 1: Comparison of Current Solutions With Vanguard

Product	Define milestones of copyright process	Chain of custody	Unchangeable data	Actively deters media piracy
Traditional Copyright Process	✓	✗	✗	✗
Blockchain based solution (Bernstein)	✗	✓	✓	✗
iTunes Match Service	✗	✗	✗	✗
Custos Screener Copy	✗	✗	✓	✓
Vanguard	✓	✓	✓	✓

II. Methods

a) Objectives and Design

Figure 2 below depicts the overview of all processes under the Vanguard Piracy Protection Suite. The diagram depicts all four sub-components under Vanguard, and how data flows through each of them.

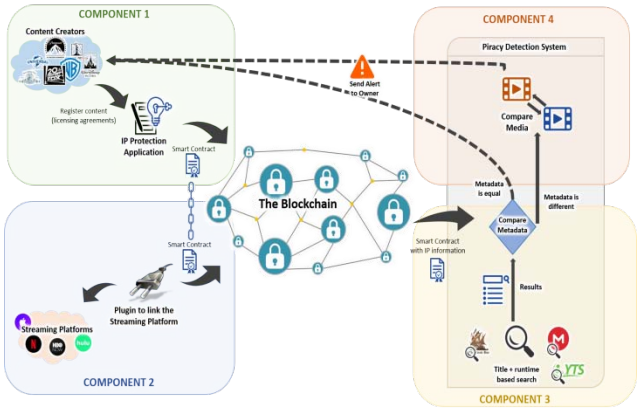


Fig. 2: Vanguard System Overview

i. *IP Protection Component Design*

The diagram below (Figure 3) represents the system overview for Vanguard's IP Protection component. First, the client enters registration details to create their account, which is saved in a smart contract in the blockchain. After the client has created their account, they are able to start the media licensing process. For this process, the client must upload all necessary documents during the time of producing the media, and the system will save time-stamped records of the documents in the blockchain using smart contracts. The system will compare the documents with any previously uploaded documents to assure the originality of the documentation process.

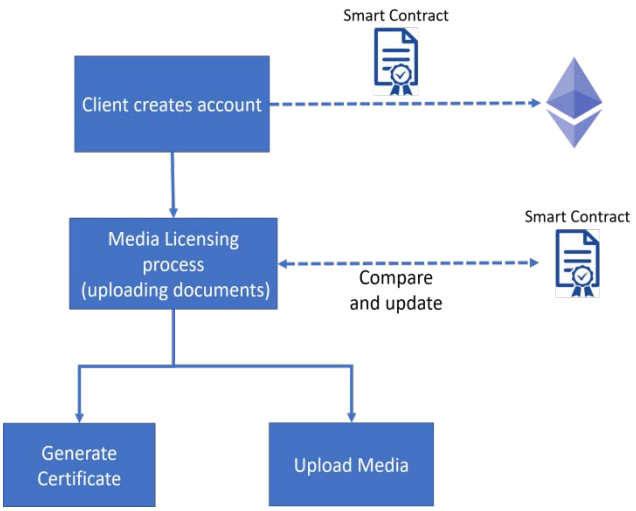


Fig. 3: Vanguard IP Protection Component Overview

After the media production process is complete, all records and documents will exist as tamper proof records in the blockchain. The client must then upload the final media file to Vanguard, which will be used for Vanguard's Streaming Service component (component 2), whenever the client chooses. Finally, the client can generate a fully digitalized certificate that proves their IP rights to the Media product, which they can use in cases of IP violations to prove their rights.

ii. *Streaming Service Component Design*

The diagram below (Figure 4) represents the system overview for Vanguard's Streaming Service component. To access this service, the client must already have a registered account with Vanguard through Component 1.

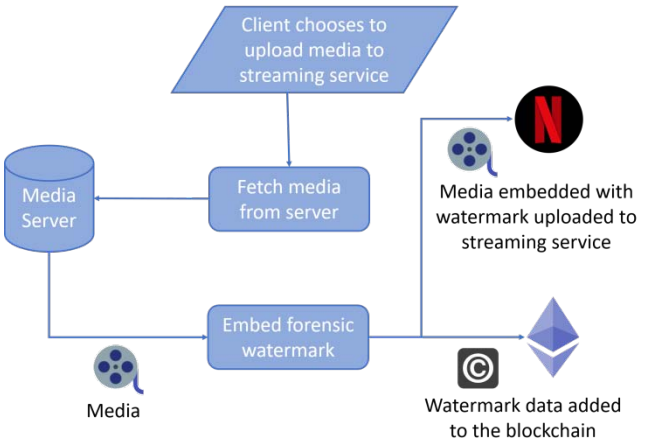


Fig. 4: Vanguard Streaming Component Overview

When the client wishes to upload their media to the streaming service, the system will first fetch the media from the blockchain. The system will then generate the traceable forensic watermark that contains details about where it is being uploaded to, the original content owner etc. The generated forensic watermark will then be embedded into the media in a way that cannot be easily removed by pirates, while remaining imperceptible to the human eye. Finally, the watermarked media file is uploaded to the streaming service and will be available to stream for consumers, and the data contained in the watermark will also be added to the blockchain for future reference.

iii. *Piracy Search Algorithm Component Design*

Shown below is the overview diagram of the search function of the Vanguard suite. This component will be sold separately to customers as a paid package. The customer should have bought the IP Protection component as a pre-requisite. Whenever a client signs a contract with the Vanguard system and uploads a movie to the system, it's metadata will be saved in a blockchain contract. The algorithm will fetch this data from the blockchain contracts and save it in its memory which is on the media server.

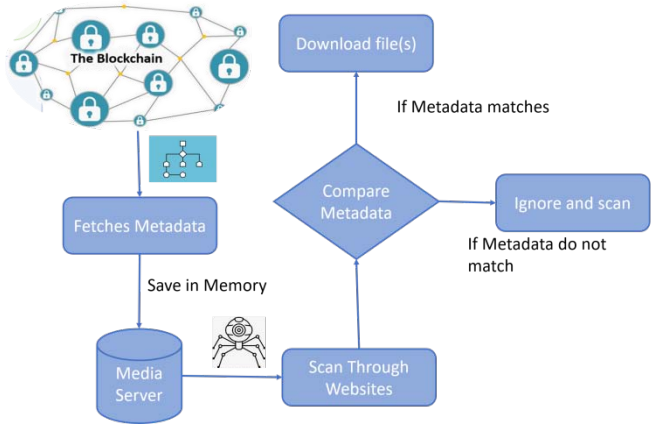


Fig. 5: Vanguard searching algorithm overview

The searching algorithm will then go through a list of previously determined websites and will try to match the titles of the movies in its memory with the results through the website. Once it compares the metadata and has decided that it has found a copy of a movie that has been pirated, it will download that file. If not, it will move on to a different website.

iv. *Video Copy Detection Component Design*

The proposed solution is a system with watermark comparing, image matching and motion and audio comparing. As described in figure 6, the system first extracts the watermark from the downloaded video and compares it with the original watermark, if it matches, the system sends an alert to the IP holder and they can take a relevant action against the pirated video.

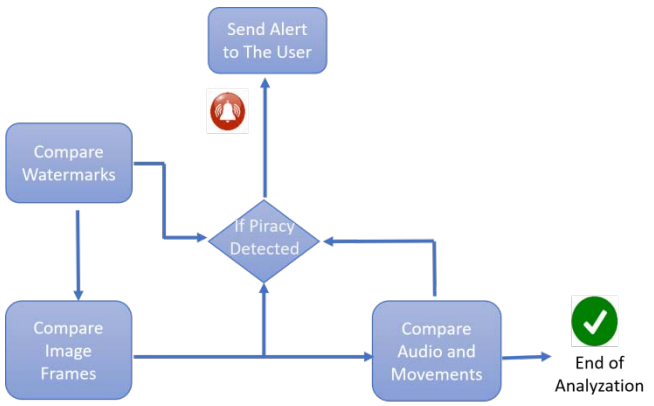


Fig. 6: Vanguard Video Copy Detection Overview

If the system cannot detect any matching watermark, then it will compare videos based on frame by frame image matching and will calculate the similarity percentage. If the similarity percentage exceeds the prescribed limit, the IP holder will be alerted. If not, as the final process the system will detect details that can't be detected from image frame matching. The most common distorts done by pirate uploaders are color unbalancing and brightness changing. The number of motions within two frames and audio files of the videos avoid those. So that the system will compare the similarities through them. This process will continue until all the downloaded videos from the search results are analyzed.

b) Implementation

i. *Media Licensing*

Vanguard's IP Protection (component 1) is responsible for developing a platform that specifically addresses movie copyrights. As the first step of implementation, the development environment is setup by combining a few different technologies. The Ganache Personal Blockchain is used as a local development of blockchain which has similar behaviors to public blockchain. This is equipped with 10 accounts that are credited with 100 ether, which allows to develop applications, deploy smart contracts and run tests. Node Package Manageris used to configure the environment for developing smart contracts. Next, the Truffle Frameworkis installed, which provides functionalities to develop and deploy smart contracts, which can run on Ethereum Virtual Machine and in an any public Ethereum blockchain network, and is able to write tests against smart contracts before uploading to the blockchain and support client-side development. Finally, the browser extension Metamask Ethereum Wallet is installed to connect to blockchain network through the browser.

a. *Milestone Checklist*

One of the main problems identified through research gap is that not enough information has been collected during the movie development process and it is difficult to prove the originality of the work. A Milestone checklist is developed to address this problem through requesting essential documents that movie producers should upload in the traditional copyright process to obtain the certificate. This checklist is developed through smart contracts using Solidity programming language and deployed to the Ganache Ethereum blockchain to ensure the integrity of the certificate.

b. *Uploading documents to IPFS*

```
// Add file to IPFS and return a CID
async saveToIpfs(files) {
  const web3 = window.web3
  console.log('file captured', files)
  console.log('file captured A')
  const source = ipfs.add(
    [...files],
    {
      progress: (prog) => console.log(`received: ${prog}`)
    }
  )
  try {
    let ipfsId = await (source)
    console.log('Results:', source)
    this.setState({ added_file_hash: ipfsId.path })

    for (var i = 0; i < files.length; i++) {
      var Name = files[i].name
    }
    this.setState({ file_name: Name })
    let ipfshash = ipfsId.path

    let timestamp = Math.floor(Date.now() / 1000).toString()
    this.setState({ file_timeStamp: timestamp })

    if (Name !== "" && timestamp !== "" && ipfshash !== "") {
      const data = await this.state.contract.methods.add(web3.utils.asciiToHex(ipfshash),
        Name, timestamp).send({ from: this.state.account });

      return data
    }
```

Fig. 7: Funtions that captures the user files and convert it to buffer

Because storing large files on blockchain costsa large amount of gas, uploading documents to the blockchain is identified as expensive. The Interplanetary File System (IPFS) is recognized as an ideal solution. When connecting to the IPFS, Infura which is a hosted Ethereum node is used, rather than running an IPFS daemon on the host due to it being identified as faster when uploading documents. As the figure shows, the program captures the user file, converts it to a buffer, sends it to IPFS, and returns a hash.

c. *Comparison and Updating the Blockchain*

The hash received from the IPFS is uploaded to the blockchain using HashStorage.sol smart contract. When uploading the hash, as the figure shows it captures the timestamp of the file hash added to the blockchain. This file also compares and checks the availability of the hash in the blockchain

```
function add(
    string memory _ipfshash,
    string memory _filename,
    uint256 _dateAdded
) public onlyOwner {
    count++;
    require(
        collection[_ipfshash].exist == false,
        "this hash already exists in contract"
    );
    DocInfo memory docInfo = DocInfo(_ipfshash, _dateAdded, true);
    collection[_ipfshash] = docInfo;
    emit HashAdded(_ipfshash, _filename, _dateAdded);
}
```

Fig. 8: Comparision of the hashes and genataion of Timestamp

Ownerable.sol smart contract uses the Open Zeppelin library to identify the owner of the account and ensures that only the owner can add files to the blockchain.

ii. *Forensic watermark*

The generation and embedding of the forensic watermark are done automatically by Vanguard's Streaming Service Component. The component is implemented using the Python programming language. The initial steps of creating this process required a way to hide a message inside a single frame of a video. The python library Numpy was used to represent a 3-dimensional array, each dimension representing the row, column, and color, which combine to form a single pixel of the video frame (figure 9). The watermark message is taken as an input and its binary form is computed and appended into the 3-dimensional array, and finally concatenated with the image pixels to embed the data.

```
def embedding_info(picname, savename, text):
    text += '#%#'  # As end flag
    try:
        im = np.array(Image.open(picname))
    except:
        print("Cannot obtain image, please check file name")
        time.sleep(3)
        sys.exit()

    rows, columns, colors = im.shape
    embed = []
    for c in text:
        bin_sign = (bin(ord(c))[2:]).zfill(16)
        for i in range(16):
            embed.append(int(bin_sign[i]))

    count = 0
    for row in range(rows):
        for col in range(columns):
            for color in range(colors):
                if count < len(embed):
                    im[row][col][color] = im[row][col][color] // 2 * 2 + embed[count]
                    count += 1

    Image.fromarray(im).save(savename)
```

Fig. 9: Using Numpyto embed a Forensic Watermark

a. *Attention Model*

One main disadvantage observed through the above approach is that any form of transformation of the watermarked frame causes the watermark to be destroyed. Prior researches such as RivaGAN [15] proved how deep learning-based approaches can be used to embed more robust watermarks in media. Following this approach, Vanguard's watermarking system uses an Attention-based model[16]to learn the probability distribution over data dimensions for each pixel.

The Attention model is equipped with a neural network with the ability to focus on different subsets of its inputs [16]. In the case of a video frame, it has the ability to study the frame and identify different objects and textures within the frame. The data gathered through the Attention model is used to generate an Attention mask, which is used to determine which bits to embed the watermark into during the embedding process.

b. *Training the Model*

To improve the performance of the training model, Adam Optimizeris used. It is easy to implement, more computationally efficient, requires little memory space, and works well on problems with noisy or sparse gradients.

c. *Main Functions*

Some of the most important functions of Vanguard's watermarking system is the Attention model, Encoder, and Decoder functions. For the Attention model, the Pytorch library is used to create the attention mask on a tensor level. This was chosen over Tensorflow because of how well it integrates with Python.

In the Encoder and Decoder functions, Open CV library is used to obtain and manipulate frames from video files (Figure 10). The generated Attention mask is used by the Encoder function to determine which bits to pay attention to at each pixel, and the Encoder function computes a compact form of the data tensor which includes the watermark and concatenates it with the frame. The reverse operation is done by the Decoder function to retrieve the watermark from the frame.

```
def encode(self, video_in, data, video_out):
    assert len(data) == self.data_dim

    video_in = cv2.VideoCapture(video_in)
    width = int(video_in.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(video_in.get(cv2.CAP_PROP_FRAME_HEIGHT))
    length = int(video_in.get(cv2.CAP_PROP_FRAME_COUNT))

    data = torch.FloatTensor([data]).cuda()
    video_out = cv2.VideoWriter(
        video_out, cv2.VideoWriter_fourcc(*'mp4v'), 20.0, (width, height))

    for i in tqdm(range(length)):
        ok, frame = video_in.read()
        frame = torch.FloatTensor([frame]) / 127.5 - 1.0      # (1, H, W, 3)
        frame = frame.permute(3, 0, 1, 2).unsqueeze(0).cuda()  # (1, 3, 1, H, W)
        wm_frame = self.encoder(frame, data)                   # (1, 3, 1, H, W)
        wm_frame = torch.clamp(wm_frame, min=-1.0, max=1.0)
        wm_frame = (
            (wm_frame[0, :, 0, :, :].permute(1, 2, 0) + 1.0) * 127.5
        ).detach().cpu().numpy().astype("uint8")
        video_out.write(wm_frame)

    video_out.release()
```

Fig. 10: Watermarking system encode function

d. *Adding Resilience to the watermark*

In order to increase the resilience of the watermark against various forms of transforms, noise layers are added to the video before it undergoes the embedding process. The implemented watermarking system is currently resistant to scaling and cropping of the video.

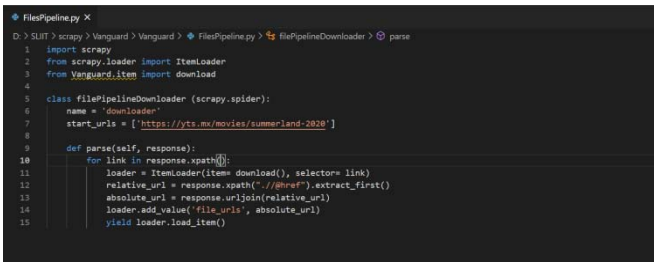
The scaling layer re-scales the video to a random size between 80-100% of the height and width of the original. This allows the system to learn to embed the watermark in a scale-invariant manner. Similarly, the cropping layer selects a sub-window of 80-100% of the original video's height and width randomly and uses that sub-window to embed the watermark.

iii. *Web Crawler Algorithm*

The web crawler component of the Vanguard system is focused on detecting any pirated copies of the movies that has a contract with the system. It is designed to search through known and new torrenting websites and scan through for any offending files. This component is implemented using a framework called 'Scrapy'. This is a python framework that allows the user to fine tune a web crawler to their personal needs. Scrapy allows the user to download a file that matches any of the criteria given to the web crawler. The name and other metadata of the movie will be fetched from the Blockchain database which hold all contract information.

a. *Detecting Web Pages*

The Scrapy framework provides the user with a class called a 'Spider Class'. This class defines the rules and boundaries of the web crawler that is being made. The websites the web crawler needs to visit is defined in this class. The spider class takes the URL of the website that we are defining, and the framework will use this to make a request to the website to scan through it.



FilesPipeline.py

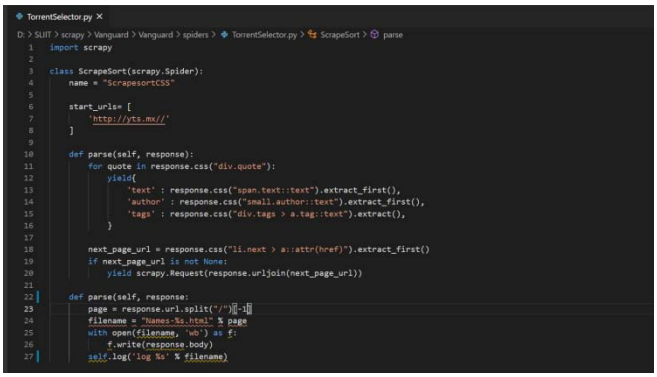
D: > SLIIT > scrapy > Vanguard > Vanguard > FilesPipeline.py > filePipelineDownloader > parse

```
import scrapy
from scrapy.loader import ItemLoader
from Vanguard.item import download

class filePipelineDownloader (scrapy.spider):
    name = 'downloader'
    start_urls = ['https://yts.mx/movies/summerland-2020']

    def parse(self, response):
        for link in response.xpath():
            loader = ItemLoader(item= download(), selector= link)
            relative_url = response.xpath(".//@href").extract_first()
            absolute_url = response.urljoin(relative_url)
            loader.add_value('file_urls', absolute_url)
            yield loader.load_item()
```

Fig. 11: The URLs that are defined in the Spider Class

b. *Matching Content*

The Scrapy framework scrolls through the given webpages in a depth-first order. Which means the algorithm starts at the inner-most page of the website and scans outwards to ensure that no page has been missed. The filenames that are passed onto the framework via the media server will then be compared. Scrapy uses a mechanism known as Selectors to extract data from the website. Selectors work by converting the source of the website into HTML source code and then scanning through them. The name of the content that we want to match is given through a response.css function to the framework.



TorrentSelector.py

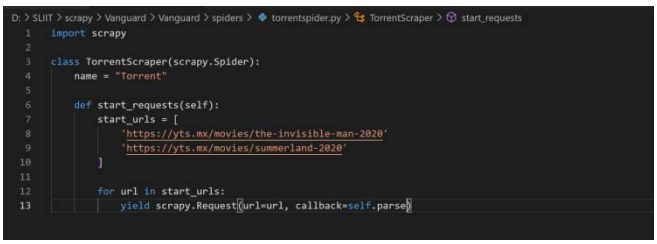
D: > SLIIT > scrapy > Vanguard > Vanguard > spiders > TorrentSelector.py > ScrapeSort > parse

```
import scrapy

class ScrapeSort(scrapy.Spider):
    name = "ScrapesortCSS"

    start_urls= [
        'http://yts.mx//'
    ]

    def parse(self, response):
        for quote in response.css("div.quote"):
            yield{
                'text' : response.css("span.text::text").extract_first(),
                'author' : response.css("small.author::text").extract_first(),
                'tags' : response.css("div.tags > a.tag::text").extract(),
            }

        next_page_url = response.css("li.next > a::attr(href)").extract_first()
        if next_page_url is not None:
            yield scrapy.Request(response.urljoin(next_page_url))

    def parse(self, response):
        page = response.url.split("/")[-1]
        filename = "Names-%s.html" % page
        with open(filename, 'wb') as f:
            f.write(response.body)
        self.log('log %s' % filename)
```

Fig. 12: Scraper Sort class that looks through specific content

c. *Downloding Content*

Once the framework finds a filename that matches any of the titles of the movies that the Vanguard system has a contract with, the framework will automatically download it. The Scrapy framework uses a mechanism known as a Files-pipeline. Once the files-pipeline detects the division of the webpage, in which the link to the file is, it goes to the division and downloads the file. The Files-pipeline lets the user configure which file type can be downloaded.



D: > SLIIT > scrapy > Vanguard > Vanguard > spiders > torrentspider.py > TorrentScraper > start_requests

```
import scrapy

class TorrentScraper(scrapy.Spider):
    name = "Torrent"

    def start_requests(self):
        start_urls = [
            'https://yts.mx/movies/the-invisible-man-2020'
            'https://yts.mx/movies/summerland-2020'
        ]

        for url in start_urls:
            yield scrapy.Request(url=url, callback=self.parse)
```

Fig. 13: The files-pipeline class of the framework

iv. *Copy Detection*

a. *Using OpenCV*

OpenCV (Open Source Computer Vision Library) has more than 2500 optimized algorithms. These algorithms can be used to detect objects, track movements, etc. Throughout the implementation of the system, OpenCV library is used with python to process videos to detect pirate movies.

b. *Converting Videos to Image Frames*

Before being compared through watermarks and image frames, the videos should be converted into image frames. Image frames of every 0.5 seconds should be captured.

c. *Watermark Comparison*

The original watermark added by the legitimate movie company should be searched and matched to detect pirated movies among downloaded movies. OpenCV feature matching technique detects objects from images even though there are slight brightness changes, size changes, or the images are rotated into different directions.

d. *Image Frame Matching*

If the system unable to detect watermarks, converted image frames from downloaded movies and the original movie get compares with each other to detect similar images. FLANN is Fast Library for Approximate Nearest Neighbors. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features. After FLAAN checks for similar key points and matches, next, the system checks and matches the image frame sizes and color channels. After checking for image frames matching, the system calculates the percentage of the image frame similarity.

e. *Motion and Audio Detection*

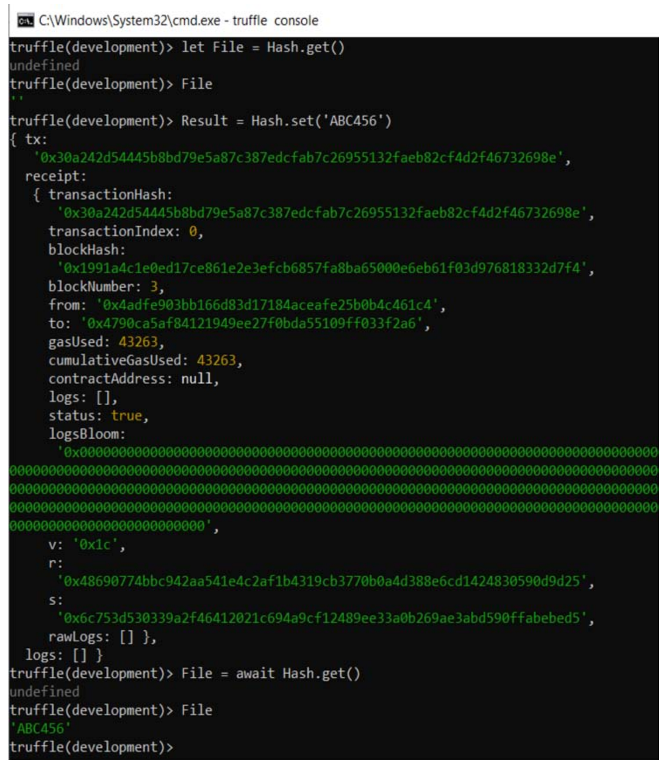
If the percentage of the image frame similarity is unable to confirm pirated videos, the system uses motion detection and audio detection to identify them. OpenCV motion detection technique detects the number of motions within two frames. The system uses this technique to calculate motions in videos. It detects each and every motion within two frames, which might occur inaccurate results. Because of this, the motion matching is further improved to detect motions in an area size of 1000 or more pixels.

Moviepy is a python module for editing videos. Moviepy can extract audio files from a video by simply calling the audio method. Audiodiff is a simple python library that is used to compare audio files. It can detect if two audio files contain the same audio streams and normalized tags.

III. Results and Discussion

a) Media Licensing

To ensure the smart contracts are working as intended, test cases were written to test the functionality. Tests were written using the Chai assertion library and Mocha Testing Framework which was received with truffle framework. These are client-side simulations of the smart contracts and it shows the extract results that was intended to obtain (Figure 14).



```
C:\Windows\System32\cmd.exe - truffle  console
truffle(development)> let File = Hash.get()
undefined
truffle(development)> File
''
truffle(development)> Result = Hash.set('ABC456')
{ tx:
   '0x30a242d54445b8bd79e5a87c387edcfab7c26955132faeb82cf4d2f46732698e',
  receipt:
   { transactionHash:
      '0x30a242d54445b8bd79e5a87c387edcfab7c26955132faeb82cf4d2f46732698e',
     transactionIndex: 0,
     blockHash:
      '0x1991a4c1e0ed17ce861e2e3efcb6857fa8ba65000e6eb61f03d976818332d7f4',
     blockNumber: 3,
     from: '0x4adfe903bb166d83d17184aceafe25b0b4c461c4',
     to: '0x4790ca5af84121949ee27f0bda55109ff033f2a6',
     gasUsed: 43263,
     cumulativeGasUsed: 43263,
     contractAddress: null,
     logs: [],
     status: true,
     logsBloom:
      '0x000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000',
     v: '0x1c',
     r:
      '0x48690774bbc942aa541e4c2af1b4319cb3770b0a4d388e6cd1424830590d9d25',
     s:
      '0x6c753d530339a2f46412021c694a9cf12489ee33a0b269ae3abd590ffabebed5',
     rawLogs: [] },
  logs: [] }
truffle(development)> File = await Hash.get()
undefined
truffle(development)> File
'ABC456'
truffle(development)>
```

Fig. 14: Test cases to obtain the file hash from blockchain

b) Forensic Watermark

To train the model, the Hollywood2dataset is used. This dataset is a collection of over 2500 short clips taken from various movies, all depicting certain actions such as getting into a car, opening a door etc. These clips train the Attention model to identify specific objects in a frame.

After the embedding process, the resulting watermarked frame does not show any visible degradation when compared to the original image (Figure 15).

Fig. 15: Comparison of original image and watermarked image

The table below shows the video quality after embedding 32 bits of random data as a watermark into the video. The PSNR column is the computed peak signal-to-noise ratio. The high value of the resulted PSNR shows that the watermarked frame is of a good quality.

Global Journal of Computer Science and Technology (E) Volume XX Issue IV Version I Year 2020

Table 2: Quality of Watermarked Frame

Model	Bits	PSNR
Attention	32	42.71
Attention + Noise	32	42.61

c) Web Crawler Algorithm

The framework has been tested and tried on known pirating websites such as yts.ag and Piratebay. It has been able to locate the division the link to the pirated files exist, however, the downloading process is haphazard. The files-pipeline should be tuned to download magnet files, which store the hashcode for the torrented media files. This has been an issue since the framework needs to invoke a different application to download the pirated media file through the magnet file.

d) Copy Detection

When using watermark comparison, the accuracy of the results decreases with the opacity (figure 16). When comparing, using low opacity image as the template and using the original image as a template, using the original image shows more accurate results than using images with low opacity.

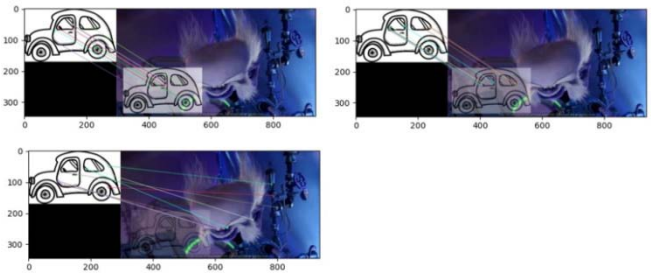


Fig. 16: How accuracy changes with the opacity

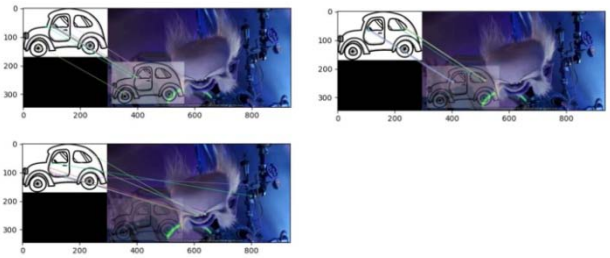


Fig. 17: When using an image with low opacity as the template

When comparing image frames, the system checks for the distance between key points, image size, and color channels. Image noise reduces the accuracy of the results. For example, it can show high accuracy when comparing completely different images with a darker background.

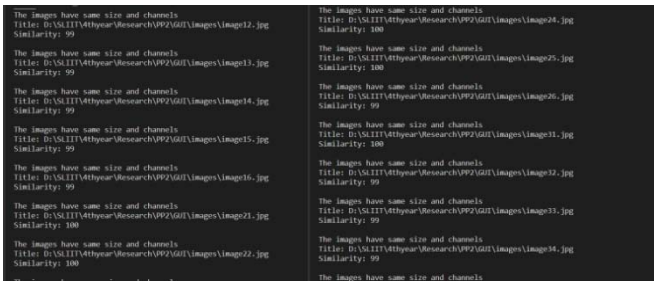


Fig. 18: Results of image frame matching

In the motion detection module, detecting every motion makes the calculation harder and inaccurate. Especially when calculating motions in animation movies, detecting every change between two video frames can give wrong similarity results. Assigning an area size of 1000 or more pixels makes the system only detects major motions and therefore gives more accurate results.

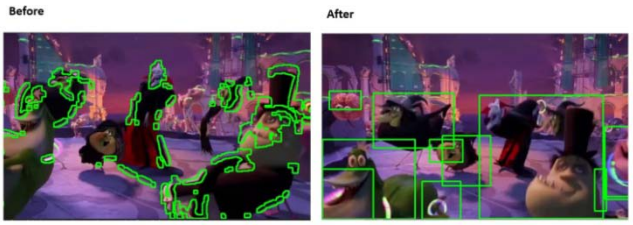


Fig. 19: Before and after assigning an area size

IV. Future Work

a) IP Protection Component

The functionality of the entire component can be improved in the future by developing a user friendly interface that is compatible with the blockchain. Also:

- The search function should be developed to search the already deployed documents and to obtain the details. As per now only the owner can view the details of the documents, but this should further improve to allow the owner to grant access to other users to view the documents.
- The ownerable.sol smart contract should future develop to save the registration details of the account holder in blockchain when they first created the user account.

The final task is to generate the digital certificate that allows IP holders to prove their ownership of the media.

b) Streaming Service Component

Further improvements can be made to the watermarking system. There are two key areas that can be improved further:

- Increasing the performance of the embedding and extraction process: The system currently uses CUDA for GPU-based acceleration. The issue with this is that it is currently only supported by NVIDIA.

The system must be modified so that it supports Intel and AMD based chipsets as well.

- Further improvement of the robustness of the watermark. As of now, the watermark is resistant to cropping and scaling of the video. Further persistence can be obtained by adding Compression resistance. This can be done using another noise layer that uses discrete cosine transform (DCT) before embedding the watermark, thus forcing the system to embed the watermark in a compression-resistant manner.

After completing the watermarking system, the Streaming service must be developed which has the ability to automatically embed the watermark into the video before hosting on the platform.

c) Web Crawler Algorithm

There are a few areas the framework and the controller classes for this algorithm can be improved further:

- Increasing the accuracy of the files' title detection is currently a main target as the title matching often turns out inaccurate. This is due to the Selector classes not being able to properly extract the desired filename.
- The downloading process should be streamlined. As of now, the web crawler is only able to download the magnet file which contains the hashcode for the pirated content. The user needs to have a third-party application to open the hashcode and access the pirated movie file. This process will be streamlined into an automatic sequence for the ease of use.

After the web crawler is perfected, it will be connected to the online blockchain database and the media server, where it'll be able to access all the metadata of the movies.

d) Copy Detection System

Video copy detection system can be further improved to get more accurate results.

- Image noise can be reduced from both watermark comparison and image frame comparison. Image noise affects the accuracy of the results.
- Audio matching comparison and motion detection comparison can be calculated to get the overall comparison and check similar movies from the downloaded movie results.

After detecting pirate movies in each step, the system should be developed to send alert messages to legitimate movie owners, allowing them to take relevant legal actions against the pirated movies.

V. Conclusion

Proper IP protection and legal action against media piracy has been a continuous struggle for many years. Over the years there has been a multitude of approaches that tackle these problems in various ways. Vanguard aims to actively search and take action against pirated media. The blockchain-based IP protection system offers a digital alternative to the traditional IP registration system, that will help ensure the owners IP rights in a case of illegitimate distribution of their media. The forensic watermarking system can help prove the origin of the media and directly prove the owner's rights. The Web crawler and Copy detection systems cooperatively aim to actively find pirated copies of media throughout the internet and alert the owner if they are found. All these components together can provide its clients with all-round protection against digital piracy, and in turn will actively aid in deterring the continuation of piracy of digital media.

Acknowledgment

We sincerely thank our evaluation panel examiners, Mr. Amila Senarathne, Mr. Buddhika Harshanath, and Ms. Thamali Dassanayake for their valuable feedback and suggestions given during presentation, evaluation, and discussion sessions. Special thanks also go to Dr. Janaka Wijekoon for hislectures and advice given to carry out a successful research. Finally, our sincere thanks go to each and every one who aided us in every step towards the completion of this research project.

References Références Referencias

1. "What is intellectual property? / World Intellectual Property Organization." Geneva: World Intellectual Property Organization, 2005.
2. J. Haskins, "How to Copyright A Movie or Short Film," legalzoom.com, 16-Jan-2018. [Online]. Available: https://www.legalzoom.com/articles/how-to-copyright-a-movie-or-short-film [Accessed: 21-Feb-2020].
3. Pace Technical, "A Brief History of Streaming Media": [online], Available: https://www.pace technical.com/brief-history-streaming-media/.
4. C. Peukert and J. Claussen, "Piracy and Movie Revenues: Evidence from Megaupload," SSRN Electronic Journal, 2012.
5. "United States Copyright Office - Annual report for fiscal 2018." [Online]. Available: https://www. copyright.gov/reports/annual/2018/ar2018.pdf [Accessed: 23-Jan-2020].
6. C.-E. Renault, "From script to screen: the importance of copyright in the distribution of films." Geneva, Switzerland: World Intellectual Property Organization, 2011.
7. Hill, Charles W. L. "Digital Piracy: Causes, Consequences, and Strategic Responses." Asia Pacific Journal of Management 24, no. 1 (2007): 9–

25. [online], Available: https://doi.org/10.1007/s10490-006-9025-0.
8. DigitalTV. (2017, October 30). DigitalTV Europe. Retrieved from Piracy to cost TV and film industry US$52bn by 2022. [online], Available: https://www.digitaltveurope.com/2017/10/30/piracy-to-cost-tv-and-filmindustry-us52bn-by-2022/.
9. W.-T. Tsai, L. Feng, H. Zhang, Y. You, L. Wang, and Y. Zhong, "Intellectual-Property Blockchain-Based Protection Model for Microfilms," 2017 IEEE Symposium on Service-Oriented System Engineering (SOSE), 2017.
10. Coin Telegraph, "What is Ethereum. Guide for Beginners," [online], Available: https://cointelegraph.com/ethereum-for-beginners/what-is-ethereum.
11. H. Sudler, "Effectiveness of anti-piracy technology: Finding appropriate solutions for evolving online piracy," Business Horizons, vol. 56, no. 2, pp. 149–157, 2013.
12. CustosTech, "Watermarking is not enough - How Blockchain can be used to stop online piracy," [online], Available: https://www.custostech.com/wp-content/uploads/2019/04/custos-technology-whitepaper.pdf.
13. C. Hill, "How Long Does It Take for Your Work to Get Copyrighted?," LegalZoom Legal Info, 18-Jul-2016. [Online]. Available: https://info.legalzoom.com/long-work-copyrighted-23878.html [Accessed: 19-Feb-2020].
14. C.-E. Renault, "From script to screen: the importance of copyright in the distribution of films." Geneva, Switzerland: World Intellectual Property Organization, 2011.
15. Zhang, Kevin Alex and Xu, Lei and Cuesta-Infante, Alfredo and Veeramachaneni, Kalyan. Robust Invisible Video Watermarking with Attention. MIT EECS, September 2019.
16. Kosiorek, "Attention in Neural Networks and How to Use It": [online], Available: http://akosiorek.github.io/ml/2017/10/14/visual-attention.html.