Journal of Positive School Psychology
2022, Vol. 6, No. 4, 278-282

http://journalppw.com

# SDN based pollution attack detection and prevention in cloud computing

**P.Santhosh Kumar, K.R.Nithesh, Dr. R. Jebakumar**
*Department of Computer ScienceEngineering*
*SRM Institute of Science and Technology*
Kattankulathur, India

*Abstract*—Web applications are vulnerable to malicious attacks performed by the intruders. Thus software defined network based policies are used to detect and prevent the attacks in an efficient manner. Thus improvising Software defined Networks (SDN) policies can protect data owner confidentiality and trust. The security is the major concern for the data owners during cloud migration. The malicious packet are detected during file transmission through deep packet inspection before processing the packet to the server. Thus SDN Based Pollution Attack Detection and Prevention (SDN PADP) is proposed here to detect the pollution attack in prior eliminating the server being compromised. In the process, if the pollution attack is been detected during file transmission, the respective IP is been blocked and the respective request process would be hold transmitting to the server. The pollution attack during file transmission is prevented by hashing. The hash value is been generated and append to the respective file during transmission to secure the file. SDN PADP invokes generation of convincing file during data the file upload ~~to~~ and preserve the privacy by cloud service providers against the legal authorities. For experimental results, java based web application is used in local host as testbeds.

***Keywords: Pollution Attack, cloud security, SDN prevention measure***

## I. Introduction

Software-Defined Network (SDN) are important aspect in recent days to define the conditions, policy for both software, hardware applications to detect external attacks and also train the system to provide recommendation of prevention measure to the network operators when the attack happens[a,b..c]. The SDN based policy renders effective network, data security eliminating the complexity level and attack incidence within the network. Usage of SDN trains the controller to inspect the files during data transmission identifying packet injection attack. This is achieved by training the controllers with pre-defined instructions. The another important feature is that SDN controller can be trained more efficiently based on new polices to upgrade the controller against new attack patterns [1].

Due to the improvement, technology adoption of cloud computing, security is the major concern for data owner acquisition and technology migration. Survey of several cloud attacks, packet injection attack is a vulnerable one making the
server compromised. Packet injection attack is defined as injecting malicious script in the file been transmitted from the source to the destination path. The malicious script can be framed to steal user information, compromise the server thus making data, response unavailability. Sometime the malicious script can be framed to pollute the file which can take control over the server, infrastructure. Thus security plays an important role in data security, confidentiality. Thus rendering security is more essential to gain user trust. Aside p
reserving user privacy from high government officials is also important to protect data owner privacy [2]. Elimination of duplication of files also optimize the cloud storage and reduce cost for the end users, thus saving encryption cost and storage space. The packet injection is inspected and prevented using the hash value generation. During file transmission the hash values are appended to the transmitted file by our SDN polices. If external attack is performed the hash values changes and in the received side the hash value is been matched and if there is any change in the hash value unauthorized access performed is been identified. Thus this polluted packet is been hold in further processing. The SDN policies is been trained to inspect the polluted and unpolluted packets before processing the requests.

In existing approach many data owners think that their data stored in cloud environment is secure against unauthorized access but their privacy is been leaked by the cloud service providers when government officials request for few data owners file or information's. The cloud service provider also cannot deny their commands. Hence to preserve the user privacy we have used generation of convincing files during data storage so that whenever government officials request the convincing file is been provided and when the respective data owner requests the exact file is been rendered [3]. Thus through this the data owner privacy is been protected from unauthorized access.

## II. RelatedWork

**[4]** Pollution attack is been addressed in both servers by detection and prevention measures in FLOWGUARD[4]. A security scheme is used, that utilizes two types of servers which are data and key servers. The data server is used to save the data and key server is used to save the keys. This system has many challenges in inspecting the polluted packet during transmission to data and key servers.

**[5]** Pollution attacks is defined during the flow of packets by inspecting the injected packet by the intruder externally.Thus this attacks creates an impact to the overall flow. For inspecting the packet injection attack the authors used codeguard as the prevention measure to prevent pollution attack. The challenge in this proposed methodology is that codeguard can able to inspect and identify one pollution attack packet only during an event. Thus during huge data transmission inspecting the malicious packet is anchallenge.

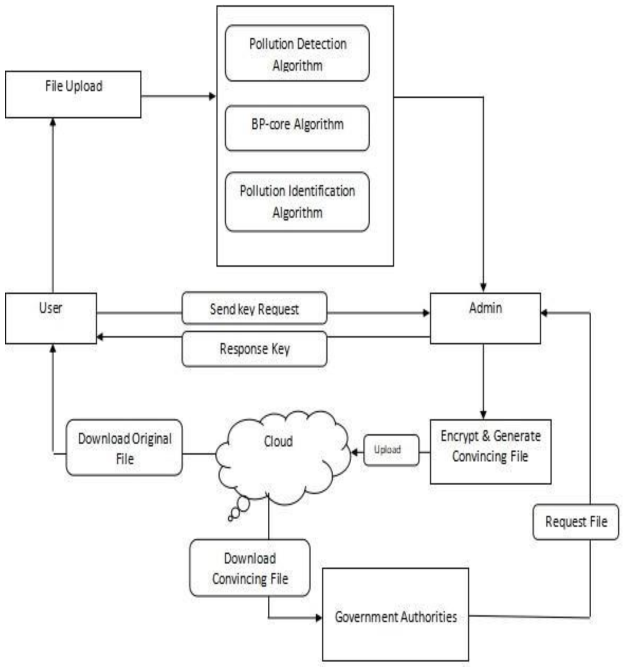
**[6]** The cloud computing aspect and how pollution attack is been identified during data transmission in the network are discussed in this research article. Because of emergence of cloud computing security of files during data transmission is very important. This paper inspects pollution attack by using encryption schemes that's is encoding the source files while data transmission. The encryption schemes always involve the risk of been compromised by the intruders using various hacking tools and techniques. Thus this proposed scheme doesn't render complete security against packet injection attack.

**[7]** The packet injection attack in networking is been elaborated in this research article. The packet injection attack is been detected and prevented using XOR network coding technique. The proposed XOR network coding invokes private key based authentication for authenticating the message. Thus all the data files are distributed into two parts in which each parts would be authenticated by various MAC's. This system might end up in congestion problem when huge data are been transmitted in a complex architecture.

**[8]** The traditional approaches for inspecting the pollution attack are not effective, this paper proposes cryptographic checksum values to the packet while transmission in the wireless sensor network. This paper is a theoretical survey paper explain the packet injection attack and prevention in wireless sensornetwork.

Thus from the survey of papers, the pollution attack with regard to file, web url still prevails in the web application thus creating a threat to the data owners during cloud migration. An strong defense system is essential to address the pollution, packet injection attack. In this paper, SDN PADP is proposed for cloud computing to ensure the security invoking encryption, checksum value, convincing filemethodologies.

## III .ARCHITECTURE DESIGN



## IV.MODULS

- User Registration &Validation
- Transforming data intofragments
- Pollution attackdetection
- HashingMethod
- Convincing file creation
- RC5encryption
- Cloud Storage

### A. *User InputValidation*

Data validation is the process of ensuring that a program operates on clean, correct and useful data. It uses routines, often called "validation rules" "validation constraints" or "check routines", that check for correctness, meaningfulness, and security of data that are input to the system. The rules may be implemented through the automatedfacilities

***B.** Transforming Data intoFragments**:***

A program change is any activity that takes a computer program and produces another program. Much of the time the changed program is needed to be semantically comparable to the first, comparative with a specific conventional semantics and in less cases the changes bring about projects that semantically contrast from the first predictablely While the changes can be performed physically, it is frequently more down to earth to utilize a program change framework that applies details of the necessary changes. Program changes might be indicated as computerized methodology that adjust compiler information structures speaking to the program text, or might be determined all the more advantageously utilizing designs speaking to defined source code text sections

*C. Pollution attackdetection:*

Pollution attack is defined as the injection of bogus coded fragments sent by malicious nodes in response to read requests, and thus in charge of reconstructing the original sectors starting from the set of corresponding coded fragments can be also used to carry out pollution detection. It is possible that a single polluted coded fragments propagates to many original sector fragments. However, in this work we show that coding brings also significant benefits in terms of pollution detection, since it can be exploited to both detect pollution and identify the nodes responsible of thedamage.

*D. HashingMethod:*

In this hashing method we encrypt the given data and stored into the database, after if anyone ask the request to get the file it will check the hash value is same or not, then only it allow to view or download the data fromdatabase.

*E. Convincingfile:*

The majority of the proposed plans expect distributed storage specialist co-ops or believed outsiders dealing with key administration are trusted and can't be hacked; be that as it may, a few elements may capture interchanges among clients and distributed storage suppliers and afterward force stockpiling suppliers to deliver client data by utilizing government power or different methods. For this situation, encoded information are thought to be known and capacity suppliers are mentioned to deliver client data. Since it is hard to battle against outside pressure, we meant to construct an encryption plot that could help distributed storage suppliers evade this scrape. In our methodology, we offer distributed storage suppliers intends to make counterfeit client data. Given such phony client data, outside coercers can just acquired produced information from a client's put away code text. When coercers think the got files are genuine, they will be fulfilled and all the more significantly distributed storage suppliers won't have uncovered any genuine insider facts. Accordingly, client security is as yetensured.

*B. RC5 Encryption:*

- The RC5 encryption algorithm is a fast, symmetric block cipher suitable for hardware or software implementations. A novel feature of RC5 is the heavy use of data-dependent rotations. RC5 has a variable-length secret key, providing flexibility in its security level. The algorithm can be broken into two stages: initialization, andoperation.
- In the initialization stage the 256-bit state table, S is populated, using the key, K as a seed. Once the state table is setup, it continues to be modified in a regular pattern as data is encrypted. The initialization process can be summarized by thepseudo-code;

```
j = 0;
fori = 0 to 255:
S[i] = i;
fori = 0 to 255:
j = (j + S[i] + K[i]) mod 256;
swap S[i] and S[j];
```

It is important to notice here the swapping of the locations of the numbers 0 to 255 (each of which occurs only once) in the state table. The values of the state table are provided. Once the initialization process is completed, the operation process may be summarized as shown by the pseudo codebelow;

```
i = j = 0;
for (k = 0 to N-1) { i
= (i + 1) mod256;
j = (j + S[i]) mod 256;
swap S[i] andS[j];
pr = S[ (S[i] + S[j]) mod 256]
output M[k] XOR pr }
```

Where M[0..N-1] is the input message consisting of N bits.

*C. Cloudstorage:*

Public cloud storage is a cloud storage model that enables individuals and organizations alike to store, edit and manage data. This type of storage exists on a remote cloud server and is accessible over the Internet. Public cloud storage is provided by a storage service provider that hosts, manages and sources the storage infrastructure publicly to many different users. Public cloud storage service is also known as storage as a service, utility storage and online storage. For this project we are trying to use Sync publiccloud.

## V. METHODOLOGY

SDN PADP is proposed to effective defense mechanism against packet injection attack and user privacy preserving challenges. In the proposed method, the pollution attack is been detected by defining several software defined polices. Firstly while data transmission, the data file is been converted into fragments for data inspection. Next the deep packet inspection is been performed to detect the malicious content inspecting each fragments before transmitting to the server. The prevention measure invokes appending a checksum value i.e hash value to each data while transmission. This checksum value is been matches at the receiver end to validate the genuine and malicious packet. If any unauthorized access or intrusion is been approached the checksum value gets changes in that way the request won't be transmitted to the server in further processing. If the received packets are not polluted, the packets would be processed ordownloaded.

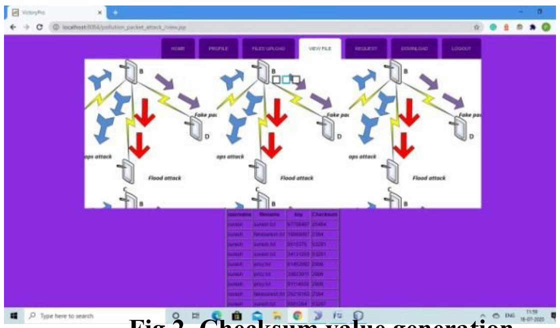
In most of the cloud environments, the data owner and cloud service provider always thinks the data cannot be hacked and secure. But the third parties holding the data owner data are not secure. When government authorities request the cloud service provider about the user secrets or confidential files, the cloud service provider cannot regret. In this case, the CSP might reveal the user data compromising the encryption policies. For this concern, the proposed system enables to provide convincing fake files during user uploads to preserve data owner secrets and privacy. When the data owner request, the user is authenticated. After authentication, the real data file is provided to access. If the government officials requests the user secrets or confidential data, the convincing file is provided by the cloud service provider. In this approach, the CSP can satisfy the government authorities by providing convincing file and not providing user real data. Thus finally the user data and privacy ispreserved.

To optimize the data owner cloud storage space we invoke de-duplication technique as well in the proposed methodology. The replication of files are been detected using Proactive Replica Checking approach which eliminates the storage of same file during data owner upload. This saves encryption cost and storage cost for the data owner optimizing the cloud storage.

For file encryption, we have used RC5 encryption algorithm and for cloud storage we have used public cloud, java programming language for experimental results.

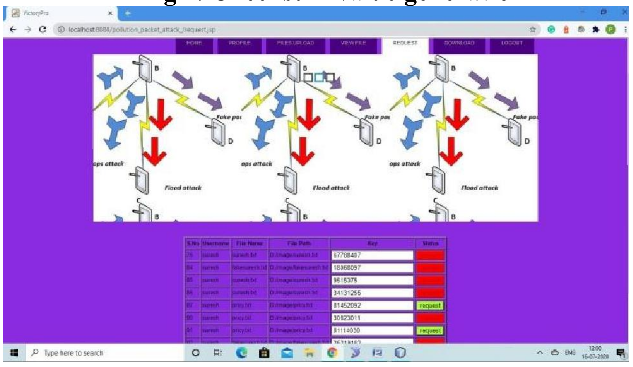
## V. EXPERIMENTAL RESULTS

In the experimental analysis, we have used java programming language to develop packet injection attack detection using prevention measures and also render data owner privacy preserving by generation of convincing files. In the experimental analysis during file transmission, an unique hash value is been appended to the file and this hash value gets altered when access externally without authorization. Also generation of convincing file is been implemented to protect the user privacy and confidentiality. Also to save the encryption and storage cost, we have used anti-replication of files technique in cloud environment



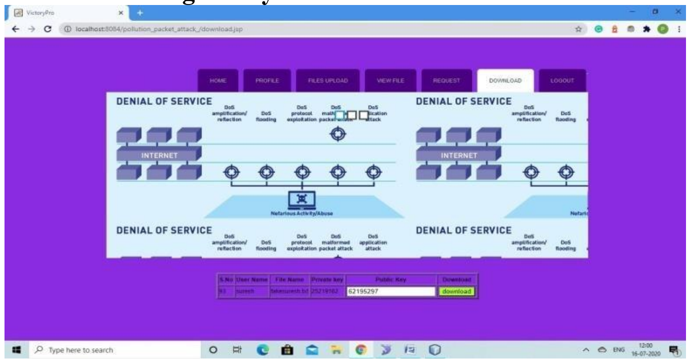
| username | filename | key | Checksum |
|---|---|---|---|
| suresh | suresh.txt | 67788407 | 25484 |
| suresh | fakesuresh.txt | 18068097 | 2384 |
| suresh | suresh.txt | 9515375 | 53281 |
| suresh | suresh.txt | 34131255 | 53281 |
| suresh | pricy.txt | 81452092 | 2906 |
| suresh | pricy.txt | 30823011 | 2906 |
| suresh | pricy.txt | 81114030 | 2906 |
| suresh | fakesuresh.txt | 25219162 | 2384 |
| suresh | suresh.txt | [illegible] | 53281 |

Fig 2. Checksum value generation



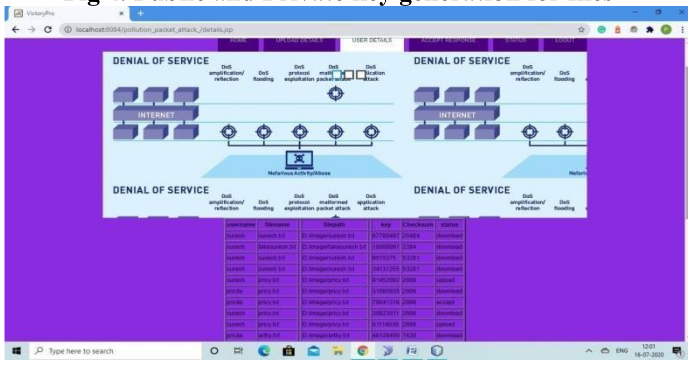
| S.No | Username | File Name | File Path | Key | Status |
|---|---|---|---|---|---|
| 78 | suresh | suresh.txt | D:/image/suresh.txt | 67788407 | 🟥 |
| 84 | suresh | fakesuresh.txt | D:/image/fakesuresh.txt | 18068097 | 🟥 |
| 85 | suresh | suresh.txt | D:/image/suresh.txt | 9515375 | 🟥 |
| 86 | suresh | suresh.txt | D:/image/suresh.txt | 34131255 | 🟥 |
| 87 | suresh | pricy.txt | D:/image/pricy.txt | 81452092 | request |
| 90 | suresh | pricy.txt | D:/image/pricy.txt | 30823011 | 🟥 |
| 91 | suresh | pricy.txt | D:/image/pricy.txt | 81114030 | request |

Fig 3. Key based authentication



| S.No | User Name | File Name | Private key | Public Key | Download |
|---|---|---|---|---|---|
| 93 | suresh | fakesuresh.txt | 25219162 | 62195297 | download |

Fig 4. Public and Private key generation for files



| username | filename | filepath | key | Checksum | status |
|---|---|---|---|---|---|
| suresh | suresh.txt | D:/image/suresh.txt | 67788407 | 25484 | download |
| suresh | fakesuresh.txt | D:/image/fakesuresh.txt | 18068097 | 2384 | download |
| suresh | suresh.txt | D:/image/suresh.txt | 9515375 | 53281 | download |
| suresh | suresh.txt | D:/image/suresh.txt | 34131255 | 53281 | download |
| suresh | pricy.txt | D:/image/pricy.txt | 81452092 | 2906 | upload |
| pricila | pricy.txt | D:/image/pricy.txt | 31085935 | 2906 | download |
| pricila | pricy.txt | D:/image/pricy.txt | 78641316 | 2906 | accept |
| suresh | pricy.txt | D:/image/pricy.txt | 30823011 | 2906 | download |
| suresh | pricy.txt | D:/image/pricy.txt | 81114030 | 2906 | upload |
| pricila | arthy.txt | D:/image/arthy.txt | 48126490 | 1630 | download |

Fig 5. Hash value validation

Fig 6. Convincing file generation

| File | Checksum Time (ms) |
|---|---|
| 2KB | 0.56 |
| 10KB | 0.78 |

Table 1. Checksum executiontime

Figure 7. Checksum executiontime

## VI. Conclusion

In this paper, the most common vulnerable attack is been addressed to gain data owner integrity and confidence towards cloud storage. The proposed system majorly focus on packet injection attack, replication of file to optimize the cloud storage space, user privacy preserving concerns. Integration of checksum values and validation of checksum values, convincing file generation and elimination of replication of files during storage are been implemented using java programming language by developing a web application. Thus this proposed system gains data owner confidentiality towards cloud storage and cloud serviceprovider.

.

## VII. Acknowledgment

The authors would like to thank the publishers, researchers for making their resources available and teachers for their guidance. We also thank the college authority for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and familymembers.

REFERENCES

[1] H. Dewan and R. Hansdah, “A survey of cloud storage facilities,” in IEEE SERVICES, jul 2011, pp. 224–231.

[2] C. Anglano, R. Gaeta, and M. Grangetto, “Exploiting rateless codes in cloud storage systems,” IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 5, pp. 1313–1322, May2015.

[3] L. Buttyan, L. Czap, and I. Vajda, “Detection and recovery from pollution attacks in coding-based distributed storage schemes,” IEEE Transactions on Dependable and Secure Computing, vol. 8, no. 6, pp. 824–838,2011.

[4] N. Cao, S. Yu, Z. Yang, W. Lou, and Y. Hou, “LT codes-based secure and reliable cloud storage service,” in IEEE INFOCOM, 2012, pp.693–701.

[5] L. Buttyn, L. Czap, and I. Vajda, “Pollution attack defense for coding based sensor storage,” in Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC),2010.

[6] M. N. Krohn, M. J. Freedman, and D. Mazieres, “On-the-fly verification of rateless erasure codes for efficient content distribution,” Security and Privacy, IEEE Symposium on, 2004.

[7] C. Gkantsidis and P. Rodriguez, “Cooperative security for network coding file distribution,” in IEEE INFOCOM,2006.

[8] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, “An efficient signature based scheme for securing network coding against pollution attacks,” in INFOCOM 2008. The 27th Conference on Computer Communications. IEEE,2008.

[9] E. Kehdiand B. Li, “Null keys: Limiting malicious attacks via null space properties of network coding,” in INFOCOM 2009,IEEE.

[10] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, “An efficient scheme for securing xor network coding against pollution attacks,” in INFOCOM 2009,IEEE.

[11] N. McKeown et al., “OpenFlow: Enabling innovation in campus networks,” ACM SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69–74, Apr.2008.

1[2] S. Jain et al., “B4: Experience with a globally-deployed software defined WAN,” ACM SIGCOMM Comput. Commun.Rev., vol. 43, no. 4, pp. 3–14, 2013.

[13] P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, “A security enforcement kernel for OpenFlow networks,” in Proc. ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw., Aug. 2012, pp. 121–126.

[14] H. Hu, W. Han, G. J. Ahn, and Z. Zhao, “FLOWGUARD: Building robust firewalls for software-defined networks,” in Proc. ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw., 2014, pp. 97–102.

[15] S. Matsumoto, S. Hitz, and A. Perrig, “Fleet: Defending SDNs from malicious administrators,” in Proc. Workshop Hot Topics Softw. Defined Netw., 2014, pp.103–108.

[16] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks,” in Proc. 20th ACM Conf. Comput. Commun. Secur., Nov. 2013, pp. 413–424.

[17] H. Wang, L. Xu, and G. Gu, “FloodGuard: A DoS attack prevention extension in software-defined networks,” in Proc. 45th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw., Jun. 2015, pp.239–250. S. Shin and G. Gu, “Attacking software-defined networks: A first feasibility study,” in Proc. Workshop Hot Topics Softw. Defined Netw., 2013, pp.165–166

[18]