Science World Journal Vol. 17(No 4) 2022
www.scienceworldjournal.org
ISSN: 1597-6343 (Online), ISSN: 2756-391X (Print)
Published by Faculty of Science, Kaduna State University

Full Length Research Article

# AN ADAPTIVE FRAMEWORK FOR COMBATING ADVANCED PERSISTENT THREATS

Ade Sandra Ngi, *Iorshase Agaji & Emmanuel Ogala

Department of Computer Science, Federal University of Agriculture, Makurdi, Nigeria

*Corresponding Author Email Address: ior.agaji@uam.edu.ng

**ABSTRACT**
Advanced persistent threats (APTs) pose a significant risk to nearly every organization. Due to the sophistication of these attacks, they can bypass existing security systems and largely infiltrate the target network. The prevention and detection of APT are challenging because attackers constantly change and evolve their attacking techniques and methods to stay undetected. As a result, APT often successfully compromises companies, organizations, or public authorities. This paper developed an adaptive security framework that continuously investigates the behavior of users of a network to protect it against threats. The framework constitutes of three main sections namely; Intrusion prevention, Intrusion detection, and Response to intrusions. The design model comprises the front end, middleware, and back end. The front end is implemented using HTML and Cascading Style Sheet (CSS) in Netbeans Integrated Development Environment (IDE) version 8.0.2. The middleware is implemented using Java Web of NetBeans IDE while the back end is implemented using MySQL server. The results show that the runtime security of the system is adapted according to the behavior patterns exhibited by the user hence, our system can detect zero-day attacks which signature-based intrusion detection systems cannot detect, thus protecting against these attacks. The work is recommended as a countermeasure against emerging persistent attacks.

**Keywords:** Cyber-attacks, Data exfiltration, Dwell times, Vulnerabilities, Threats, Zero-day attacks.

## INTRODUCTION
Today, we are faced with security threats that are more advanced than traditional security solutions can combat. Attacks and malware continue to advance faster than traditional security solutions can block them. Perimeter defenses such as network intrusion detection systems, firewalls, antivirus gateways, and also traditional endpoint security such as desktop anti-virus and host intrusion detection software aimed at keeping known threats out of the network are no longer sufficient against the exploits being used to conduct such attacks and leave organizations vulnerable to data breaches (Buecker, *et al*., 2016). Similarly, most of these security mechanisms are static which makes it difficult to achieve appropriate security for a dynamically changing environment and threat landscape. Adaptive security is defined by Jagadamba and Babu (2016) as the security protocol that detects the changes in the environment and disparities in the network services, predicts threats then adopts new security requirements, and executes itself without the intrusion of humans. Such systems aim to safeguard valuable assets in the face of changes in their working environment. This aim is achieved primarily by monitoring and analyzing its environment, and employing security functions that meet some protection requirements. Advanced Persistent Threat (APT) is a term that has been used over many years to describe a breed of insidious threats that use multiple attack methods and trajectories and which are conducted by stealth to evade detection so that hackers can maintain control over target systems for a long period (Tankard 2011). ATPs use a variety of techniques to gain initial access to a network. Attackers may use the Internet as a means to deliver malware and gain access, physical malware infection, or even external exploitation to gain access to protected networks (Brecht, 2015). ATPs do not take a general broad approach, unlike the traditional threats, instead, they are carefully planned and designed to attack a specific company or organization. Therefore, they are highly customized and sophisticated, designed specifically to evade the existing security measures put in place within a company.

## RELATED LITERATURE
Ryutov *et al*. (2015) combined two existing systems, TrustBuilder and Generic Authorization and Access-control (GAA)-API to create a framework which was more flexible and responsive to attacks than either GAA-API and Trustbuilder provided. The GAA-API provided adaptive access control that recognized changing system security requirements, on the other hand, the TrustBuilder system controlled when and how sensitive information was revealed to other groups. The security policies of the framework were modified according to the sensitivity of the access request and a suspicion level linked with the client. Despiteful activities were detected by examining failure types and forms of behavior of the access control and trust negotiation.
Saxena *et al.* (2007) presented a software framework for autonomic security which consists of an adaptation loop with monitoring, analyzing, and response components. In the monitoring modules, security-related events called security context are detected. The examples of security events given by the authors are new authentication schema available, user location change, and low memory. As a result, in an execution environment, security-related events can occur based on the received events. Each analysis component requires a high-level security action to reconfigure the system. The response component links these high-level security actions to application-specific sub-systems, for example, communication, device authentication, and application. The authors cited an example from the high-level security action which is 'increase encryption strength' that can be designed to increase the key size. The framework also consists of a support module in addition to monitoring, analyzing, and responding modules which provide a profile database for other modules. Each occurrence gotten from the monitoring component can be stored in the profile database for imminent use. Likewise, information is provided by the database for the analyzing modules to support decision-making. Finally, the information of the current configuration is stored in the profile database by the responding

node.
The framework proposed in this paper shall consist of access control, monitoring, analyzing, and responding modules. The analyzing module will analyze the behaviors of the users of the network application, these behaviors will be grouped into a set of network intrusion labels. The intrusions will be tackled by the responding module while normal activity will proceed to completion.
Gupta *et al*. (2012) propose a context profiling framework where he described device locking as an example of an application in which the locking timeout and unlocking method are decided based on the observed safety of the existing context. The familiarity and safety of a context at any instant are estimated by the framework and it uses these values to dynamically configure security policies. This approach uses the device to scan its environment for several context variables which include GPS readings, WiFi access points, and Bluetooth devices from time to time. The device can discover contexts that it encounters repeatedly according to these scans. These contexts are likely personal contexts of interest (CoIs) for the user. More so, the device can outline the CoIs by keeping a record of WiFi and Bluetooth devices that are encountered in a given CoI and the nature of those encounters. Similarly, the familiarity of a device with a context can be estimated using these profiles after which the inferred device knowledge values can then be used to estimate the knowledge of a context itself. Additionally, the device can use existing and historically accrued context knowledge information to establish the safety of the existing context.
Bardram *et al*. (2003) presented a context-aware user authentication protocol that used two different mechanisms for user identification and verification. A user was identified when the user presented his Smart Card to the system after which his correct presence was confirmed through a context-awareness system. If the context-awareness system was unavailable or cannot localize, the user was required to enter his password. The protocol may be vulnerable to an active replay attack, where an adversary steals a user's smart card.
Uszok *et al.* (2011) introduced a 3-layered security management architecture based on knowledge Agent oriented Service (KAoS) policies language. The layers were Policy specification, Policy reasoning and Policy enforcement. Agents were deployed in policy enforcement. These agents cached the policies and act as Policy Decision Point (PDP) for the applications. In the case of any security event raised by the application using Action Instance Description (AID), a java object encapsulates event details and as such the agent decides the correct action based on the cached policies. Morin *et al*. (2010) proposed a security-driven and model-based dynamic adaptation approach to adapt enforced access control policies in applications in accordance to changes in the application's context. That is, applying context-aware access control policies. The defined security policies take into consideration context information such that whenever the system context changed, the proposed approach updates the system architecture to enforce the suitable security policies. Mouelhi *et al*. (2008) introduced a model-driven security engineering approach that specified and enforced system access control policies at design time based on Aspect Oriented Programming - static weaving (AOP). Weaving merged Aspect with Classes to enable compilers to produce executable programs. The adaptive approaches required design time preparation to manually write integration code. The work supported only a limited security objective. Herzog *et al*. (2007) proposed a context-sensitive adaptive authentication framework which extended the traditional three-factor authentication by adding situational context. This involved two contexts- location and time, which were used in evaluating the probability and authentication level required. Different sensed identity tokens and location information collected from the devices in the environment were fused together to assess the level of authentication required in various situations. Hunt and Slay (2010) proposed a security architecture which can detect security breaches and in real-time record and analyse traffic logs in a forensically sound manner, provide corrective feedback to security devices and attempt to trace back to the source of the attack. In addressing computer security and forensic analysis from a real-time perspective the authors proposed methods whereby the ongoing damage and potential risk to critical infrastructure can be reduced. This required the implementation of a highly integrated approach to security and forensics such that they can work together in real-time in order to address the important security issues which currently face the industry. Muñoz *et al.* (2012) introduced a cloud computing dynamic monitoring architecture for the security attributes along with a language for expressing monitoring rules. The architecture was made up of three main layers which were; the local application surveillance (LAS) which collected measures from each application instance in virtual execution environment; the intra-platform surveillance (IPS) which collected measurements of different LAS elements and analysed them to detect violations. This approach focused on helping the service provider or administrator, however, it did not consider the involvement of the service tenants in developing and enforcing their own security metrics. Abie *et al*. (2010) proposed an Adaptive Security Manager (ASM) in a Genetic Messaging-Oriented Secure Middleware (GEMOM). ASM performs the required tasks for security adaptation using a learning mechanism. Monitoring was aided by integrating external tools, such as anomaly detectors, vulnerability discovery tools, a QoS (Quality of Service) monitor and security measurement tool. Self-protection was facilitated using the authorization component to protect against any intrusions. The authorization component also ensured confidentiality with a self-optimization capability. From context-awareness viewpoint, ASM focused mainly on QoS and security related information. The work, however, did not expound on how user requirements were addressed in their design and did not provide essential details of the analysis and adaptation components. Furthermore, the study mainly emphasized the monitoring aspects. The self-configuration, self-protection, and self-optimization properties were limited to a particular security objective. Self-optimization was limited only to confidentiality and trust services. Self-protection was restricted to authorization only, whereas self-configuration only addressed service availability. Ma and Wang (2013) introduced a self-adaptive access control model for a cloud-based system. The model was based on feedback loops and consisted of five phases. The phases were monitor, analyse, plan, execute, and knowledge base. The monitoring phase accessed requests, attributes, behaviour and history records of a user. The analysis phase examined the recorded access behaviour of a user, which determines whether it is required to update the knowledge base by selecting a sample from the history records. The knowledge base contained basic information of access control, including the relation degrees among access control attributes. The plan phase computed the relation degrees in the sample history records and updated, the knowledge base to provide decision support for access control. The execute module was used as an

interface to retrieve the new knowledge base and enforced the access decisions.

## METHODOLOGY

The proposed framework is a network framework with objects interacting with each other. Hence the research work adopts the object-oriented analysis and design methodology (OOAD). The architecture of the proposed framework is shown in figure 1. The architecture is divided into three components i.e. Intrusion prevention, Detector, and Response components. The intrusion prevention component receives inputs from the clients such as logins and monitors system access. The Detector analyzes clients' requests regarding their content and behavior. The functionality of the Detector is divided into four modules, namely, User Verifier, Role-Based Access Controller, and Behaviour analyzer. The modules are linked serially to each other in a specific order. If a suspicious request is found by any of the modules it is then forwarded to the Defender sub-component of the Response Component, else, the request is sent to the next module of the Detector for further verification. The Response component is made up of the Defender and the Defender Logger subcomponents. The Defender subcomponent takes countermeasures to prevent the network from adversarial operations in real-time by transforming the passive alert generated by the Detector into automated actions. Each response action is selected dynamically based on attack characteristics. The various actions that the Defender Component may initiate are; deactivating the user account, disabling the compromised services, recording the incident, rejecting the current request, making the user logged out, and blocking the IP address. The Defender Logger subcomponent documents details of the suspicious request and also the actions performed by the Defender in response to suspicious activity. The framework proposed in this paper incorporates Role-based Access Control and Attribute-Based Access Control assigning permissions and privileges to users based on their roles within the organization thereby, restricting unauthorized network access to data. In this case, sensitive information will be protected as only authorized users (employees) can have access to such information and perform actions they need to do their jobs.

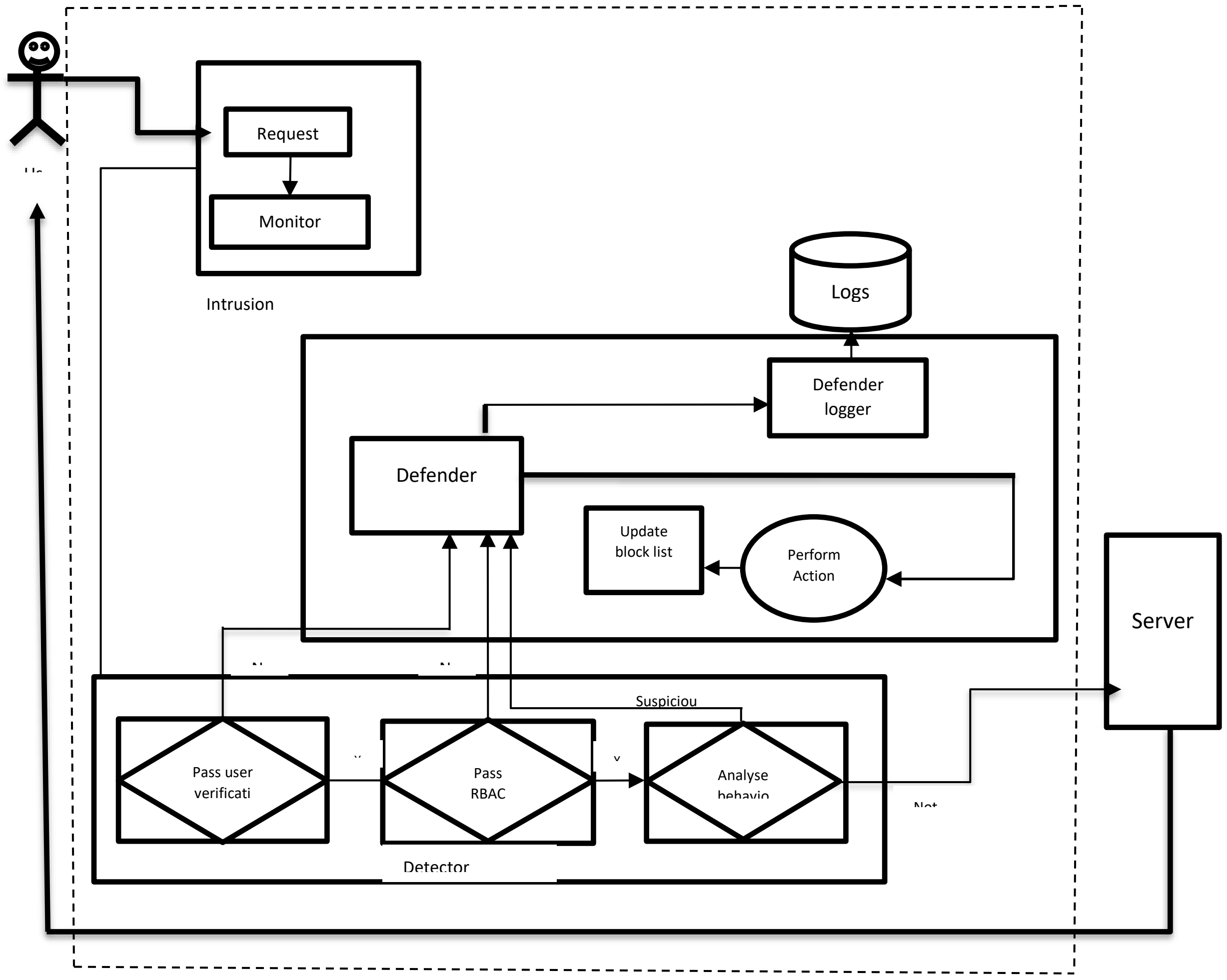


**Figure 1**: Architectural Framework of the Proposed System

*Science World Journal Vol. 17(No 4) 2022*
*www.scienceworldjournal.org*
*ISSN: 1597-6343 (Online), ISSN: 2756-391X (Print)*
*Published by Faculty of Science, Kaduna State University*

**Detailed Design of the Proposed Framework**
We used an activity diagram to model the dynamic view of the framework. An activity shows a set of actions, the sequential or branching control flow, and values that are produced or consumed by actions. Figure 2, depicts the activity diagram of the proposed framework.

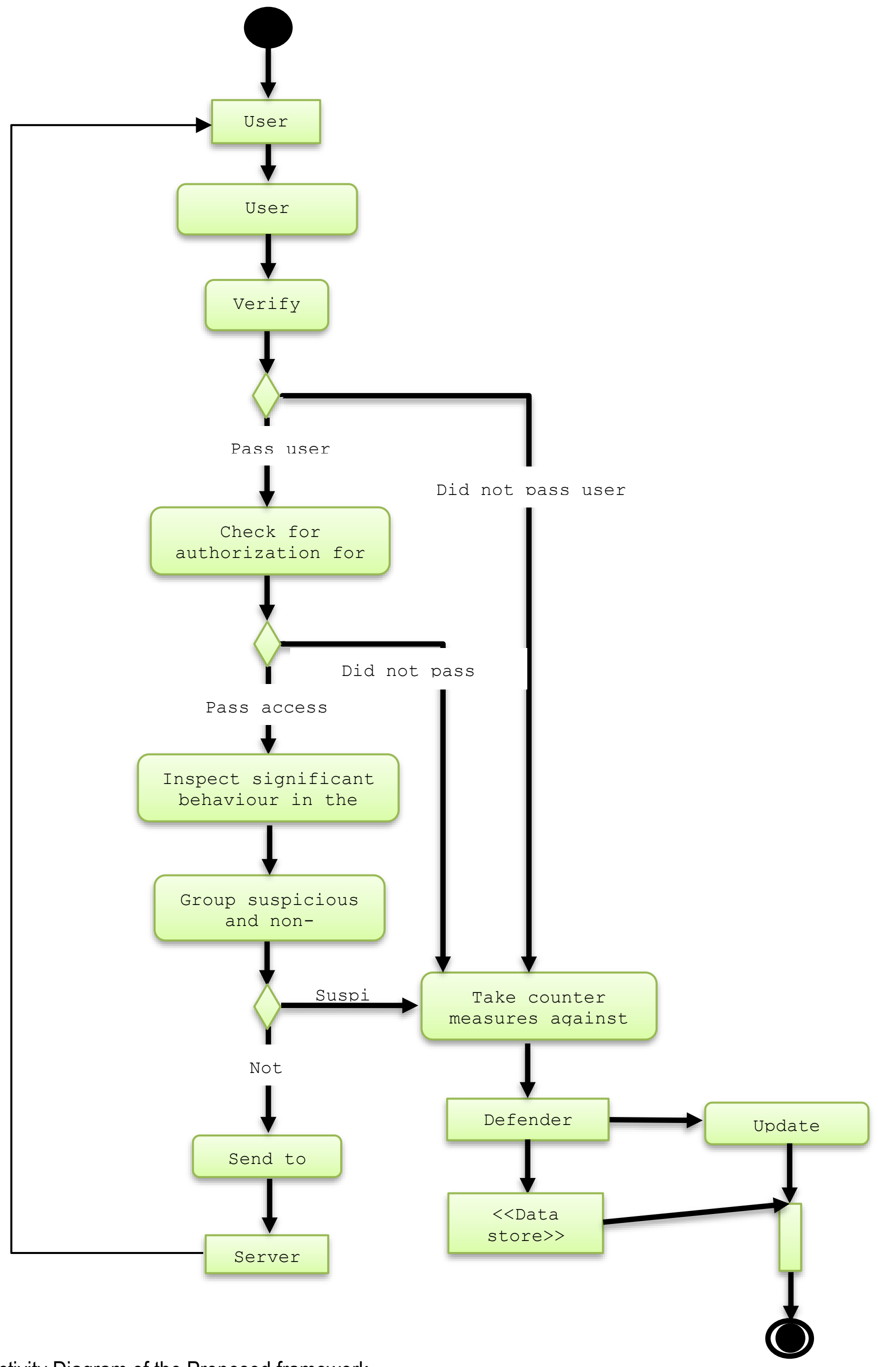

User
User
Verify
Pass user
Did not pass user
Check for authorization for
Did not pass
Pass access
Inspect significant behaviour in the
Group suspicious and non-
Suspi
Take counter measures against
Not
Send to
Defender
Update
<<Data store>>
Server

**Figure 2:** Activity Diagram of the Proposed framework

**IMPLEMENTATIONS**
The software tools used for the implementation were Windows Operating System 7, MySQL server, Java Development Kit (JDK 7) with Java Runtime Environment (JRE), Tomcat Application Server version 7, Internet Browser (Firefox, Mozilla, Google Chrome), and Netbeans IDE version 8.0.2. The framework makes use of a simple inventory to illustrate the adaptivity of a network system to advanced persistent threats. The framework consists of five modules; these are the login module that receives inputs in form of logins from the users. When a user is created, the user's details are stored in the database, upon login, the user's name and

password are cross-matched with the one in the database to check if they correspond. The user verifier module scrutinizes session-related details like the number of attempted logins, IP addresses, and token values to prevent an intruder from accessing the system using another user's identity. In the Role-Based Access Control (RBAC) and Attribute- Based module, different roles assigned by the framework are mapped to their equivalent set of allowable operations, and assessing the request is based on some predefined rules for determining if the user has the right to initiate the functionality. This module is designed to control user access to data. Therefore, each user of our system has his/her own defined roles and can only perform his sets of permitted operations. On the other hand, the ABAC controls what the user can do with a given resource and when the user can access this resource. The user behavioural activities module detects an attack by inspecting the major changes in the behaviour of the session that is operated. It monitors the set of performed operations that deviate from the characteristic profile behaviour of the legitimate user. The response module is embedded in the first four modules of our firework. Each time a suspicious request is made, it provides a countermeasure against the request. In such cases, the request is rejected by the framework, the user is logged out and IP address is blocked, and the user is suspended for a certain period.

## RESULTS AND DISCUSSION

Various experiments were carried out on the framework to demonstrate how the framework responds to threats by adapting itself to various security challenges.

**Experiment 1**: Intrusion Prevention using Multi-Factor Authentication (MFA)
This experiment aims at preventing threat actors from getting access to the network system. The inputs used here are the credentials of the user upon login which include, the user Id, one-time password, and token. A sample of the inputs are shown in table I

**Table I**: Inputs into the Framework for Experiment 1

| Users | Credentials | | |
|---|---|---|---|
| | User ID/ Roles | Password | Token |
| Administrator | admin | Ad24 | GC11673 |
| Administrator | admin | Ad24 | GC11680 |
| Administrator | admin | Ad24 | GC11685 |
| Administrator | admin | Ad24 | GC11687 |
| Mike John | sales rep | Sr56 | GC11692 |
| Administrator | admin | Ad24 | GC11701 |
| Administrator | admin | Ad24 | GC11726 |
| Administrator | admin | Ad24 | GC11740 |
| Administrator | admin | Ad24 | GC11742 |
| Mike John | sales rep | Sr24 | GC11744 |
| Sandra Ade | cashier | Ca89 | GC11751 |
| Administrator | admin | Ad24 | GC11673 |

The user's Id identifies the user, the one-time password (OTP) is used on a one-shot basis to log the user in at a particular period, and the token is used to continuously monitor in real-time that the legitimate person is using the service during the whole duration of the session. Every user of the system upon creation is assigned a username and password. To gain entry into the network application, the user must log in with his user name, password, and One-Time Password as an input is sent to the user's cell phone as token. shown in Figure 3.

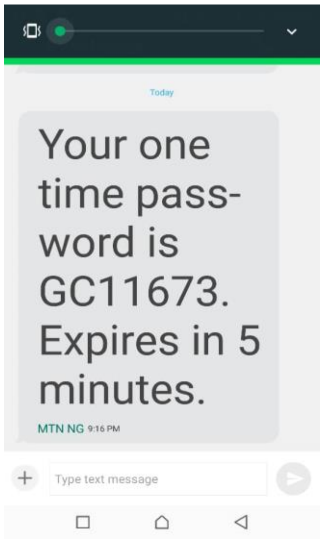
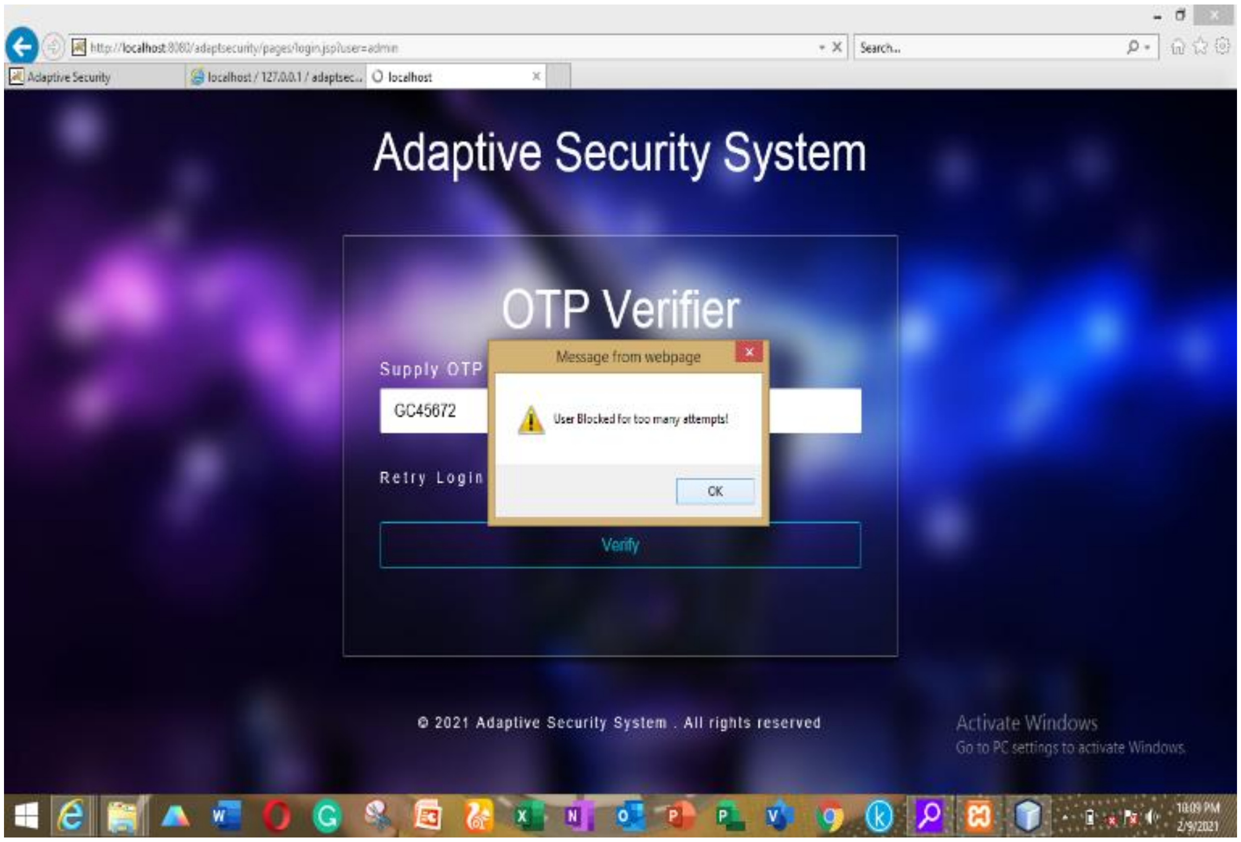

Today
Your one time pass-word is GC11673. Expires in 5 minutes.
MTN NG 9:16 PM

**Figure 3**: One-Time Password sent to User's cell phone
Once a user successfully logs in with his credentials (user-Id, one-time password), a token is generated that continuously monitors in real-time that the legitimate person is using the service during the whole duration of the session. A sample result showing more than three wrong login attempts is shown in figure 4



User Blocked for too many attempts!

© 2021 Adaptive Security System . All rights reserved

**Figure 4**: More than Three Attempted Wrong Login

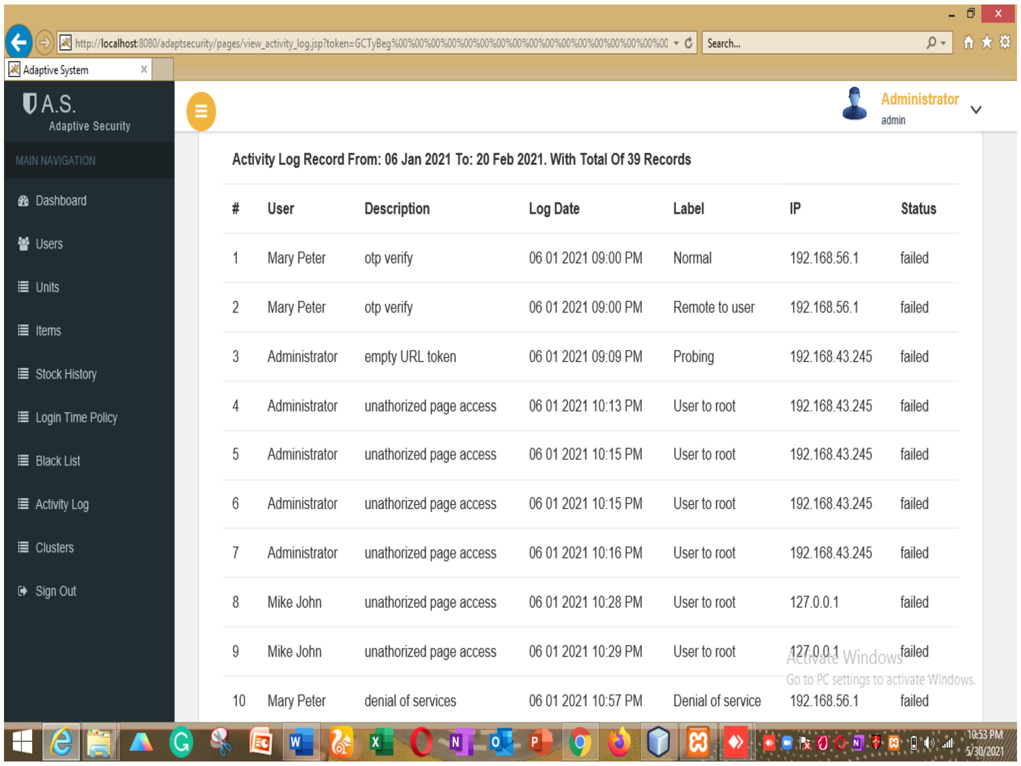
Figure 4 shows that the framework prevented intrusion by blocking an unauthorized user from accessing the application. It shows the return message when the OTP is wrongly inputted 3 times. When the verification of a user fails consecutively three times, the user is blocked. This implies that there is proper checking to ensure that an intruder is not guessing the credentials of a legitimate user. This checkmates intrusions into the network application.

**Experiment 2**: Intrusion Prevention using RBAC and ABAC
This experiment demonstrates intrusion prevention using RBAC ABAC. The input used here is defined time (in seconds) to network access as defined in ABAC policy shown in table II.

**Table II**: Inputs into the framework for experiment 2

| Users | Credentials | | | ABAC |
|---|---|---|---|---|
| | User ID /RBAC | Password | One-time password | |
| Administrator | Admin | Ad24 | GC11673 | - |
| Administrator | Admin | Ad24 | GC11680 | 07:00; 20:00 |
| Administrator | Admin | Ad24 | GC11685 | - |
| Administrator | Admin | Ad24 | GC11687 | - |
| Mike John | Sales rep | Sr56 | GC11692 | - |
| Administrator | Admin | Ad24 | GC11701 | - |
| Administrator | Admin | Ad24 | GC11726 | 08:00; 16:00 |
| Administrator | Admin | Ad24 | GC11740 | - |
| Administrator | Admin | Ad24 | GC11742 | - |
| Mike John | sales rep | Sr24 | GC11744 | - |
| Sandra Ade | Cashier | Ca89 | GC11751 | - |

Intrusion prevention is obtained first with the use of RBAC to determine who has access to a resource. Results in figure 5 shows that intrusion prevention is achieved in # 8 and #9 where a user called Mike John tried to gain administrative privileges. The operation failed because the system recognized it is unauthorized for a sales representative to perform the functions of an administrator. A failed status implies that the intended operation was not successful. It did not go into completion because the system detected it to be an intrusion so it was stopped before it could execute.



Activity Log Record From: 06 Jan 2021 To: 20 Feb 2021. With Total Of 39 Records

| # | User | Description | Log Date | Label | IP | Status |
|---|---|---|---|---|---|---|
| 1 | Mary Peter | otp verify | 06 01 2021 09:00 PM | Normal | 192.168.56.1 | failed |
| 2 | Mary Peter | otp verify | 06 01 2021 09:00 PM | Remote to user | 192.168.56.1 | failed |
| 3 | Administrator | empty URL token | 06 01 2021 09:09 PM | Probing | 192.168.43.245 | failed |
| 4 | Administrator | unathorized page access | 06 01 2021 10:13 PM | User to root | 192.168.43.245 | failed |
| 5 | Administrator | unathorized page access | 06 01 2021 10:15 PM | User to root | 192.168.43.245 | failed |
| 6 | Administrator | unathorized page access | 06 01 2021 10:15 PM | User to root | 192.168.43.245 | failed |
| 7 | Administrator | unathorized page access | 06 01 2021 10:16 PM | User to root | 192.168.43.245 | failed |
| 8 | Mike John | unathorized page access | 06 01 2021 10:28 PM | User to root | 127.0.0.1 | failed |
| 9 | Mike John | unathorized page access | 06 01 2021 10:29 PM | User to root | 127.0.0.1 | failed |
| 10 | Mary Peter | denial of services | 06 01 2021 10:57 PM | Denial of service | 192.168.56.1 | failed |

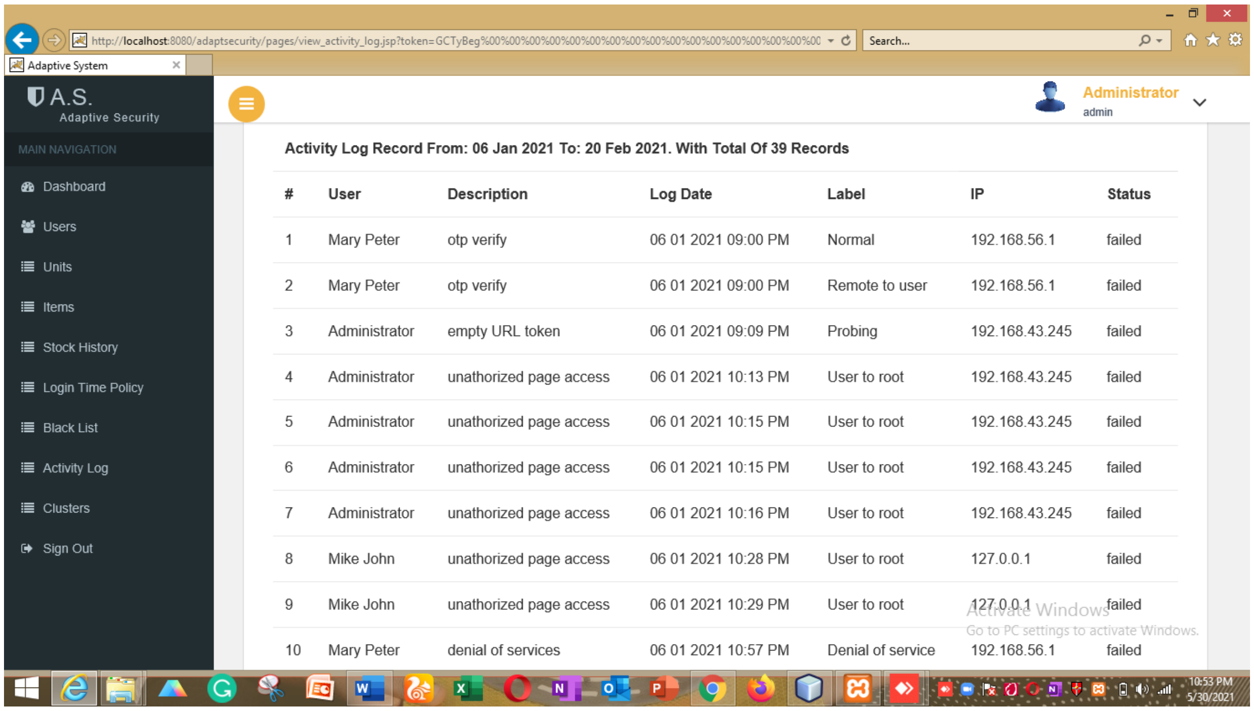
**Figure 5**: Intrusive Activities Counteracted by the Framework.

**Experiment 3**: Intrusion Detection and Response
This experiment demonstrates the behavior of the user on the network from the moment he logs into the network till when he logs out. The input used here is the activities of users on the network system as shown in table III.

**Table III**: Inputs into the framework for experiment 3

| Users | Credentials | | | ABAC Time (Hr:mm) | Activities |
|---|---|---|---|---|---|
| | User ID /RBAC | Password | One-time password | | |
| Administrator | Admin | Ad24 | GC11673 | - | Incorrect URL token |
| Administrator | Admin | Ad24 | GC11680 | 07:00; 20:00 | Set network lock policy |
| Administrator | Admin | Ad24 | GC11685 | - | Many requests in < 1 min |
| Administrator | Admin | Ad24 | GC11687 | - | Empty URL token |
| Mike John | Sales rep | Sr56 | GC11692 | - | Unauthorized page access |
| Administrator | Admin | Ad24 | GC11701 | - | Multiple connection from same user |
| Administrator | Admin | Ad24 | GC11726 | 08:00; 16:00 | Inconsistency in request sequences |
| Administrator | Admin | Ad24 | GC11740 | - | Successive failed login attempt |
| Administrator | Admin | Ad24 | GC11742 | - | Repetitive |
| Mike John | sales rep | Sr24 | GC11744 | - | Make sales |
| Sandra Ade | Cashier | Ca89 | GC11751 | - | Calculate expenses |

The system keeps track of all the activities performed by users. These activities form the behavioral pattern of users. The output of this experiment is shown in Figure 6 which displays the activity logs of users. The user names, description of the activity, time stamp during which the activity occurred, the network intrusion label, IP address, and, status of the operation whether it was successful or not.



Activity Log Record From: 06 Jan 2021 To: 20 Feb 2021. With Total Of 39 Records

| # | User | Description | Log Date | Label | IP | Status |
|---|---|---|---|---|---|---|
| 1 | Mary Peter | otp verify | 06 01 2021 09:00 PM | Normal | 192.168.56.1 | failed |
| 2 | Mary Peter | otp verify | 06 01 2021 09:00 PM | Remote to user | 192.168.56.1 | failed |
| 3 | Administrator | empty URL token | 06 01 2021 09:09 PM | Probing | 192.168.43.245 | failed |
| 4 | Administrator | unathorized page access | 06 01 2021 10:13 PM | User to root | 192.168.43.245 | failed |
| 5 | Administrator | unathorized page access | 06 01 2021 10:15 PM | User to root | 192.168.43.245 | failed |
| 6 | Administrator | unathorized page access | 06 01 2021 10:15 PM | User to root | 192.168.43.245 | failed |
| 7 | Administrator | unathorized page access | 06 01 2021 10:16 PM | User to root | 192.168.43.245 | failed |
| 8 | Mike John | unathorized page access | 06 01 2021 10:28 PM | User to root | 127.0.0.1 | failed |
| 9 | Mike John | unathorized page access | 06 01 2021 10:29 PM | User to root | 127.0.0.1 | failed |
| 10 | Mary Peter | denial of services | 06 01 2021 10:57 PM | Denial of service | 192.168.56.1 | failed |

**Figure 6**: Activity Logs

The experiment monitors the behaviors of users at runtime. Here, according to the behavior patterns displayed by the user, the runtime security of the system is adapted. The result of figure 6 shows the users' various activities viewed by the admin as it has been monitored and recorded by the system. The admin can insert the date range of the activities to be viewed. When that has been done, the activities of all users who have logged into the system within the specified date are displayed. From the result shown in figure 6, it can be seen that every user who has logged in has his IP address. This helps to know where the user is logged in from. The various activities carried out by a specific user are also documented with the time stamp of the particular activity. These activities are further grouped into denial of services, Probing, User to root, and Normal. Denial of services implies the attacker tries to prevent genuine users from using a service. Probing means a user's behavior indicates searching the network for vulnerabilities. User to root means the user of the network is making attempts to gain administrative privileges. Normal means no attack is detected in this class.

## CONCLUSION

This work includes multi-layered security in the three main sections, namely; intrusion prevention, detector, and, response. In intrusion prevention, we have incorporated the use of multifactor authentication to ensure that identity and privileges must always be verified through strict security protocols and not assumed. In intrusion detection, the use of role-based and attribute-based ensures that sensitive information is protected as only authorized users can have access to such information and perform actions needed for their jobs. In responding to threats, all abnormal behaviors from users are tackled with an appropriate response mechanism. In this work the users' activities were monitored and used to develop a threat monitoring and detection system The advantage of this work over the existing system is its ability to detect zero-day attacks which signature-based intrusion detection systems cannot detect. Results from the work also agree with the works of (Bardram *et al*., 2003) and (Gupta *et al*., 2012) who used passwords and patterns to authenticate users before granting them access to their network application system. However, their authentication method is one shot instead of continuous, therefore, their application stands a risk that a user change during a session will go undetected. Time-to-lock however used as an adaptation parameter to avoid session theft. In this case, when the user's idle period exceeds 10 minutes, verification of the user's identity is required. The work is recommended for a countermeasure against emerging persistent attacks.

## REFERENCES

Abie, H., Savola, R. M., Bigham, J., Dattani, I., Rotondi, D. and Bormida, G. D. (2010). Self-Healing and Secure Adaptive Messaging Middleware for Business Critical Systems. *International Journal of Advances in Security, 3*(1&2):34-51

Bardram, J. E., Kjær, R. E., and Pedersen, M. Ø. (2003). Context-Aware User Authentication – Supporting Proximity-Based Login in Pervasive Computing. *International Conference on Ubiquitous Computing, UbiComp2003: Uniquitous Computing*107-123

Brecht, D. (2015). *Current Trends in the APT World.* Infosec http://resources.infosecinstitute.com/current-trends-apt-world/

Buecker, A., Bulhoes, D., & Dobbs, M. (2016). Stopping Internet Threats Before They Affect Your Business by Using the IBM Security Network Intrusion Prevention System. *RedBRedbooks* http://www.redbooks.ibm.com/redpapers/pdfs/redp4683.pdf

Gupta, Adit, M., Markus, A., N., N., and Marcin. ( 2012). Intuitive Security Policy Configuration in Mobile Devices Using Context Profiling. *2012 International Conference on Privacy, Security, Risk and, Trust and 2012 International Conference on Social Computing*, 2012, 471-480

Herzog, A., Shahmehri, N. and Duma, C. (2007). An Ontology of Information Security. *International Journal of Information Security and Privacy, 1(4)*: 1-23.

Hunt, R. and Slay, J. (2010). Achieving critical infrastructure protection through the interaction of computer security and network forensics. *2010 Eighth Annual International Conference on Privacy Security and Trust (PST),* 23-30.

Jagadamba, G. and Babu, B. S. (2016). Adaptive Security Schemes based on Context and Trust for Ubiquitous Computing Environment: A Comprehensive Survey. *Indian Journal of Science and Technology*, 9(48): 1-13.

Ma, S. and Wang, Y. (2013). Self-Adaptive Access Control Model Based on Feedback Loop. *2013 International Conference on Cloud Computing and Big Data (CloudCom-Asia)*, 597-602.

Morin, B., Fleurey, F. and Mouelhi, T. (2010). Security-Driven Model-Based Dynamic
Adaptation. ASE 10 Proceediings of *IEEE/ACM International Conference on Automated Software Engineering*, 205-214.

Mouelhi, T., Fleurey, F., Baudry, B. and Le, T. Y. (2008). A Model-Based Framework for Security Policy Specification, Deployment and Testing. *International Conference on Model Driven Engineering Languages and Systems*, 37-552.

Muñoz, A., Maña, A. and Gonzalez, J. (2012). A Performance-Oriented Monitoring System for Security Properties in Cloud Computing Applications. *Computer Journal, **55**(8)*: 979-994.

Ryutov, T., Orosz, M., Blythe, J., and Winterfeldt, D. V. (2015). A Game-Theoretic Framework for Modelling Adversarial Cyber Security Game Among Attackers, Defenders, and Users. International Workshop on Security and Trust Management : Security amd Trust Management 274-282

Saxena, A., Lacoste, M., Jarboui, T., and Lücking, U. (2007). A Software Framework for Autonomic Security in Pervasive Environments. Third International Conference in Information Systems Security . *Information Systems Security* 91-109.

Tankard, C. (2011). Advanced Persistent Threats and how to Monitor and Deter them. *Network Security 2011(8)*: 16-19.

Uszok, A., Bradshaw, J., Lott, J. and Johnson, M. (2011) Toward a flexible ontology-based policy approach for network operations using the KAoS framework, *Military Communications Conference*, 2011. 1108-1114, doi: 10.1109/MILCOM.2011.6127447.