IJCSN - International Journal of Computer Science and Network, Volume 9, Issue 2, April 2020
ISSN (Online) : 2277-5420
www.IJCSN.org
Impact Factor: 1.5

Context-Integrity Access Control (CIAC): An Information Security Tool for a Secure Academic Web-Portal

[1] Onakoya, Johnson Rotimi; [2] Hammawa, Mohammed Baba

[1] Department of Computer Science, University of Abuja,
P.M.B 117, FCT. Abuja, Nigeria.

[2] Department of Computer Science, University of Abuja,
P.M.B 117, FCT. Abuja, Nigeria.

Abstract - Context-Integrity Access Control (CIAC) model is a family of Access control (AC). The CIAC is an extension of Context-Aware Access Control (CAAC) and Contextual Integrity Privacy. Context-Integrity Access Control (CIAC) is:

$u = (n(I) \cap n(A) \cap n(Au)) \cup n(t)$ → Inner Join

$t = u - (n(Au) \cap n(A) \cap n(I))$ → Outer Join

Then, $u - t = D$ → Intruder

Therefore, to defend Cyber sovereignty:

$n(Au) \cap n(A) \cap n(I)$ → Context-Integrity Mechanism

$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) = 1111$ → Allow access

To Quarantine intruder:

$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) \leq 1111$ → Deny access,

$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) \geq 1111$ → Deny access

The above set theory mathematical model is a symmetric differences of (Full Outer Join) except, (Inner Join) using hybrid tag methods in matching role with related Python with flask framework and Sequence Query Language (Flask-SQLAlchemy) that detects cyber terrorist, and blocks access control problems, distinct users and privileges, defend cyber sovereignty and silos data integrity.

Keywords – Context-Integrity, Cyber Sovereignty, Deny access, Hybrid tag methods, Intruder and Silos data integrity.

1. Introduction

There are security gaps in cyber space most especially in the Mobile and Web applications thus created a huge loss and tension in preventing cyber wars and terrorists. Many scholars had provided different types of warfare tools and concept such as Discretionary Access Control, Mandatory Access Control, Access Control List, Attributed-Based Access Control, Role-Based Access Control and several others to defend Cyber sovereignty yet many Cyber enemies are unknown and they are perpetrating evils day by day in the information Space. Therefore, Context-Integrity Access Control model is **aim** to detect Cyber criminals and problems of access control in information space. Will **objectively** distinct users, privileges and defend Silos data integrity with the use of mathematical model: set theory as related to Python with flask framework and Sequence Query Language (Flask-SQLAlchemy) for fighting, curbing, defending and safeguarding Silos data assets and permissions in a system. The Context-Integrity Access Control provide protection for remote users and insiders' influence also, provided for the mobile applications and Robots using Contextual Integrity Privacy [11].

Prosser [8] emphasised Security as a big business, fundamental and critical in any system and organisation. It is a state of Trust. The statement of the problem is that many scholars had proposed many concepts of Access Control (AC) models to guide against emerging trends of cyber insecurity but yet there are many unknown Cyber criminals and criminalities perpetrating on daily basis on the Cyber Space and sovereignty. Pachghare [12] told us that, information is powers dated back to eighteen century where military used secrete information to discredit their enemies using lock and key, fences, guard, Ceaser cipher, transposition, ciphers and steganography etc. to prevent access to information. Stratagem 1910, as quoted by Romuald[14] relating information knowledge as understanding the enemies, perceive yourself, you don't have to fright the outcome of the results of a thousands battles, but if you know yourself, not the enemy, you will suffer loss even when you won or gain victory. Most importantly, if you fail to understand neither the foe nor yourself, you will loose in every battle. The indices demonstrate that information is power or an edge of winning a War.

Cyber war is information warfare where all the actors i.e. information, attacks on the information and systems used are refer to as tools or strategy for acquiring an enemy's information [14]. Information Security access controls refer to who should do what during the interaction and privileges of a subject system policy. Information security therefore, means how access is being controlled to access computer system, read, write, update, delete object and service according to the action or domesticated policy.

1.1. Terminology Used:

The mathematical logical notations used in elucidating how Context-Integrity Access Control relates to Set theory:

Notation	Meaning
A	Authentication set
n(A)	All members/element of authentication set
Au	Authorization set
n(Au)	All member/elements of authorization set
I	Integrity check set
n(I)	All members/elements of integrity check set
Al	Authorization level 1
a2	Authorization level 2
Uf(x)	Users' function
U	Users (Insider and Remote Users)
t	Intruder
D	No. of Intruder(s)
R	Read
W	Write
P	Update
D	Delete

Source: The Proposed System, Onakoya et al. (2020)

2. Literature Review

This research works reviewed 10 difference Access control models [1] i.e. ACL [17], ABAC [5], DAC [18], HBAC [2], IBAC [9], MAC [21], OBAC, RBAC [3], RAC and CAAC [4] but wish to discuss 4 relevant concepts that were related to Context-Integrity Access control (CIAC).

Attributed-Based Access Control (ABAC) can be described as alternate model for traditional access control as pointed out by Servos [20]. The statement of the problem of the Attributed Based Access Control according to Hu et al. [5] is the inefficiency of traditional access control because of its cumbersome to manage. ABAC aim to improve upon Access control policies by introducing selected attributes for the objects and environmental conditions to receive it's globally awareness using logical Access Control Model [17]. Logical access model create access to objects by creating access rule without specifying individual relationships between each Subject and each object as a set of Subject attributes. Though, it lacks consensus on the use by the National Institute of Standard and Technology because it does not address the issue of improved Information sharing. Also, cumbersome to implement.

Access Control List (ACL) is an array of matrix [7]. The array contains access policies and pin points what should be protected and how it should be protected. ACL is a sequence of rules [7]. Access Control List is a list of Objects specifying Subjects to an Object [6]. It is very vital for Access Control [10]. ACL is a gatekeeper who permits entries as explained by Zenlayer [24]. Is not as if ACL is not having shortcoming, it has several difficulties:

i) ACL manages Information resources separately[3]
ii) Administrators finds it difficult to manage the large set of Information recourses created.

That is what led to the improved Role-Based Access Control method.

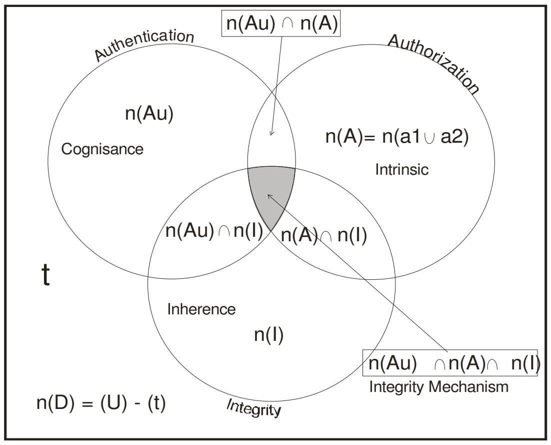
Role-Based Access Control (RBAC) according to Bourgeois [3], RBAC grants access to Users one after the other by assigning access right separately to a user and assigned role to assigned access [24]. That has demystifies the work of an Administrator but not actually having environmental information about the User hence, Context-Aware Access Control (CAAC)[4].

Context-Aware Access Control (CAAC), DuraiPandian et al.[4] worked seriously on the security needs of every Educational organisations by providing an Authentication on necessity and enforced dynamic authorization. Aim to protect information from unauthorized access and modification. The Context-Aware Access Control can be compromised with the use of Robot profile matrix permutation Strategy [16] and prediction for password and username hence, Context-Integrity Access Control (CIAC) [4].

3. Proposed Work

Context-Integrity Access Control (CIAC) research methodology is a **combination of Access models[3]** i.e. Access Control List (ACL)[7], Attributes Based Access Control (ABAC)[5] and Role Based Access Control (RBAC)[3] using hybrid tags methods and Flask-SQLAlchemy[19] to determine Subjects in the Objects by auto enforced policies for Subjects' corresponding permissions to the objects.

Domesticated policy means policies residing in a system for Security decisions, defensive and functional rules and guidelines enhancing cognisance, intrinsic, inherence and integrity mechanism. The CIAC model thus refer to inner join of a set of authentication and authorization, left outer join of authorization and integrity check, right outer join of authentication and integrity check and full outer join of a set of authentication, authorizations and integrity check, and full inner join of a set of authentication, authorizations and integrity check as expressed as Union in figure 1 below:



U=Security
Source: The Proposed System,Onakoya et al. (2020)
Figure 1: Venn diagram of Context-Integrity Access Control interceptions and mechanism

Security Operations Set theory

$u = (n(I) \cap n(A) \cap n(Au)) \cup n(t)$ → Inner Join
$t = u - (n(Au) \cap n(A) \cap n(I))$ → Outer Join
Then, $u - t = D$ → Intruder
Therefore, to defend Cyber sovereignty:
$n(Au) \cap n(A) \cap n(I)$ → Context-Integrity Mechanism

$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) = 1111$ → Allow access
To Quarantine intruder:
$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) \leq 1111$ → Deny access,
$(n(Au) \cap (a_1)) \cup (n(a_1) \cap (a_2)) \cup (n(a_2 \cap n(1)) \cup (n(a_2) \cap n(1)) \geq 1111$ → Deny access

The Table below is the interpretations of Venn diagram above into SQL commands [19], Procedure, Technique and working tool.

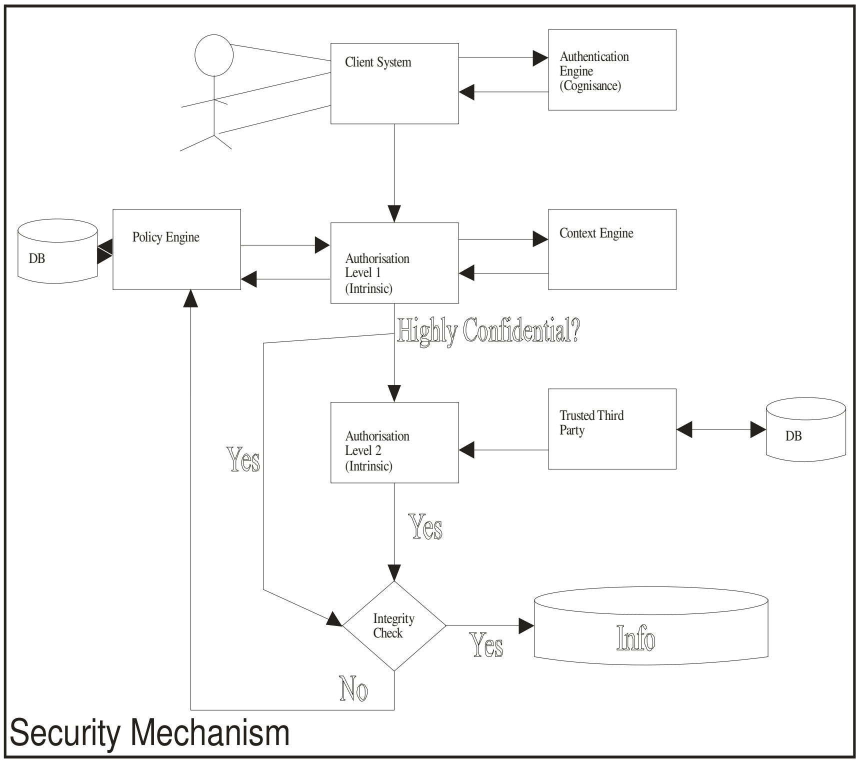
Table 1: Security Procedures for Context-Integrity Access Control Operations

Set-theory Terminology	SQL command	Procedure
Insertion n(AU) ∩ n(A)	INNER JOIN	A set of Authentication INTERSECT Authorization
Relative complement n(AU) ∩ n(I)	(LEFT OUTER JOIN) EXCEPT (INNER JOIN)	A set of Authorization INTERSECT Integrity check set
Relative complement n(I) ∩ n(A)	(RIGHT OUTER JOIN) EXCEPT (INNER JOIN)	A set of Authentication INTERSECT Integrity Check
Symmetric difference n(I) ∩ n(A) ∩ n(AU)	(FULL OUTER JOIN) EXCEPT (INNER JOIN)	A set of Authentication INTERSECT Authorization and INTERSECT Integrity Check
Union n(AU) U n(A) U n(I)	(FULL INNER JOIN) EXCEPT (OUTER JOIN)	A set of Authentication UNION Authorizations and UNION Integrity Check = SECURITY

Source: The Proposed System by Onakoya et al. (2020)

3.1. System Design and Algorithms

The context integrity security mechanism domesticated policy for Access to a secure academic web portal [20]. The information security tool combined constraints, behaviour of the system and consistency argument that uses array list to solve information insecurity issues using hybrid tags. The array starts from 0 to maximum. The System design is a modification of DuraiPandian[4]'s work. The figure below best illustrate the Security mechanism Architecture:



Source: N. DuraiPandian et.al (2006) but extended by the Proposed System with the Integrity Check

Figure 2: Proposed Role Based Access Control (RBAC) Context Integrity Information Security Architecture.

The context-integrity information security architecture as seen above is in 4 levels i.e.:

I. Trust level means authentication
II. Authorization level 1
III. Authorization level 2
IV. Integrity decision

3.1.1 Trust Level

Trust level in context-integrity means authentication. i.e. level 1 of the security mechanism. In authentication, data must be inputted. The authentication must not be empty and it must be valid. If the data trust failed, it returns 0 and ask to re-login. When trust request is positive, it returns 1 and pass to the next level. That is represented by tag 1 i.e. n(Au). See authentication code below:

```
@app.route('/login', methods=['GET','POST'])
def login():
    tag = []
    form = LoginForm()
    if form.validate_on_submit():
        user =
User.query.filter_by(email=form.email.data).first()
        if user is None:
            tag=[1,0,0,0]
        flash('Login Unsuccessful, please check email
and password!','danger')
        else:
            Go to Authorization Level 1
        flash(f"Tag = {tag}", 'warning')
    return render('login.html', title='Login', form=form)
```

3.1.2 Authentication Level 1

This is the second level of the security mechanism. The authorization level 1 distinguishes set of users and their interwoven details in authentication. The context-integrity information security mechanism checks for the correctness of the user's authentication. That is represented by n(Au)$\cap$(a_1). The security question here is “Who is the user? Is the identity (password) correct? The mechanism introduces encryption using Bcrypt [13] module to hash the users' password and tagged the operation, tag 2 as coded below:

```
@app.route('/login', methods=['GET','POST'])
def login():
    tag = []
    form = LoginForm()
    if form.validate_on_submit():
        user =
User.query.filter_by(email=form.email.data).first()
        if user and user.tag < 3:
            tag.append(1)
            if user:
                pwd = user.password
                if bcrypt.check_password_hash(pwd,
form.password.data):
                    tag.append(1)
                elif bcrypt.check_password_hash(pwd,
form.password.data) == False:
                    tag.append(0)
                if tag[0] != 1 or tag[1] != 1:
                    flash('Login Unsuccessful, please check
email and password!','danger')
                else:
                        Go to Authorization Level 2
        elif user is None:
            tag=[1,0,0,0]
            flash('Login Unsuccessful, please check email
and password!','danger')
        flash(f"Tag = {tag}", 'warning')
    return render('login.html', title='Login', form=form)
```

3.1.3 Authorization Level 2

Authorization level 2 is a sub set in authorization stage where identity is matched with role. If true, it returns 1 for tag is tag 3. If false return 0 for tag is tag3. That is represented in Set theory by $n(a_2) \cap n(I)$. The code is as below:

```
@app.route('/login', methods=['GET','POST'])
def login():
    tag = []
    form = LoginForm()
    if form.validate_on_submit():
        user =
User.query.filter_by(email=form.email.data).first()
        if user and user.tag < 3:
            tag.append(1)
            if user:
                pwd = user.password
                role = user.role
                if bcrypt.check_password_hash(pwd,
form.password.data):
                    tag.append(1)
                elif bcrypt.check_password_hash(pwd,
form.password.data) == False:
                    tag.append(0)
                if role == form.role.data:
                    tag.append(1)
                elif role != form.role.data:
                    tag.append(0)

                if tag[0] != 1 or tag[1] != 1:
                    flash('Login Unsuccessful, please check
email and password!','danger')
                elif tag[2] != 1:
                    flash('Login Unsuccessful, please contact
admin. You are not an authorized user!','danger')
                else:
                    Go to Integrity Check
        elif user is None:
            tag=[1,0,0,0]
            flash('Login Unsuccessful, please check email
and password!','danger')
        flash(f"Tag = {tag}", 'warning')
    return render('login.html', title='Login', form=form)
```

3.1.4 Integrity Decision

Integrity decision is the fourth level of the context-integration security mechanism. The integrity decision in terms of context-integrity, is the check for details in authentication, authorization level1 and 2 for correctness where 1 is the true and 0 is false. And returns answer for tag 5. To combine authorization for generation of n(Au) n n(a1) n(a2) Tag 5 if true return 1, false return 0 for tag 4. Then compute TTag for access therefore if TTag is true enable access, if TTag is false ask to re-login. The code is as below:

```
@app.route('/login', methods=['GET','POST'])
def login():
    tag = []
    form = LoginForm()
    if form.validate_on_submit():
        user =
User.query.filter_by(email=form.email.data).first()
        if user and user.tag < 3:
            tag.append(1)
            if user:
                pwd = user.password
                role = user.role
                if bcrypt.check_password_hash(pwd,
form.password.data):
                    tag.append(1)
                elif bcrypt.check_password_hash(pwd,
form.password.data) == False:
                    tag.append(0)
                if role == form.role.data:
                    tag.append(1)
                elif role != form.role.data:
                    tag.append(0)
                if tag[0] == 1 and tag[1] == 1 and tag[2] ==
1:
                    tag.append(1)
                else:
                    tag.append(0)
                if tag[0] != 1 or tag[1] != 1:
                    flash('Login Unsuccessful, please check
email and password!','danger')
                elif tag[2] != 1:
                    flash('Login Unsuccessful, please contact
admin. You are not an authorized user!','danger')
```

```
            if tag[3] == 1:
                user.tag = 0
                login_user(user,
remember=form.remember.data)
                next_page = request.args.get('next')


                return redirect(next_page) if next_page
else redirect(url_for('account'))
             if tag[3] == 0:
            user.tag = int(user.tag) + 1
            db.session.commit()

        elif user and user.tag >= 3:
          try:
            send_cica_email(user)
            # flash("Your account has been temporarily
locked", "danger")
          except:
            flash("Email was not sent, Connection
unexpectedly closed", "warning")
          # flash("Your account has been temporarily
locked", "danger")
          return redirect(url_for('tag'))
        elif user is None:
          tag=[1,0,0,0]
          flash('Login Unsuccessful, please check email
and password!','danger')
          flash(f"Tag = {tag}", 'warning')
    return render('login.html', title='Login', form=form)
```

4. Output

The context-integrity information security components comprise of five engines. These five engines interact together to become Context-Integrity Information Security mechanism. They include:

I. Authentication: Identifies and issuing authentication certificate who has access.
II. Authorization Engine: Monitors and Recognizes who had access to certain level of authority at a point of entry.
III. Classification Engine: Classifies roles and put restriction.
IV. Validation Engine: Validates access rights and authorized role.
V. Context-Integrity: Evaluates all policies in the objects and resulted to enable subject authorization.

4.1 Implementation screen snapshot

The below figures 3-9 are the samples of input forms and dialogue boxes from the new system (CIAC). We have adopted the use of valid email where identity can be double sure to replace Username, Password is still password and marching role as the new feature in Context-Integrity Access Control as indicated from figure 3-7 in addition to biometric, graphic designs and situational awareness.

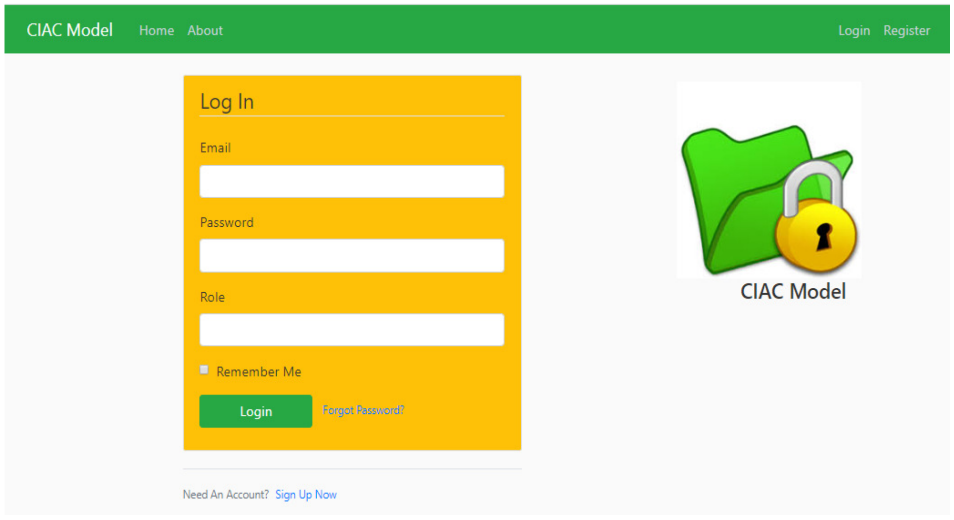
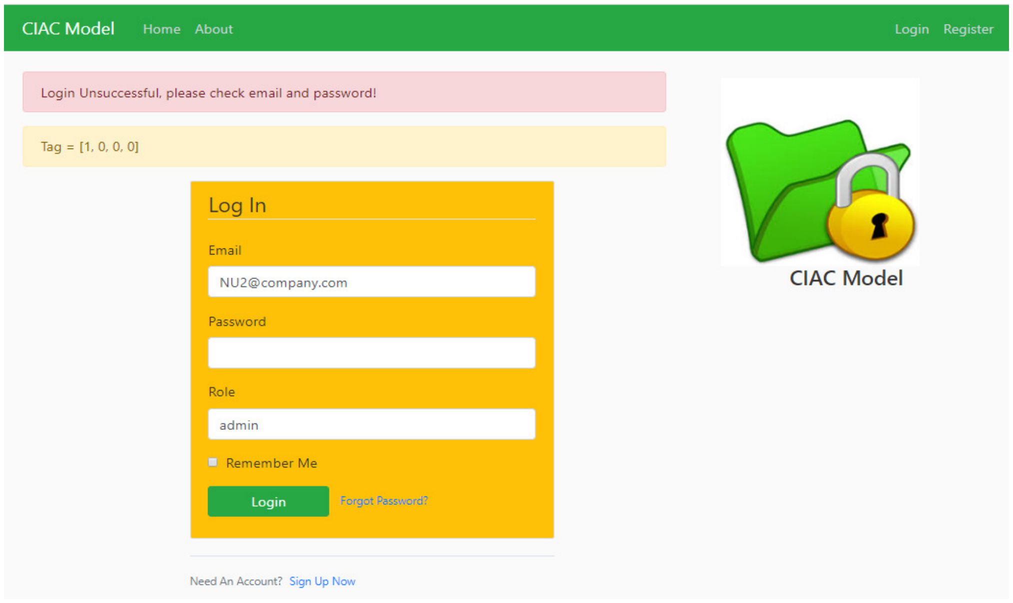


Fig. 3: Context-Integrity Access Control Model Login
Source: The proposed system by Onakoya et. al., (2020)

IJCSN - International Journal of Computer Science and Network, Volume 9, Issue 2, April 2020
ISSN (Online) : 2277-5420
www.IJCSN.org
Impact Factor: 1.5



Login Unsuccessful, please check email and password!

Tag = [1, 0, 0, 0]

Log In

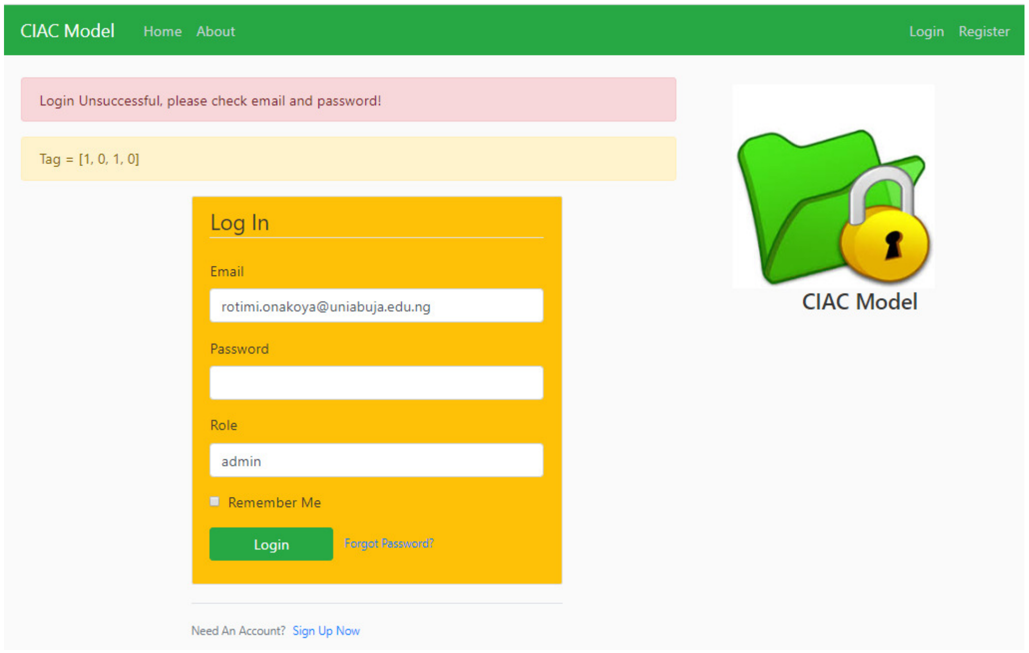
Email: NU2@company.com

Password:

Role: admin

Remember Me

Fig. 4: Context-Integrity Access Control Model Login for Wrong Email Details error message
Source: The proposed system by Onakoya et. al., (2020)



Login Unsuccessful, please check email and password!

Tag = [1, 0, 1, 0]

Log In

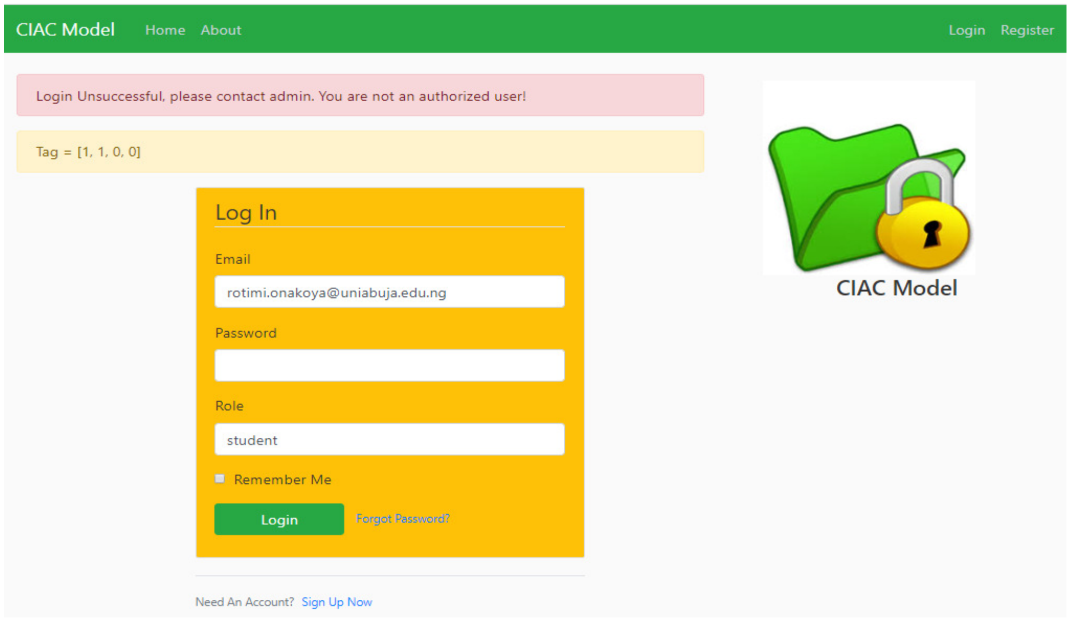
Email: rotimi.onakoya@uniabuja.edu.ng

Password:

Role: admin

Remember Me

Fig. 5: Context-Integrity Access Control (CIAC) Model Login for Wrong Password error message
Source: The proposed system by Onakoya et. al., (2020)



Login Unsuccessful, please contact admin. You are not an authorized user!

Tag = [1, 1, 0, 0]

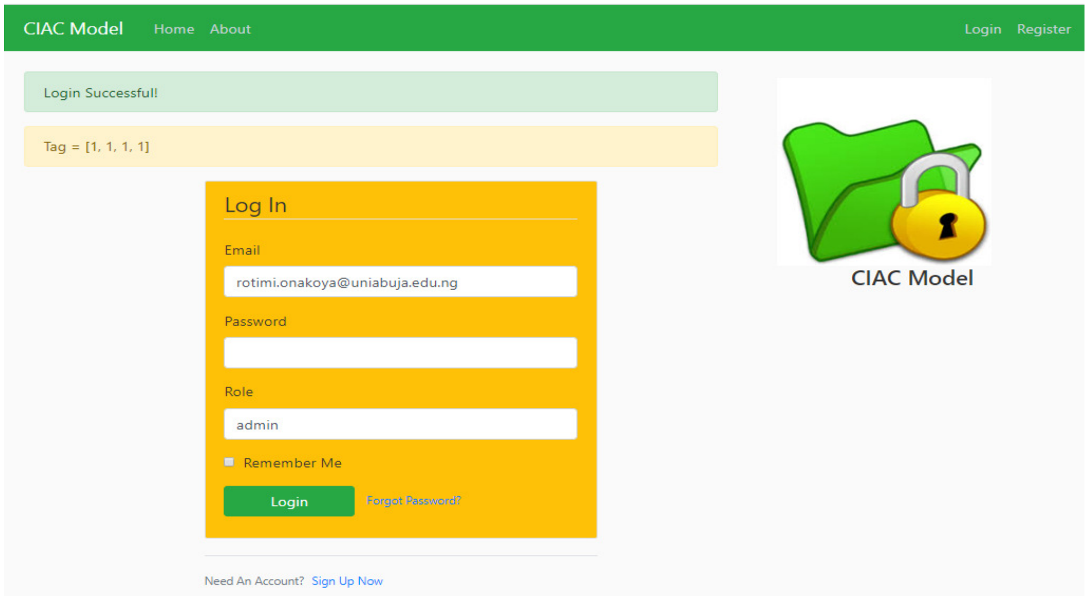
Log In

Email: rotimi.onakoya@uniabuja.edu.ng

Password:

Role: student

Fig. 6: Context-Integrity Access Control (CIAC) Model Login for Wrong Password error message
Source: The proposed system by Onakoya et. al., (2020)



Login Successful!

Tag = [1, 1, 1, 1]

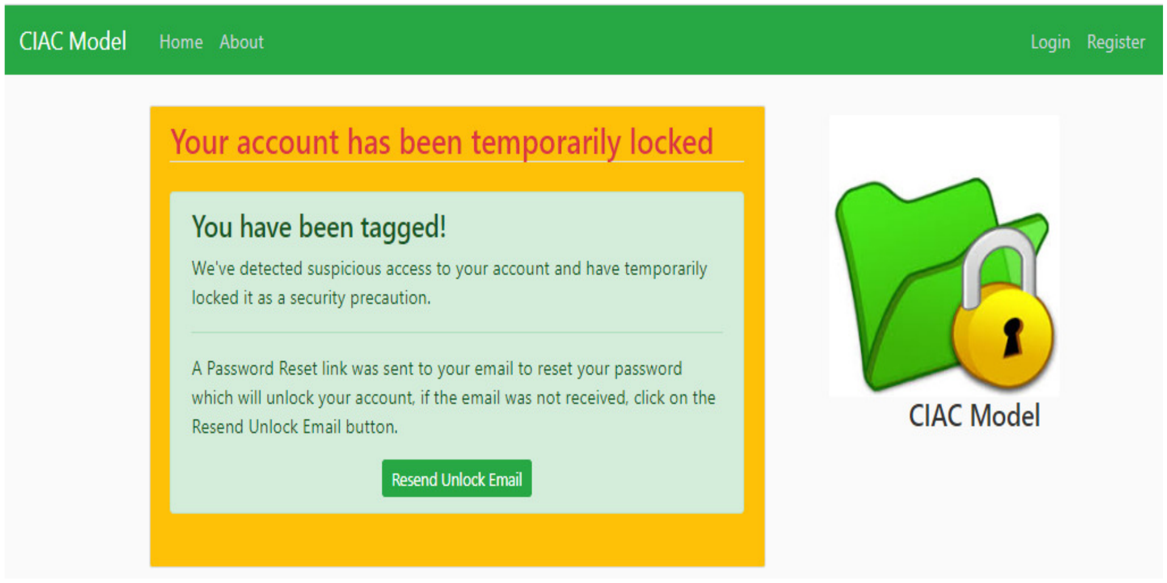
Log In

Email: rotimi.onakoya@uniabuja.edu.ng

Password:

Role: admin

Fig. 7: Context-Integrity Access Control Model Login for Correct Access Details message
Source: The proposed system by Onakoya et. al., (2020)



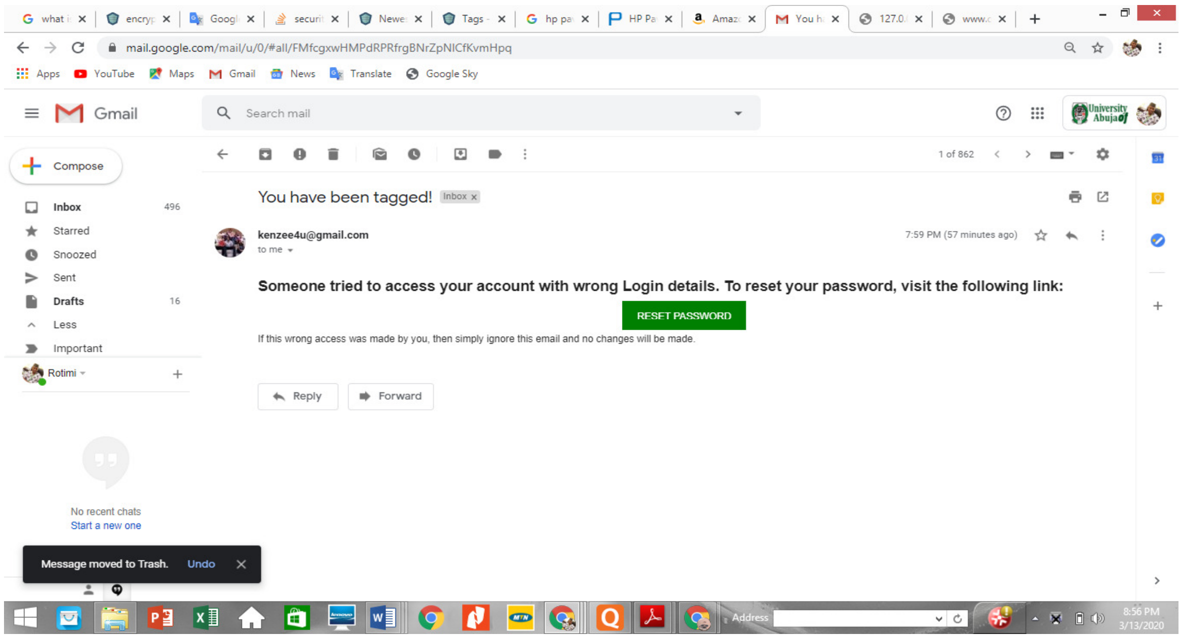
Your account has been temporarily locked

You have been tagged!

We've detected suspicious access to your account and have temporarily locked it as a security precaution.

A Password Reset link was sent to your email to reset your password which will unlock your account, if the email was not received, click on the Resend Unlock Email button.

Fig. 8: Context-Integrity Access Control Model error message dialogue box for Access Deny as a result of hybrid tag computation and response
Source: The proposed system by Onakoya et. al., (2020)



You have been tagged! Inbox

kenzee4u@gmail.com
to me
7:59 PM (57 minutes ago)

Someone tried to access your account with wrong Login details. To reset your password, visit the following link:

If this wrong access was made by you, then simply ignore this email and no changes will be made.

Fig. 9: Context-Integrity Access Control Model sent tagged email for Access Deny as a result of hybrid tag computation and response
Source: The proposed system by Onakoya et. al., (2020)

5. Conclusion

We want to say that we have adequately explained the context-integrity information security mechanism. We had extended traditional role-based model to a new security infrastructure called context integrity information security. Also modernized the traditional Username and password to valid email, Password and Role matching in Context-Integrity Access Control. The model is a user friendly; easy to use, based permission on the context-integrity decisions and return authorization. The new security system is secure based on the following proof [1]:

I. The context-integrity security system knows who has access
II. The context -integrity security mechanism knows who had access certain object
III. The context-integrity security measure quarantines intruders.
IV. The new system denies access to unauthorized users
V. The context-integrity security system protects data
VI. The context-integrity access control defends cyber sovereignty.
VII. The Context-Integrity Security system blocks Robotic breaches.

Based on the achieved objectives of this project, therefore, we have been able to solve insiders influence, Robotic and remote users' breaches into an academic web portal.

Future Work
We hope to work further on the Context-Integrity Access Control's Users Privileges in the near future.

References

[1] Balaban David (2020) "On Authorization and Implementation of Access Control Models" The State Security News. Trends. Insights. By Tripwire Inc. https://www.tripwire.com/state-of-security/security-data-protection/authorization-implementation-access-control-models/

[2] Banerjee, Anindya and Naumann, David A,(2004) "History-Based Access Control and secure Information Flow" International Workshop on Construction and Analysis of Safe, Secure, and Interoperable Smart Devices CASSIS 2004: Construction and Analysis of Safe, Secure, and Interoperable Smart Devices pp 27-48l https://link.springer .com/ conference/cassis or http://software.imdea.org/~ab/ Publications/hbacsif.pdf Downloaded on 27-02-2020

[3] Bourgeois, David (2019) "Information System for Business and BEYOND (web) Licensed under a creative Commons Attribution-Non Commercial 4.0. International License. August 1, 2019.

[4] DuraiPandian, N. Shanmughaneethi, V. and Chellappan (2006) "Information Security Architecture – Context Aware Access Control Model for Educational Applications. IJCSNS International Journal of Computer Science and Network Security, Vol. 6. No. 12 December, 2006.

[5] Hu, V.C., Kuhn, D. R. and Feraiolo, D.F. (2015) "Attribute-Based Access Control (ABAC) National Institute of Standards and Technology https://www.profsandhu.com/cs6393_s20/Hu-2015.pdf

[6] Kaushik Sharat, Tomar Anita and Poonam (2014) "Access Control List Implementation in a Private Network" International Journal of Information & Computer Technology ISSN 0974-2239 Volume 4, Number 14, 2014. Pp. 1361 @ International Research Publications House. http://www. irphouse.com

[7] Kulkarm Nahush, Kothari Hash, Ashar Hardik and Patil Sanchit (2005) "Access Control List" International Journal for Research in Applied Science & Engineering Technology (IJRASET) VOL.3. Issue XI, November 2015. ISSN: 2321-9653. www.ijraset.com . IC Value:13.98

[8] Prosser, Marc (2020) https://articles.bplans.com/4-easy-steps-increase-businesss-online-security/ Downloaded on 15-03-2020

[9] Martin, James A. and Walters, John K. (2018) "What is IAM? Identity and Access Management explained" https://www. csoonline.com/article/ 2120384/what-is-iam-identity-and-access-management-explained.html Downloaded 20-12-2019.

[10] Mudarri Tawfik, Al-Rabeei Samer and Abdo, Samer. (2015). SECURITY FUNDAMENTALS: ACCESS CONTROL MODELS. Interdisciplinarity in theory and practice. https://www.researchgate.net/publication/282219117_ SECURITY_FUNDAMENTALS_ACCESS_CONTROL_MODELS/citation/ Downloaded on 20-01-2020.

[11] Nisenbaum, Helen (2018): "Contextual Integrity Privacy" Symposium. New York University https://www.nyu.edu/ Projects/nisenbaum. August, 2018. Video downloaded in August, 2018.

[12] Pachghare V.K. (2009): "Cryptography and Information Security:, Asoke K. Ghosh, PHI Learning Private Limited, M-97, Connaught circus, New Delhi – 110001. ISBN: 978-81-203-3521-9.

[13] Rajebhosale Sagar, Choudhari Shashank, Patil Sachin, Vyavahare Akshay and Khabiya Sanket (2016) "SMART CAMPUS-An Academic Web Portal with Android Application" International Research Journal of Engineering and Technology (IRJET) eISSN:2395-0056, pISSN:2395-0072. Volume:03 Issue:04 April, 2016.

[14] Romuald Thion (2008) "Network-Based Passive Information gathering" International Outsourcing, Personal Data, and Cyber Terriorism. https://books. google.com.ng/books?id= XWK9AQAAQBAJ&pg=PT151&dq=Romuald+(2008)+enemies&hl=en&sa=X&ved=0ahUKEwiMgOHYpp3oAhUTUBUIHUSIBGwQ6AEIYTAH#v=onepage&q=Romuald%20(2008)%20enemies&f=false

[15] Sarvepalli, Vijay (2013) "Practical math for your Security Operations-Part1-3" Posted in Network Situational Awareness. August 6, 2013. https://insights.sei.cmu.edu/cert/ 2013/08/Practical-math-for-your-security-opertions----part-1-of-3.html

[16] Tan, Ying (2015) "Profile Matrix Permutation Strategy" pp.548 in Handbook of Research on Design, Control, and Modeling of Swarm Robotics.

https://books.google.com.ng /books?id=x_otCwAAQBAJ&pg=PA548&lpg=PA548&dq=Robotics+permutations&source=bl&ots=c2RaqOX4wF&sig=ACfU3U0DSlHw7PtHM1LODDRXg0cwE9w&hl=en&sa=X&ved=2ahUKEwiRn8y3lJ3oAhWL3oUKHZhpDhoQ6AEwCHoECAkQAQ#v=onepage&q=Robotics%20permutations&f=false edited by Tan, Ying

[17] Techopedia (2012) “Access Control List” (ACL) https://www.techopedia.com/definition/24766/access-control-list-acl downloaded on 14-03-2020

[18] Techopedia (2011) “Decretionary Access Control” (DAC) https://www.techopedia.com/definition/229/discretionary-access-control-dac Downloaded 2-3-2020

[19] Todd Birchard (2020) “Managing Database Models with Flask-SQLAlchemy” https://hackersandslackers .com/flask-sqlalchemy-database-models/ Downloaded on 27-2-2020.

[20] Servos, Daniel and Osborn, Sylvia I. (2017) ”Current Research and Open Problems in Attribute-Based Access Control” ACM Journal. ACM Computing Surveys Vol. 49, No. 4. https://doi.org/10.1145/3007204

[21] Wikipedia (2018 ’Mandatory Access Control” https://en.wikipedia.org/wiki/Mandatory_access_control Downloaded on August 21, 2019.

[22] W3School.com/Sql-join-full.asp

[23] Yalagi, pratibha S. and Dangare, Chaitrali S. (2013) “Design of An Academic Web Portal Providing E-Facilities ”International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR) ISSN 2249-6831. Vol. 3, Issue 1. March, 2013. pp. 85-90. ©TJPRC Pvt. Ltd.

[24] Zenlayer (2019) “What is an Access Control List (ACL)? https://www.zenlayer.com/blog/what-is-access-control-list/ downloaded on 15-03-2020

Authors’ Biographies

Onakoya, Johnson Rotimi is a Ph.D. Scholar, a Deputy Director, MIS at University of Abuja, Abuja. Nigeria. He has many publications to his credit. Member, British Computer Society, Computer Professional of Nigeria (CPN), Nigeria Computer Society and Nigeria Institute of Management.

Hammawa, Baba Mohammed is a Professor of Computer Science with bias in Information Security. Currently a Dean of Science in Baze University, Abuja. Nigeria. He has many publications both in International and local Journals to his credit. He has rich experience in Academic and Information Security world..