# OMICS GROUP

OMICS Group International through its Open Access Initiative is committed to make genuine and reliable contributions to the scientific community. OMICS Group hosts over **400** leading-edge peer reviewed Open Access Journals and organizes over **300** International Conferences annually all over the world. OMICS Publishing Group journals have over **3 million** readers and the fame and success of the same can be attributed to the strong editorial board which contains over **30000** eminent personalities that ensure a rapid, quality and quick review process. OMICS Group signed an agreement with more than **1000** International Societies to make healthcare information Open Access.

# OMICS Journals are welcoming Submissions

OMICS Group welcomes submissions that are original and technically so as to serve both the developing world and developed countries in the best possible way.

OMICS Journals are poised in excellence by publishing high quality research. OMICS Group follows an Editorial Manager® System peer review process and boasts of a strong and active editorial board.

Editors and reviewers are experts in their field and provide anonymous, unbiased and detailed reviews of all submissions.

The journal gives the options of multiple language translations for all the articles and all archived articles are available in HTML, XML, PDF and audio formats. Also, all the published articles are archived in repositories and indexing services like DOAJ, CAS, Google Scholar, Scientific Commons, Index Copernicus, EBSCO, HINARI and GALE.

**For more details please visit our website:**
**http://omicsonline.org/Submitmanuscript.php**

# Research Summary 2012 - 2014

Dr. Yen-Hung Hu, Department of Computer Science, Norfolk State University

# Outline

- Internet Traffic Pattern
  - A Fluid-Based Approach for Modeling Network Activities
- Attack Pattern Recognition
  - Building a Network Symptom Checker for Identifying Abnormal Network Activities
- Trustworthy Network
  - Challenges in Building a Trustworthy Network
- Secure Software Engineering
  - Adopting Secure Software Development Life Cycle in the Computer Science Capstone Projects

# A Fluid-Based Approach for Modeling Network Activities

- For a given network topology, if resources of every link and node are pre-defined and limited, behaviors of such a network will be bounded.
- Malicious flooding-based Denial of Service network activities are not isolated, but related as different stages of a series of cyber-attacks
- Normal and abnormal traffic can be simulated by fluid-based approach

# Research Procedures

- Analyze existing Internet traffics
  - 4 network traffic traces provided by the CAIDA (www.caida.org/data)
  - Statistics of protocol, connection, and overall are collected
- Adopt fluid-based approach for modeling TCP and UDP
  - TCP represents responsive traffic
  - UDP represents best effort traffic
- Tune models to comply with the gathered normal traffic characteristics.
- Study potential abnormal traffics affecting the stability of normal traffics.

# Existing Internet Traces – Packet Level Analysis

Table 1: The Selected Traffic Traces

| Traffic | File Name |
|---|---|
| 2009-01 | Equinox-sanjose.dirB.20090115-130000.UTC.canon.pcap |
| 2009-02 | Equinox-sanjose.dirB.20100121-130000.UTC.canon.pcap |
| 2009-03 | Equinox-sanjose.dirB.20110120-130000.UTC.canon.pcap |
| 2009-04 | Equinox-sanjose.dirB.20120119-130000.UTC.canon.pcap |

Table 3: Percentage of Packets in the Selected Traffic Traces

| | 2009-01 | 2010-01 | 2011-01 | 2012-01 |
|---|---|---|---|---|
| TCP | 85.95% | 88.09% | 78.26% | 86.16% |
| UDP | 10.39% | 9.07% | 20.15% | 10.07% |
| Others | 3.66% | 2.84% | 1.59% | 3.77% |

Table 2: Composition of Packets in the Selected Traffic Traces

| | 2009-01 | 2010-01 | 2011-01 | 2012-01 |
|---|---|---|---|---|
| TCP | 12261116 | 16456196 | 24846485 | 26542956 |
| UDP | 1482744 | 1694519 | 6396887 | 3101842 |
| Other | 522407 | 530530 | 505827 | 1159979 |

Table 4: Connection Number and Percentage in Traffic Trace 2009-01

| | All | TCP | UDP | Others |
|---|---|---|---|---|
| Number | 595611 | 277697 | 237927 | 79987 |
| Percentage | 100% | 46.62% | 39.95% | 13.43% |

# Existing Internet Trace – Connection level Analysis

Table 5: Connection with Life ≤ X Seconds in Traffic Trace 2009-01

| X | TCP Connection | UDP Connection | Other Connection |
|---|---|---|---|
| 1 | 25.03% | 77.93% | 68.53% |
| 10 | 44.13% | 89.88% | 76.42% |
| 20 | 50.19% | 93.69% | 80.69% |
| 30 | 54.48% | 96.05% | 85.13% |
| 40 | 86.43% | 97.51% | 90.05% |
| 50 | 91.91% | 98.66% | 94.52% |

Table 6: Connection with Size ≤ X Packets in Traffic Trace 2009-01

| X | TCP Connection | UDP Connection | Other Connection |
|---|---|---|---|
| 1 | 76.66% | 97.67% | 93.40% |
| 100 | 94.62% | 99.65% | 99.36% |
| 200 | 96.90% | 99.77% | 99.73% |
| 300 | 97.79% | 99.82% | 99.85% |
| 400 | 98.29% | 99.85% | 99.89% |
| 500 | 98.59% | 99.87% | 99.92% |

Table 7: Hurst Parameter of Four Different Streams in the Selected Traffic Traces

| Traffic Trace | All-stream | TCP-stream | UDP-stream | OTHERS-stream |
|---|---|---|---|---|
| 2009-01 | 0.84 | 0.84 | 0.55 | 0.55 |
| 2010-01 | 0.75 | 0.76 | 0.55 | 0.59 |
| 2011-01 | 0.81 | 0.81 | 0.62 | 0.73 |
| 2012-01 | 0.80 | 0.80 | 0.69 | 0.72 |

# Network Modeling

- Single Congested Network
- Normal Traffic
- Multiple Connections
- Malicious Traffic

# Some Results

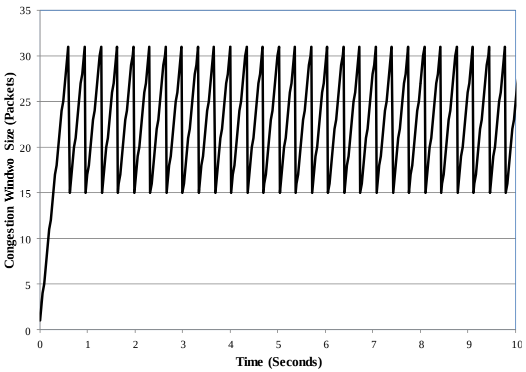



Figure 1: Congestion Window Size of a TCP Connection vs. Time

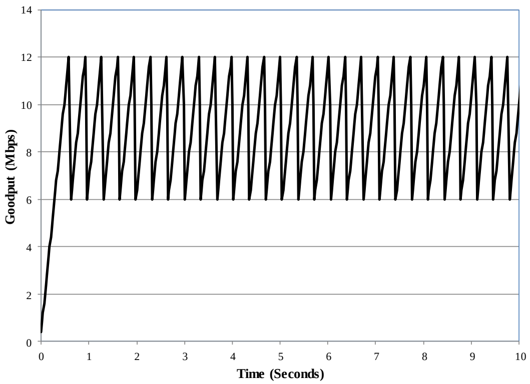



Figure 2: Goodput of a TCP Connection vs. time

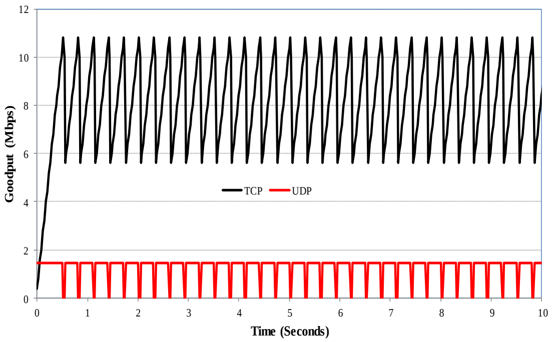



Figure 3: Goodput of a TCP Connection and an UDP Connection vs. Time

# Future Works

- Study more network models (e.g., a network with multiple congestion points) to study performance of the simulated traffic
- Study other malicious activities and to evaluate their influences as well

# Building a Network Symptom Checker for Identifying Abnormal Network Activities

- Let graph G = (V, E) represent a network, where $V = \{v_1, v_2, ..., v_n\}$ denotes nodes and $E = \{e_1, e_2, ..., e_m\}$ denotes edges in the graph G
- Network activities can be modeled by mathematical theory of dynamical system
- since resources (i.e., bandwidth, memory, services capacity, etc.) of the monitored network are limited, it is expected that trajectories of every network characteristics of the network G during the monitoring period are bounded

# Research Procedures

- Use dynamic system to model network activities
- Build network symptom checker

# Modeling Network Activities - 1

- Let $C = \{c_i | 0 \leq i \leq c_{max}\}$ represent a set of network characteristics, $\varphi = \{\varphi_i | 0 \leq i \leq c_{max}\}$ is a set of state spaces associates with *C*, and $c_{max}$ is the number of network characteristics that will be monitored
- When the evolution is deterministic, then for each time $t$, it is observed that $(t) = \varphi^t(C^0)$ , where $\varphi^t$ represents a set of state space of the evolution at time *t* and $C^0$ represents initial state of *C*
- Therefore, any given network characteristic *i* of the network at time *t* can be represented as $c_i(t) = \varphi_i^t(c_i^0)$

# Modeling Network Activities - 2

- Let $Q = \{q_i | 0 \leq i \leq c_{max}\}$ represents a set of three different network conditions (*i.e.*, 1: safe, 0: marginal, -1: unsafe), $q_i$ represents three different conditions of $c_i$, and $q_i = -1, 0, or\ 1$
- When $q_i = 1$, $c_i$ is in the safe zone. When $q_i = 0$, $c_i$ is in the marginal zone. When $q_i = -1$, $c_i$ is in the unsafe zone
  - Safe (1): if the current state is in this condition, the network characteristic will remain in this condition unless there is major change in the network
  - Marginal (0): if the current state is in this condition, the network characteristic will end up in an unsafe condition. Appropriate warnings and controls need to take place to bring the network characteristic back to safe condition
  - Unsafe (-1): if the current state is in this condition, the network characteristic will remain in this condition. In this case, administration actions need to be issued to stop the services that cause this result

# Modeling Network Activities - 3

- We then introduce two vectors, $W = \{w_i | 0 \leq i \leq c_{max}\}$ and $Z = |W \cdot Q|$, to represent weight factor of network characteristics and weighted network condition of the network
- If $Z > R$, the network is normal, where $R$ is a threshold related to network information. Otherwise, the network is abnormal

# Network Symptom Checker

- A recognizable 3D surface
- Apply a mapping function *M* on *{C, Q, W, Z, R}*, and the network symptom checker could be represented as *M(C, Q, W, Z, R)*
- To make it possible to attract potential users, this network symptom checker must be easy to use and identify malicious activities of the monitored network
- It should be as easy as we read information from the face of an illness patient

# Future Works

- Need to evaluate every parameter motioned

# Challenges in Building a Trustworthy Network

- There is no mature implementation of building a trustworthy network although the concept has been discussed more than ten years
- We have observed that trustworthiness could not be achieved if there is no proper integration of major network components

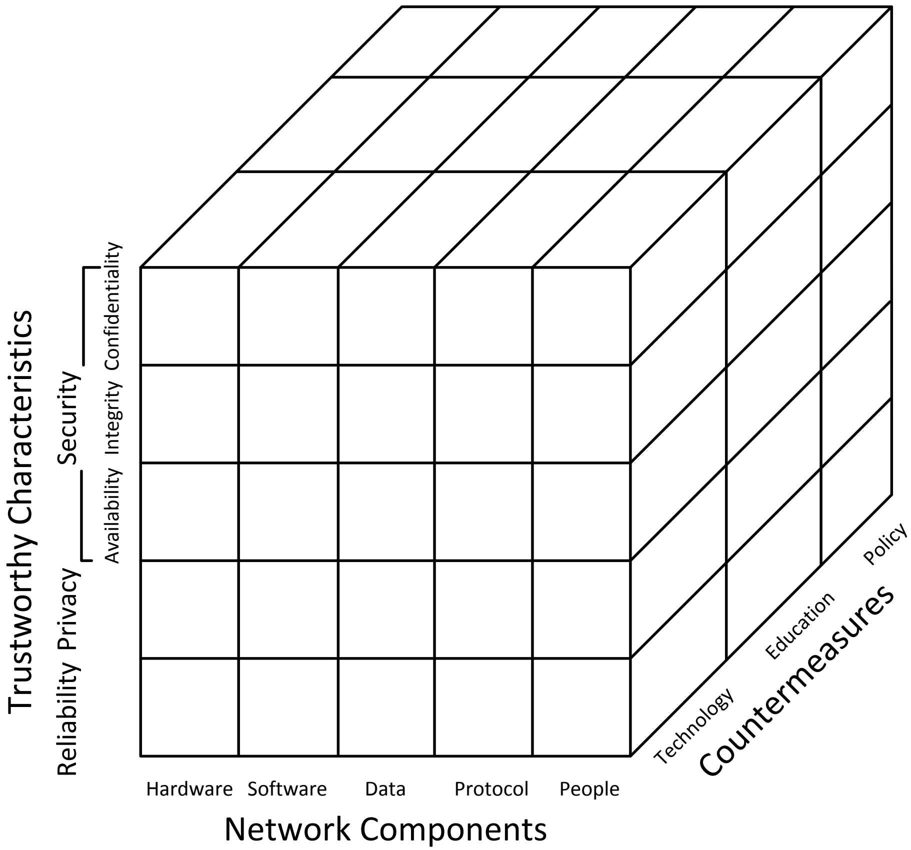
# Examples

- Difficulty in verifying network components:
  - The biggest challenge of the implementation is in verifying network components to ensure they are capable of protecting security, privacy, and reliability
- Difficulty in administrating network components:
  - Since network components usually cross several different domains and are managed by various administrations, the operation to ensure the protection of network security, privacy, and reliability is very complicated and difficult to achieve
- Difficulty in protecting data crossing over different network components:
  - Although every network component could implement its own mechanisms to protect network privacy, security, and availability, there are some potential threats could exist in the gap between two components

# Trustworthy Network Model

- This 3D model includes three axes and 75 cells (*i.e.*, 5 x 3 x 5)
- X-axis: This axis introduces five major network components
  - Hardware, software, data, protocol, and people
- Y-axis: This axis covers three countermeasures
  - Technology, education, and policy
- Z-axis: This axis represents five trustworthy characteristics
  - Reliability, privacy, confidentiality, integrity, and availability

# Trustworthy Network Model



# X-Axis: Countermeasures

- Technology:
  - Technical solutions, no matter hardware or software, relating to the protection of network security, privacy, and reliability are included in this category
- Policy:

  Policies are regulations and rules in the workplace relating to the protection of network security, privacy, and reliability
- Education:
  - Education includes formal and informal training programs relating to the protection of network security, privacy, and reliability

# Y-Axis: Trustworthy Characteristics

- Security:
  - The assessment of a security implementation is in the measurement of the degree of protecting information confidentiality, integrity, and availability
- Privacy:
  - You shall gain control over your own information. Others involving in using your information shall adhere to fair information principles
- Reliability:
  - In brief, reliability means availability and correctness. Systems for providing services must be always available and correct and commit to fulfill every request from the legitimate users

# Z-axis: Network Components

- Hardware:
  - Hardware is the physical technology that stores, processes, and transmits data; executes software; interacts with applications and operating systems; and compromises with protocols
- Software:
  - The software component of the network includes operating systems, applications and utilities
- Data:
  - the data component of the network indicates any form of information appearing in the network
- Protocol:
  - Protocol indicates criteria and mechanisms used in the network communication.
- People:
  - We may often overlook this topic. People have always been a threat to the network security

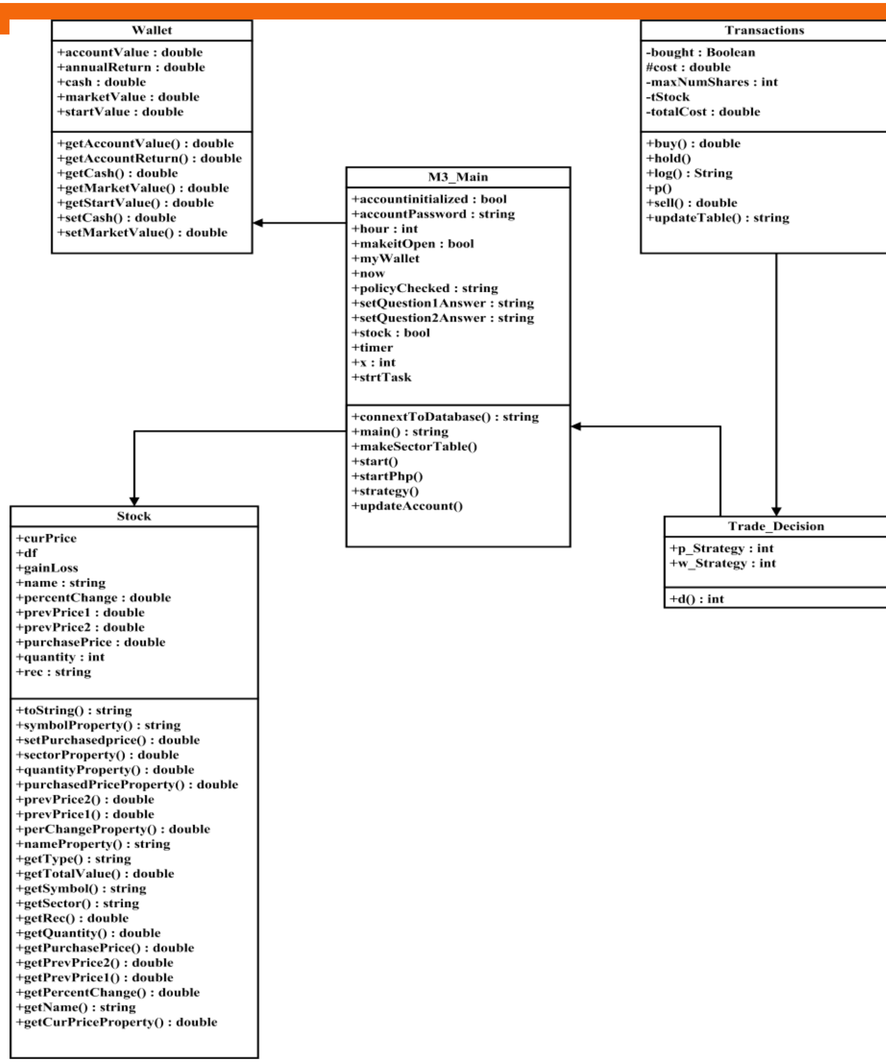
# Adopting Security Software Development Life Cycle in the Computer Science Capstone Projects

- Use McCumber's Cube model to assess components in the selected capstone program
  - confidentiality, integrity, and availability

# Research Procedures

- Select/build a Java security guideline
- Link this Java security guideline to McCumber's Cube model to form an assessment table
- Use the assessment table to evaluate every component in the selected capstone program

# Sample Capstone Project – 4 Loop Product



# The Assessment Table

| Critical Questions | Explanation | McCumber Cube: Confidentiality, Integrity, Availability |
|---|---|---|
| Q.1. Limit the accessibility of classes, interfaces, methods, and fields | Use an access modifier to limit their accessibility. The four access levels are:<br>• Default: visible to the package. No modifiers are needed.<br>• Private: visible to the class only<br>• Public: visible to the world<br>• Protected: visible to the package and all subclasses. | If wrongly declared, data confidentiality, integrity, and availability may be violated. |
| Q.2. Use a try-with-resources statement to safely handle closeable resources. | The try-with-resources statement ensures that each resource is closed at the end of the statement. | When resources are not closed properly, data confidentiality, integrity, and availability may be violated. |
| Q.3. Avoid using try-catch-finally block. | Ordinary try-catch-finally block can raise some issues: such as failing to close a resource because an exception is thrown as a result of closing another resource, or masking an important exception when a resource is closed. | When resources are not closed properly, data confidentiality, integrity, and availability may be violated. |
| Q.4. Use the same type for the second and third operands in conditional expressions. | Use different types in a conditional expression may cause unintended type conversions. | When types are not the same, data integrity may be violated. |
| Q.5. Avoid using static field variables. | Static variables are class variables, not instance variables. Science any other class in the same scope can access the static variables it is very difficult to secure them. | Since static variables can be modified by other classes in the same scope, data integrity may be violated. |
| Q.6. If possible make public static fields final | Otherwise, attacker may change the value. | If value changed, data integrity may be violated. |
| Q.7. If possible, use immutable objects. | Contents of the mutable object can be changed. | If contents changed, data integrity may be violated. |
| Q.8. Avoid storing user-given mutable objects directly. | Contents of the mutable objects can be changed. Clone the objects before processing them internally. | If contents changed, data integrity may be violated. |
| Q.9. Avoid using inner classes. | After compilation, any class in the package can access the inner class. Meanwhile, private filed of the inner class will be converted into non-private to permit access by the inner class. | If other classes in the package can access the inner class, data confidentiality and integrity may be violated. If private filed of the inner class changed, data integrity may be violated. |
| Q.10. Avoid using the clone() method to copy untrusted method parameters | Inappropriate use of the clone() method can allow an attacker to exploit vulnerabilities by providing arguments that appear normal but subsequently return unexpected values. Such objects may consequently bypass validation and security checks. (Oracle, Secure Coding Guidelines for the Java Programming Language, Version 4.0) | Data confidentiality and integrity may be violated. |
| Q.11. Make secure classes uncloneable. | Otherwise, malicious developers can instantiate a class without running its constructors. (Sinn, 2008) | This may violate data confidentiality and integrity. |
| Q.12. Avoid embedding sensitive information | Malicious developers can obtain such a sensitive information | This may violate data confidentiality. |
| Q. 13. Prevent constructors from calling methods that can be overridden | Constructors that call overridable methods give attackers a reference to the object being constructed before the object has even been fully initialized | This may violate data integrity |

# Observation

| Critical Questions | Reasons Causing Vulnerabilities | Recommended Solutions |
|---|---|---|
| Q.1 | Classes are public | The product should make all classes package-private, since they are in the same package (M3Application) and not served as an API or interface for external classes. |
| Q.1 | Methods are public | This implementation is not appropriate. Methods in the product should obtain at least default access modifier privilege. Most of them should be in private access modifier privilege, since they are used internally. |
| Q.1 | Variable are public | Variables should limit the accessibility. In this product, most variable should be in private access modifier privilege. |
| Q.2 | Resource is not proper closed | This product should use try-with-resources statement to ensure each resource is closed at the end of the statement. |
| Q.2 | More than one resource operations in the try-catch-finally block | Since several exceptions may be thrown, some exceptions will be masked. |
| Q.5 | Static variables | Since other classes in the same scope may be able to access and modify a static variable, this variable should better be claimed as final static. |
| Q.9 | Inner class | Since there are some security issues related to inner class. It is better to move inner classes to outer classes. |
| Q.12 | Sensitive information | Use Java security APIs to handle sensitive information. |

# Future Works

- Apply the assessment to more capstone programs

# Acknowledgement

- Dr. Yen-Hung Hu is a faculty member in Department of Computer Science at Norfolk State University. Before joined the Norfolk State University, he was the director of the Information Assurance Center at Hampton University and a full time faculty member in the Department of Computer Science at Hampton University. Dr. Hu has participated in several funded grants including two NSF awards. He has attended a number of Information Assurance and Cyber Security related research and education workshops including NSA IA CAE principals meeting, NIST Cryptographic Key Management workshop, Colloquium for Information Systems Security Education, and other NSF funded workshops. His research focuses on computer and network security. He has authored/co-authored more than 20 publications and made numerous professional presentations.

# Journal of Electrical & Electronic Systems

- Electrical Engineering & Electronic Technology
- Telecommunications System & Management
- Information Technology & Software Engineering

# Electrical & Electronic Systems Related Conferences

- 2nd International Conference and exhibition on Lasers optics and Photonics.

# OMICS Group **Open Access Membership**

OMICS publishing Group Open Access Membership enables academic and research institutions, funders and corporations to actively encourage open access in scholarly communication and the dissemination of research published by their authors.

For more details and benefits, click on the link below:

http://omicsonline.org/membership.php

# Thank you