• Cloud computing vulnerabilities, and risk factor
• Developing metrics for quantifying risk and resilience in cloud computing
  – Determining asset dependency and criticality, criticality surface
  – Modeling with multi-disciplinary approaches (e.g., control theory, time Petri net, Bayesian network)
• Controllability and Observability
• An example for identifying critical nodes and controllability
Cloud computing security requires defense at both server and client sides
- Some subsystems of cloud are not controlled by cloud subscribing organizations

Metrics are needed to measure and verify the correct functioning of a subsystem and effectiveness of security controls

How does cloud computing affect the models and metrics development on risk, resilience, and criticality surface?
Risk Factors in Cloud Computing

- ISO 27005 defines risk as “the potential that a given threat will exploit vulnerabilities of an asset or group of assets and thereby cause harm to the organization”
- Risk management assesses risk and aims at reducing to an acceptable level
- How does cloud computing affect risk factors?
  - Risk factors
    - The likelihood of a harmful event (or loss event frequency as per ISO 27005)
    - The consequence of an harmful event
What are cloud computing influence on risk factors from both customer and service provider perspectives?

* www.opengroup.org/onlinetpubs/9699919899/toc.pdf
Problem Statement

- Given **incomplete** information on vulnerabilities, threats, attacks, fully/partially compromised nodes, topology, types of hosts and servers over a cloud system (e.g., hybrid cloud, private cloud, public cloud, community cloud), determine how to accurately and efficiently
  - Assess dynamically the risk and resilience of the system,
  - Steer the system with some compromised nodes towards a system without compromised nodes by
    - Implementing recovery and resilience operations on the partially/fully compromised nodes or assets,
    - Strengthening resilience of the system
Consider a network of $N$ nodes/clients that are served by a cloud, where $N = C+E+F+G+V$ and
- $G(t)$ denote the number of those nodes that do not have any known vulnerability at time $t$
- $V(t)$ denote the number of those nodes that have some known vulnerabilities at time $t$, but are not exploited yet
- $C(t)$ denote the number of those nodes that are compromised partially/fully through the exploitation of their vulnerabilities
- $E(t)$ denote the number of those nodes that are evicted due to that they cannot be recovered
- $F(t)$ denote the number of those nodes that have failed and do not operate due to physical failures
- We can control the states and operation of nodes by $r(t)$ and $p(t)$.
- We can measure $C(t)$ and $V(t)$

- $o$: vulnerability occurrence rate
- $p$: vulnerability patching rate
- $e$: vulnerability exploitability rate
- $r$: recovery rate
- $d$: cyber compromised-node eviction rate
- $f$: physical failure rate
An Example of Cyber-Physical System (cont’d)

- The dynamics of the system can be described by the following differential system:
  - \( \frac{dG}{dt} = pV - oG - fG \)
  - \( \frac{dV}{dt} = oG + rC - pV - eV - fV \)
  - \( \frac{dC}{dt} = eV - rC - dE - fC \)
  - \( \frac{dE}{dt} = dC \)
  - \( \frac{dF}{dt} = f(G + V + C) \)

- \( u(t) = (r(t), p(t)) \)
- \( y(t) = (C(t), V(t)) \)

- \( o \): vulnerability occurrence rate
- \( p \): vulnerability patching rate
- \( e \): vulnerability exploitability rate
- \( r \): recovery rate
- \( d \): cyber compromised-node eviction rate
- \( f \): physical failure rate
A dynamical system like a cloud system can be modeled as a linear time-invariant dynamic system

\[
d\mathbf{x}(t)/dt = A \mathbf{x}(t) + B \mathbf{u}(t)
\]

\[
y(t) = C \mathbf{x}(t)
\]

where

- the vector \( \mathbf{x}(t) = (x_1(t), \ldots, x_N(t))^T \) captures the state of a network with \( N \) nodes at time \( t \).
- The \( N \times N \) matrix \( A \) describes the network’s connectivity and interaction strength between nodes.
- The \( N \times M \) input matrix \( B \) identifies the nodes controlled by a time-dependent input vector \( \mathbf{u}(t) = (u_1(t), \ldots, u_M(t))^T \) controlled by an outside controller, where \( M \leq N \).
• Control-theory offers a powerful mechanism
  • to deal with disturbances, unpredictable changes, and uncertainties
  • to model, analyze, and design resource control and feedback systems

• Controller aims to maintain the difference between the reference input (e.g., performance targets as desired values) and the measured output (e.g., measured performance metrics), in spite of disturbance, noise, or attack that are not under control
• **Controllability** is related to the ability of forcing the system into a particular state by using an appropriate control signal
  – A system is **controllable** if it can be driven in a desired direction, so that it will perform according to the specifications when inputs change
  – If the $N \times NM$ controllability matrix $(B, AB, A^2B, \ldots, A^{N-1}B)$ has full rank (i.e., rank $N$), then the system is controllable, according to Kallman’s controllability rank condition

• **Observability** is related to the possibility of observing states of a system via output measurements
  – The state equation is **observable** if for any input state $x_0$ and for any input signal $u$, finite the output $y$ sequence determines uniquely $x_0$
  – The pair $(A,C)$ or the system is observable if the observability matrix has full rank $N$. 

\[
\begin{bmatrix}
  C \\
  CA \\
  CA^2 \\
  \vdots \\
  CA^{N-1}
\end{bmatrix}
\]
• Attacker’s input \( u_a \), process noise \( w_p \), and measurement noise \( w_m \) can be added to a linear time-invariant dynamic system

\[
\frac{dx(t)}{dt} = A \, x(t) + B \, u(t) + u_a(t) + w_p(t)
\]

\[
y(t) = C \, x(t) + B \, u(t) + w_m(t)
\]
A system is called **observable** if its complete internal state can be reconstructed from its outputs.

**Question:** Is it possible to measure (simultaneously) all state variables?
- What if only a subset of state variables can be measured in practice?

**Problem:** How to identify minimum set of sensors whose measurements can allow us to determine all state variables
- What is the impact of noise and measurement uncertainties on determining the minimum number of sensors?

(This problem is still not solved*, though it may be possible to compute a lower bound for the number of system variables)

---

• A system is called **controllable** if it can be driven from any initial state to any desired final state in finite time using a suitable choice of input
  – This is like capturing an ability to guide a system’s behavior towards a desired state via proper manipulation of input variable*

• **Problem 1**: How to identify the minimum number of driver nodes to achieve controllability
  – How to characterize and predict driver node

• **Problem 2**: How to avoid having perfectly (or nearly-perfectly) attackable systems
  – How to distinguish a compromised system from a healthy system

An Example for Identifying Critical Nodes and Controllable Input Signals
Determine individual risk, resilience, and control effectiveness of cyber assets within a node using various techniques, including a Bayesian network:

- **P(A)**, **P(B)**: probability that node receives various threats.
- **P(C)**, **P(D)**: probability that node has vulnerabilities that can be exploited by threats.
- **P(E)**: probability that node receives aggregated threat.
- **P(F)**: probability that node has aggregated vulnerabilities that can be exploited by threats.
- **P(G)**: probability that vulnerability can be exploited at node.
- **P(H)**: probability that node is infected despite the presence of control mechanisms.
- **P(I)**: probability that node has proper internal/external recovery mechanisms for mitigating node’s infection.
- **P(J)**: probability that node infection is mitigated.
Identifying Critical Nodes based on Network Connectivity and Attributes

- Enable network connectivity to be considered along with node attributes (e.g., resilience, control, influence) in identifying and classifying critical nodes
- We develop **Hierarchically well-Separated Tree with Attributes (HSTA)**, $\alpha$-HSTA
  - obtained through Hierarchically well-Separated Tree (HST) to make it depend on not only the hops but also the attributes of nodes
  - HST is usually used for resource matching and management
- **Construct** a Hierarchically Well-Separated Tree (HST) for a network of nodes
  - $\alpha$-HST: a weighted tree
    - The edge weights from the root to leaf decrease by a factor of $\alpha$
    - All root-to-leaf paths have the same hop distance
    - The weights from each node to its children are the same
  - For construction of $\alpha$-HST, centralized (top-down approach) or distributed (bottom-up approach) algorithm is used
  - **2-HST** is considered often, where $\alpha = 2$
- Given (i) thresholds $T_r = 0.05$, $T_c = 0.10$ and $T_i = 0.08$, (ii) attribute values range from 0.1 to 1, where 1 is highest desirable attribute value, (iii) primary attribute values (i.e., resilience values) with subscript $R$ are shown in graph.
- All the nodes of this network are considered to be the leaves level (or level 1).
Flood each node 1-hop to determine the ancestors of all the leaves nodes
• compare resilience values, and choose highest one (shown with arrows in blue) if the difference of their resilience values is greater than $T_r = 0.05$; otherwise, consider their control values
  – For instance, note the difference between resilience values of $N_{10}$ is $N_{11}$ is less than 0.05
  – Compare their control attribute values (i.e., 0.60 and 0.45), so $N_{11}$ does not nominate $N_{10}$

Nominated nodes at level 1 are: $N_2, N_5, N_6, N_9, N_{11}, N_{15}, N_{16}, N_{17}, N_{20}$
• For each node:
  – flood 2-hops to determine potential ancestors
  – compare resiliency values, and choose the highest resilience value
  – nominate level 2 ancestors (nodes with labels in red)

Nominated nodes at level 2 are: N6, N9, N16
Flood 4-hops, and nominate level 3 ancestors in 2-HSTA

- For each node:
  - flood 4-hop to determine potential ancestors
  - compare resilience values, and choose the highest resilience value
  - nominate level 3 ancestors (node label in green)

Nominated nodes at level 3 are: N6
Let the critical nodes of level 2 have external inputs controlled by an outside controller: $N_6$, $N_9$, $N_{16}$ (called “driver nodes” for network controllability)

- So, the external inputs $u_6(t)$, $u_9(t)$, and $u_{16}(t)$ are non-zero, and all the other external inputs of nodes are zero. This implies $b_6$, $b_9$, and $b_{16}$ are non-zero.

Compute the weights of resiliency dependencies or influences $a_{ij}$ and $a_{ji}$ between neighboring nodes $i$ and $j$, for all nodes, where $a_{ij}$ is the weight associated with the directed edge from $j$ to $i$ in the graph.

The same type of weights could be computed for other types of influences, namely, control, security, and vulnerability.
Consider a network comprising $N$ nodes or assets, where each node $i$ is associated with a status vector $\mathbf{x}_i(t)$ that describes the physical properties of the node at time $t$.

Define a directed graph $G$ with $N$ vertices such that a directed edge from vertex $i$ to vertex $j$ indicates that the status of node $j$ directly depends on the concurrent status of node $i$. An edge from a vertex back to itself indicates that the status change of the vertex depends on its current status.

The status trend or update of a node depends on the current status of itself, neighbors, and local input.
Network Controllability Support by Resilience Critical Nodes (cont'd)

The N x NM controllability matrix:

\[ C = \begin{bmatrix} B, AB, A^2B, \ldots, A^{N-1}B \end{bmatrix} \]

If rank(C) = N, system is controllable

\[
A = \begin{bmatrix} \begin{array}{ccc} a_{11} & \cdots & a_{1,20} \\ \vdots & \ddots & \vdots \\ a_{20,1} & \cdots & a_{20,20} \end{array} \end{bmatrix}
\]

\[
B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \\ b_6 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & b_9 & 0 \\ \vdots & \vdots & \vdots \\ 0 & b_{16} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]

\[
B^T = \begin{bmatrix} 0 & 0 & 0 & b_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{16} & 0 & 0 \end{bmatrix}^T
\]
Equiangular polygon

Consider metrics, $M_i$, $i > 1$

As an example, let $i=6$

- $M_1$: risk
- $M_2$: resilience
- $M_3$: control
- $M_4$: influence
- $M_5$: recovery
- $M_6$: robustness

Angle: $360/6 = 60$ degree

Each metric

initially, has its own unit and range

then, is normalized between 0 and 1

The (2-dimensional) area of polygon is defined as being the criticality surface of its metrics
• Determine coordinates of equiangular polygon
• Consider a polygon with N-1 sides and N vertices \((x_i,y_i), i=0\) to \(N-1\), such that the first vertex \((x_0,y_0)\) and the last vertex \((x_N,y_N)\) are the same
• Area of this polygon is given by

\[
\text{Area} = \frac{1}{2} \sum_{i=0}^{N-1} (x_i y_{i+1} - x_{i+1} y_i)
\]
Questions?