**John Carroll – jcc0044/902521946 – Midterm exam 2**

**VULNERABILITY**: security flaw: failure of security policies, procedures, and controls that allow a subject to commit an action that violates the security policy to detect it, use formal verification and property based testing. In **formal verification**, the preconditions place constraints on the state of the system when the program is run and the postconditions state the effect of running the program. 1) Mathematical proof that system satisfies constraints 2) Preconditions state assumptions about the system 3) Postconditions are result of applying system operations to preconditions, inputs 4) *Required*: postconditions satisfy constraints.***( For formal verification to prove absence, proof and preconditions must include all external factors***). In **penetration testing**, the preconditions describe the state of the system in which the hypothesized security flaw can be exploited and postconditions are result of the testing. It is a testing technique. 1. Testing to verify that a system satisfies certain constraints 2. Hypothesis stating system characteristics, environment, and state relevant to vulnerability 3. Result is compromised system state 4. Apply tests to try to move system from state in hypothesis to compromised system state. **For penetration Testing**: 1) Test for evaluating the strengths of all security controls on system (aim is to violate site security policy. Its aka tiger team attack/red team attack/white hat). 2) Two types of studies, with slightly differing goals**. 2.1** Attempt to violate specific constraints **2.2** No specific target – attempt to find some number of vulnerabilities within a period of time . **Layering of Tests 1.** External attacker with no knowledge of system (Locate system, learn enough to be able to access it) 2. External attacker with access to system (Can log in, or access network servers & Often try to expand level of access) 3. Internal attacker with access to system (guest/regular user/admin). **False Hypothesis Methodology 1.** Info gathering 2. Flaw hypothesis. 3. Flaw testing 4. Flaw generalization 5. Flaw elimination. RISOS classified flaws in 7 categories: 1) Incomplete parameter validation 2) Inconsistent parameter validation 3) implicit sharing of privileged/confidential data 4) inadequate identification/authentication/authorization 5) Violable prohibition/limit 6) Exploitable logic error. **Advantages of classification** 1) can lead to detection of similar flaws 2) demonstrates potential impact 3) structured & disciplined way to document flaws. **Problems** 1) frameworks not standerized 2) classififcation is subjective 3) vulnerability may belong in multiple flaw classes **NRL Vulnerability Taxonomy:** Flaw by genesis Flaw by time of introduction Flaw by Location

**Common Flaws 1) Improper protection domain** 1.1 Improper choice of initial protection domain 1.2 Improper isolation of implementation detail 1.3 Improper change 1.4 Improper naming 1.5 Improper deallocation or deletion 1.6 Improper validation **2) Improper Synchronization** 2.1 Improper indivisibility 2.2 Improper sequencing **3) Improper choice of operand or operation 3.1** calling inappropriate or erroneous instructions. **Reverse Engineering** Belief: executable format = secure Reality: Machine code analysis/ Memory Dumps/Reverse Engineering Tools/ Debuggers usually have disassemblers/ Decompilers are not as mature as disassemblers. **Targeted protection belief:** security reality: anti-tampering can be circumvented and DRM mechanism **Code Obfuscation** Belief: unreadable code = security Reality: slows down vulnerability analysis, but doesn't prevent it. Common obfuscation techniques 1. Add code that never executes or that does nothing 2.Make calculations more complex 3.Move code around. **Attacker’s Choice:** Old code/ Code that runs by default /Code that runs in elevated context/ anonymously accessible code /Code listening on a globally accessible network interface. **Development Best Practices**: **DESIGN** 1.Make security-sensitive parts of code small 2. Least privilege principle: Don't require more 3. Choose safe defaults 4. Deny by default 5. Limit resource consumption 6. Fail securely 7. Don't trust the client in a tiered architecture 8. Use trusted, public algorithms, protocols and products. 9. Don't trust input data 10. Beware of race condition 11. Don't make any assumptions about the environment 12. Validate all input data **CODING** 1. Deal with errors and exceptions 2. Fail gracefully 3. Protect passwords and secret information 4. Consciously handle files 5. Avoid temporary files where possible 6. Use shell calls, eval functions etc. sparingly 7. Avoid stubs 8. Enable compiler checks 9. Disable debug information. **Planning/Disaster Recovery** agency response procedures and continuity of operations development of plans for recovery actions after a disruptive event. Development of procedures for off-site processing. **Physical Security Measures**: alarms /building construction /cabling/communications centers/ environmental controls (humidity and air conditioning)/filtered power /fire safety controls. **Software Security** 1) assurance 2)configuration management 3)software security mechanisms to protect Information. **Auditing and Monitoring 1)** conducting security reviews 2)effectiveness of security programs 3)investigation of security breaches 4)review of accountability controls 5)verification, validation, testing, and evaluation processes **BUFFER OVERFLOW**: A buffer overflow occurs when data written to a buffer also corrupts data values in memory addresses adjacent to the destination buffer due to insufficient bounds checking. This can occur when copying data from one buffer to another without first checking that the data fits within the destination buffer Almost any C function that can read in a character is a candidate for an overflow. **STACK** **OVERFLOW**: Find a stack-allocated buffer we can overflow that allows us to overwrite the return address of the stack frame. Place some hostile code in memory to which we can jump when the function we’re attacking returns. Write over the return address on the stack with a value that causes the program to jump to our hostile code.  
**HEAP OVERFLOW VS STACK OVERFLOW**: Common goal of overflow attacks are root shells; Attacks are typically against a particular architecture (OS/machine combination); One common technique is to find a buffer overflow in a suid program; Heaps are harder to exploit because they are dynamic, not static; programming strategy is to new or malloc() everything main protection is that fewer people know how to exploit heap overflows. Generally takes longer to set up a heap overflow attack. **CONFINEMENT PROBLEM**: Problem of preventing a server from leaking information that the user of the service considers confidential. Confinement is a mechanism for enforcing the principle of least privilege. the confinement mechanism must distinguish between transmission of authorized data and transmission of unauthorized data. **TOTAL ISOLATION**: Process cannot communicate with any other process & Process cannot be observed (by Lampson) **Lipner’s Notes:** All processes can *obtain rough idea of time* (by reading clock time and determining the # of Instructions executed). All processes *can manipulate time* (wait some interval of wall clock time and execute a set number of instructions then, block).**Kocher’s Attack** computes **x=azmod n** (length of runtime is related to number of 1 bits in z). **ISOLATION** **1. VIRTUAL MACHINES (**Emulate computer; Process cannot access underlying computer system, anything not part of that computer system) 2**. SANDBOXING** Does not emulate computer; Alters interface between computers, process.   
**VIRTUAL MACHINES**: A program\* that simulates hardware of computer system. Hypervisor (aka Virtual machine monitor, or VMMonitor) provides VM on which conventional OS can run. Hypervisor mediates all interactions of VM with resources, other VMS. Hypervisor may or may not know about processes running on each VM. Hypervisor is, essentially, a security kernel and Invoked on trap to execute privileged instruction **HV subjects:** users, VM.

**SANDBOX:** Environment in which actions of process are restricted according to security policy. 1. Can add extra security checking mechanisms to libraries, kernel 2. Can modify program or process to be executed. *Categories* **Jail (**network-access restrictions, and a restricted filesystem namespace) **Rule-based execution (**Users specify a set of determined rules that govern system behavior (such as starting process, access levels) **pastebin** (Allows users to execute pasted code snippets) **Secure Computing Mode (seccomp)** Sandbox built in the Linux kernel. When activated, only write(), read(), exit() and sigreturn() system calls are allowed.  
**COVERT CHANNEL**: A path of communication not designed to be used for communication e.g. file system. Communicates outside direct connection .*Covert storage channel* uses attribute of shared resource *Covert timing channel* uses temporal or ordering relationship among accesses to shared resource. *Premise*: covert channels require sharing Objective: identify manner of sharing. By: modeling flow of information through shared resources *Approach*: identify resources that are shared & construct flow tree for a shared resource & determine communication protocol, if a covert channel is present. The key properties of covert channels are *existence* and *bandwidth*. Existence tells us that there is a channel along which information can be transmitted. Bandwidth tells us how rapidly information can be sent. Covert channel analysis establishes both properties. Then the channels can be eliminated or their bandwidths can be reduced. **Detection of Covert Channels: 1. Noninterference 2. Shared resource matrix methodology 3. Info flow analysis 4. CFT:** Tree nodes: ***Goal symbols***Modification: attribute modified ; Recognition: attribute modification detected ; Direct recognition: subject can detect attribute modification by referencing attribute directly or calling function that returns it ;Inferred recognition: subject can detect attribute modification without direct reference; Inferred-via: info passed from one attribute to another via specified primitive (e.g. syscall) ;Recognized-new-state: modified attribute specified by inferred-via goal**. Other symbols** 1. Operation symbol represents primitive operation 2. Failure symbol indicates information cannot be sent along path 3. And symbol reached when for all children: Child is operation; and If child goal, then goal is reached 4. Or symbol reached when for any child: Child is operation; or If child goal, then goal is reached. **Construction rules:** **1)** topmost goal (modification and recognition)  
**2)**modification goal – what modifies attribute? (mod op1 or mod op2 … n) **3.** recognition goal – what recognizes changes in attribute (direct-recog or inferred-recog) 4.direct recognition goal - what returns information on attribute ( rec op1 or rec op2) **5.** inferred recognition – what returns info if attribute has been modified ((ref op and mod op) or (ref op and mod op) …) **6**.inferred-via goal – what indirectly signals change in attribute ( inferred op and recog-new-state) **7.**recognize-new-state – what is new state of attribute) recog or recog … **MITIGATION** Goal is to obscure amount of resources a process uses (receiver cannot determine what part sender is using and what part is obfuscated). *TO DO THIS*: devote uniform, fixed amount of resources to each process and inject randomness into allocation, use of resources. **MITIGATING COVERT CHANNELS** Covert channels are difficult to eliminate. Countermeasures focus on making the channel less useful by decreasing its capacity, usually through the addition of randomness to obscure the regularity that sending and receiving requires.

**TSU:** Confinement problem: prevent leakage of information Solution: separation and/or isolation. Shared resources offer paths along which information can be transferred Isolation mechanisms include virtual machines sandboxes. Covert channels difficult if not impossible to eliminate .But, can reduce bandwidth.

**Exploit**: it’s a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability in order to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic.

**Common Web Application Vulnerabilities: 1*.****Injection flaws* ***2****.Cross Site Scripting (XSS)* ***3.*** *Insecure Direct Object References* ***4.*** *Security misconfiguration* ***5.****Cross Site Request Forgery (CSRF)* ***6****.Using components with known vulnerabilities*. **1**. IJ: Occurs when untrusted data is sent to an interpreter as part of a command or query, that can trick the interpreter into executing unintended commands or accessing data without proper authorization e.g. SQL Injection, Command Injection **2. XSS:** An attacker gives your web application JavaScript tags on input. When this input is returned to the user, the user’s browser will execute it. E.g. *stored XSS* (attacker, web server, bad code, user’s cookie, user) and *reflected XSS (*most frequent: an attacker injects browser executable code within a single HTTP response. The injected attack is not stored within the application itself; it is non-persistent and only impacts users who open a maliciously crafted link or third-party web page. The attack string is included as part of the crafted URI or HTTP parameters, improperly processed by the application, and returned to the victim. **3. IDOR:** This is a classic case of trusting user input and paying the price in a resulting security vulnerability. A direct object reference means that an internal object such as a file or database key is exposed to the user **4.** **SM**: Web servers and applications that have been misconfigured are way more common than those that have been configured properly. **5. CSRF** An attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. (‘ or 1=1; /\* )  
**MALWARE** *The word malware (malicious software) describes any piece of code designed to infect your computer (or mobile device) and make it do things that you don't want it to do, such as mass-mail spam or steal your banking passwords. Trojans, worms, and rootkits are all types of malware***.** Availability (Deletion of files and directories/Renaming of files /Encryption of files, disks, system calls /Unauthorized calls to system software such as FORMAT, FDISK, etc.) Integrity (Corruption of system files and system areas (MBRs, FAT, etc.)/ Garbling data such as spreadsheet formulas /Corruption of both application and data files by unauthorized file writes) Confidentiality (Capture and forwarding of passwords/Forwarding of personal and confidential files to newsgroups and elsewhere) **TROJAN HORSE:** A program with an overt effect and a covert effect is a legitimate program with hidden illegitimate effects. Does NOT self-replicate.Free program made available to unsuspecting user /Actually contains code to do harm. NetBus Malicious logic placed in several small game programs. Allowed the attacker to control a machine remotely, and access the system acting as an administrator Zeus (Zbot) Spread through phishing schemes. Often used to steal banking information through key logging and form grabbing. **VIRUS** A virus is an entity that uses the resources of the host to spread and reproduce itself, usually without informed operator action. Strong viruses use normal computer operations to achieve the virus design goals. There is no single characteristic that can be used to identify a previously unknown virus program. Types **Boot sector infector** – virus that inserts itself into the boot sector of a disk **Executable infector** – virus that infects executable programs **Multipartite virus** – one that can infect either boot sectors or applications **Terminate and Stay Resident Virus (TSR)** – one that stays active in memory after the application has terminated. **Intendeds**: reproductive mechanism never triggers, or if triggered, code never attaches to host. ex. virus intended to execute on Sundays and uses DOS system call Get Date. Virus waits for Get Date to return “7” but Get Date only returns values between “0..6.”**Corruptions**: may be caused by system transfers, incomplete “cleansing” and poorly maintained virus collections. Antivirus programs often detect corrupted non-viral programs simply to avoid being penalized by incompetent testers and reviewers. M/w design consideration: infection, detection, habitation, trigger, payload, reproduction and removal.   
**HW 5: Footprinting** is the process of using various tools and technologies to understand and learn the best way to attack a target. "Footprinting" generally refers to one of the pre-attack phases; tasks performed prior to doing the actual attack. Attackers find out as much as possible without actually giving themselves away. They find public information or appear as normal users. The attacker/hacker does a ‘whois’ lookup to find as much information as possible about the network along with the domain name. **Hackers**: Helps successful hackers building their information database about a company’s security weaknesses. Information footprinting can help them learn these weaknesses in advance to take proper action .Therefore, security personnel need to incorporate footprinting as one of the preliminary security actions and strengthen their security stance. **Crackers:** Crackers might stroll through our DNS tables using nslookup, dig, or other utilities to do domain transfers to find the names of machine. Specifically, they look for domain names, network blocks, particular IP addresses, networking protocols in use, internal domain names, IDSs (Intrusion Detection Systems), telephone numbers, ACLs (Access Control Lists), etc. Crackers can get a lot of data to use against us like: Locations, Related companies, Merger or acquisition news ,Phone numbers ,Contact names and email. **Whois:** As Wiki defines it, is a query and response protocol that is widely used for querying databases that store the registered users or assignees of an Internet resource, such as a domain name, an IP address block, or an autonomous system, but is also used for a wider range of other information. The protocol stores and delivers database content in a human-readable format. **Nslookup** is a program to query Internet domain name servers. Nslookup has two modes: interactive and non-interactive. Interactive mode allows the user to query name servers for information about various hosts and domains or to print a list of hosts in a domain. We can run multiple commands one after the other (as shown for auburn.edu) **Interactive mode** is entered in the following cases**: 1**. when no arguments are given (the default name server will be used) **2.** when the first argument is a hyphen (-) and the second argument is the host name or Internet address of a name server. **Non-interactive mode** is used when the name or Internet address of the host to be looked up is given as the first argument. The optional second argument specifies the host name or address of a name server. **Tracer**( Trace Route): a powerful network troubleshooting utility. It is a route-tracing diagnostic utility that determines the path it takes for a packet to travel from our computer to a destination. tracert can print the entire route it takes for us to contact google.com or yahoo.com or any other destination. Network Mapper( **Nmap**) reveals open ports and services running on a remote system. **Blacklists ZapBL and DRONEBL** DRONEBL : DroneBL is a realtime monitor of abusable IPs, which has the goal of stopping abuse of infected machines. ZapBL: ZapBL is a DNS-based realtime blacklist. ZapBL does NOT block email. ZapBL is merely a list of opinions of the administrators as to where they do not want to receive email from. **Purpose**: Blacklist is a method of keep the Internet clean from unsolicited emails or spam’s. The ISP’s usually make use of a blacklist database which they make use to filter incoming email messages. This has been enabled so as to encourage and increase server security. E.g of a tool is mxtools.com **Grey-listing** is a simple way of blocking emails. in Grey listing, what it checks is that if the mail is legitimate or not. If there is a prior relationship between the sender and the receiver, the message get delivered without any issue. If the sender is suspicious and unknown the message gets temporarily rejected with 451 error message. *How this works*: As we discussed earlier, if no relation is established between the sender and the recipient, there are chances of getting a mail to be rejected with 451 bounce back error. A retry will be there since its a temporary rejection.A server with Greylisting enabled, mainly checks for the following information.1. Email address of sender2. IP address of host3. Recipient email address.This is commonly known as triplet. It is usually cached when a mail delivery gets rejected. If the mail is still getting rejected, the triplet get retires from its cache. **Whitelist:** If you are greylisted or blacklisted, you would need to get whitelisted so that you can start to use the mail services again. This process mainly deals with keep the spam filters allow certains email address or IP from being getting blacklisted or grey listed. If your mail server IP was reported for a spamming activity, possibly you would be blacklisted on ISP’s databases. If you are a spammer, your mail will either gets bounced back or end delivering in the spam or junk folder of an email address.  
**Domain Keys Identified Mail (DKIM)** lets an organization take responsibility for a message that is in transit. The organization is a handler of the message, either as its originator or as an intermediary. Their reputation is the basis for evaluating whether to trust the message for further handling, such as delivery. Technically DKIM provides a method for validating a domain name identity that is associated with a message through cryptographic authentication. DKIM attaches a new domain name identifier to a message and uses cryptographic techniques to validate authorization for its presence. The identifier is independent of any other identifier in the message, such in the author’s From: field. **SUM UP** as already said above DKIM alone doesn't grant in any way that the sender server is allowed to send outgoing mail for the specific domain  SPF is powerless with messages forged in shared hosting scenario as all the mail will appear as the same coming IP .DMARC is still in its early age and unfortunately not used as much as hoped to make a huge difference. DMARC can (and will) break your mail flow if you don't set up both SPF and DKIM before changing DMARC policy to anything above "none".  
 **DNS** cannot go down and if a DNS service goes down, network attached devices stop working. A company loses connectivity to the internet and hence cannot conduct business online. This leads to loss of revenue, customer defection and negative brand impact. Not only for notoriety, DNS attacks can gain access to political data and financial data of an organization. ( READ DMARK vs SPF vs DKIM from hw 5)  
**An intrusion detection system (IDS)** is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a management station. IDS come in a variety of “flavors” and approach the goal of detecting suspicious traffic in different ways. There are network based (NIDS) and host based (HIDS) intrusion detection systems. NIDS is a network security system focusing on the attacks that come from the inside of the network (authorized users). Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, and reporting attempts. In addition, organizations use **IDPSes** for other purposes, such as identifying problems with security policies, documenting existing threats and deterring individuals from violating security policies. **IDPSes** have become a necessary addition to the security infrastructure of nearly every organization.[1] IDPSes typically record information related to observed events, notify security administrators of important observed events and produce reports. Many IDPSes can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g. reconfiguring a firewall) or changing the attack's content.[1]