Some Technical Tips for success – Mentor's Point of View (Gary Palmer – Team-Wilson)

General (defensive Cyber):

1. I was not able to do this because I began mentoring after the team was formed, but a each team member should undergo a Cyber knowledge baseline assessment at the beginning of the team formation. It is always a good thing to know the experience level of all you team members as early as possible. It will allow you to focus your session in order to cover material most urgently needed for all members to be adept. See 4 below for possible topics to include for assessment.
2. Build a portfolio of current (no older than 1 year) Cyber events in the news to mention/discuss in practice with team (e.g., STUXNET). Try to make the connection by breaking down the method/technique of attack used and try to have the team formulate valid defenses. Especially good if the techniques map to specific areas you are focusing practice on but any Cyber training has positive benefit.
3. Spend some time discussing the historical nature of computing. Lots of times the team, being so young in most cases, will take a lot of the technology for granted. It's my opinion but I sincerely believe that doing this will provide perspective to the team and may even allow them to see the technology more clearly. *Of course, these practices are limited as well as time for perform them so it will be a delicate balance regards just how much time you spend discussion historical information. After all, it is a competition and not a university course.*
4. Make sure you have a curriculum with proposed timeline mapped out for each practice. This takes some work and oftentimes it will change due to some technical glitch but you need a plan and it will make the practice run much more smoothly. Crawl, walk, run, sprint…
    a. Suggested Map of topics in order:
        i. Computer hardware: Open a desktop box, take a look inside. Discuss motherboard/mainboard and components (CPU, clock, memory (RAM), firmware, BIOS, I/O (SATA), Cooling (why?), PCI bus, etc. Video/Display adapters, peripherals (DVD/CD, smartcard reader, sound card,etc.), printing, and more
        ii. Computer Operating Systems (OS): Purpose (what does an OS do and why is it needed?): It is the *Ambassador* between user and hardware, feeds instructions to the CPU. 32 vs 64 bit architectures, differences? How do application interact with the OS? (this is a serious attack vector topic but may require teaching at a later date due to the overall experience and knowledge level of the team). Generally explore the two major current OS offering, Windows & Unix/Linux. This may be a place where a little history may be in order (SAGE, UNIX, IBM-VM, RSX-11, VMS, Ultrix, Windows {3.1, 95, 98, NT, 2000, 2xK Srv, 7} Linux; and Trusted variants MULTIX, ULTRIX, XENIX,

SEVMS, Trusted Solaris, SELinux, etc) – A brief discussion of what a trusted OS is probably a good Idea as well because it will try to show the areas of the OS that researchers tried to "lock down" in order to secure known vulnerable areas in the general OS model.

iii. Basic Networking: Again historical context may help here but suffice to say these topics are to be discussed at a minimum: Differentiate between types of networks e.g., WAN (wide area net), LAN (local area net), PAN (personal area net), etc. Discuss IPV4 addressing scheme in terms of protocol. I like to use the postal service (**PS**) allegory. The United States **PS** is the main dispatcher controlling the guaranteed transmission of information (physical packages) on a huge network (the United States of America) it has access to. The **PS** requires a letter to be addressed a certain way for guaranteed mail delivery. The envelope must be addressed just so for machines and people to understand where the package came from (*source*) and where it is going (*destination*). The set of rules about how to address a letter to make sure it gets where it is going is called a *protocol*. We are all very familiar with this *protocol* (at least most of us who still mail physical letters are still familiar with it). The same holds true for sending information on a computer network. It this case the dispatcher is called the *network stack* (**NS** is a combination of firmware on the network adapter and software in the host **OS**) and the package is called a *datagram.* The *datagram* has a header attached with a very precise format. Part of that format is a dedicated area for source and destination address (there are other dedicated areas too but that can be covered later – projected or hand out graphics will help here). The **PS** expects to see name, number, street, zip, etc in precise places on the package. The **NS** expects the same from the address it will read from the datagram header for each packet of information it views (a *datagram* may be made up of several packets – in **PS** speak its like breaking a shipment into several pieces e.g., 1of 3, 2 of 3, 3 of 3 = 1 shipment.

iv. System Administration: This set of practices should cover; programming languages, Compliers, Interpreters, Bitcode, shell scripts, power shell, and other methods used for System administrators and users to interact on-line with a particular OS and even the network. It should cover: File system types (NTFS, ext2, HFS, RMS, etc.) directory structures, folders, and files. How an operating system enforces access via its privilege structures (Read, write, exec, delete), groups, sticky bits (Linux), and how that structure is organized (different in Win and Linux). Here is where one might want to start introducing the "checklist" concept into the practices. Team Wilson (**TW**)

built a pretty extensive checklist that was used as a baseline for almost all practice sessions. It basically directed members to a prescribed set of actions that would, for the most part, serve to assist the defender in 1. Assessing the state of his machine/computer and 2. Taking steps to make sure that any security problem found were remedied using the most secure method available and in the best order. The **TW** checklist also categorizes information by type of security concern (Accounts, patching, authorizations, ports, services, monitoring, etc…), which will help in ordering practice topics as well. Instead of just presenting the checklist use it as a template. Let the team create its own as a team. Perhaps just use the major categorizations and them allow the tem to derive details. Make sure all members see the logic and agree to the final product. Personalize the result let them make it there own, it will be much more effective that way.

5. Use as many examples as possible during practice sessions and also try to shy away from reading slides to the team, rather try and make the sessions interactive. Lecture a bit then break the team into groups and issue sample challenges for the groups to work on and then report findings. This will keep the team engaged, it will allow them to work together and will also allow you to observe the state of that teamwork to help you make it more effective. Intervene when "Alpha" members monopolize the challenge and try to get all to interact toward solutions, etc.
6. Take frequent short breaks. You have to "play this by ear" there may be times when the majority or all of the team is very engaged in the material but for the most part their attention is limited and should not be allowed to stray into distraction or boredom.

Competition Specific:

1. This is a competition and that competition has specific challenges associated with it. As such all of these teams are formed in the hopes of **winning** that completion.
2. So, practice sessions will be focused on teaching material that will enhance team members overall computer knowledge and security skills but will also, at some point, shift focus to a particular challenge issued by the competition organizers (**CO**).
3. Those points in the process must be assessed by team staff (coach, assistant, mentor, parents and team members) when they approach.
4. Generally, the **CO** will present a particular challenge to be addressed and that challenge will be described to some level of detail. The level of that detail will help determine if and how much deviation from your current practice curriculum will have to be made. For example, in CP III elimination challenges were issued with little detail. The teams were told that they would have to assess and remediate one or more "bugged" virtual machines. They

were told what specific OS was involved but nothing more regarding potential vulnerabilities, etc. In that case, it was only a matter of making sure that the as much of the team as possible was familiar with the OS(s) in question and it would be smart to simply stay on course using the developing team checklist and modifying it where necessary to include the OS present in that challenge. Typically, the announcement of these elimination rounds was made about 1-1.5 weeks before the event. Depending on practice schedule this seemed like enough time to adjust sessions accordingly.

5. In the case of the major competition events (semi-final and final rounds) much more detail was provided and usually about 2 weeks before the actual event was planned. The information included an Operations Plan (OPlan) that laid out the network topology, OSs, services required, applications, databases and defensive tools made available to the teams during the event. When these events occurred the practice schedule became much more focused on the specifics of the event. The following questions were asked to help formulate an approach:
    a. How is the network laid out? How many domains? Where are my monitoring points? What if any relationships/dependencies are there between computers on this network?
    b. Do I have Intrusions Detection/Prevention available? Where are the best points in the topology to monitor this network?
    c. What applications/databases/utilities are involved? Have we ever discussed these in sessions? What are specific vulnerable points for these apps?
    d. Who on the team has what strengths or weaknesses given the challenge detail? Who needs to cover what part of the challenge? How will the team work together to secure this network?

    With answers to as many of these questions as possible the team can continue by focusing first on items never before discussed. The mentor/coach will need to do some preliminary work researching "foreign" items. Once preliminary detail is known each particular app/utility involved in the network should be assigned to a particular team member for research (internet or otherwise). What is the main function of this tool? How is it administered? What accounts are needed? Must it run as *root/administrator*? Any known security problems/vulnerabilities? Have the member report on findings at the next practice (2-3 days minimum assignment lead – make sure parents agree with assignment – team member may have heavy homework load or other priorities – If it can't be assigned the mentor may have to research and present information themselves – time allowing)
6. By the time of the event the team should have a plan of attack/defense including:
    a. Who is responsible for what systems/services?
    b. Who is the lead (captain – tie breaker) for decisions?
    c. What are the team rules for discussion before making a change to any system – outside what the checklist prescribes for a problem?

d. Make sure the whole team knows that its good to talk and bad to hold information to yourself.
e. If you have questions ask the team.
f. Be patient there may be a few problems happening at once.
g. If you get stuck, it's ok. Refer to the checklist; try to reason it out as best you can. If you can't figure out the issue tell the team. Don't make knee-jerk reactions without talking it over with another team member.
h. Document all changes you make to all systems. Include, at a minimum: Time of change, files and directory location involved, path or change (if GUI used- e.g., System>control panel> admin tools> users, etc), reason change made, initials of member making the change.