ResearchSOC

Example Security Exercises

1 It's Gone

Pick a system, any system. Think of a reason why it's completely hosed–failure of the entire RAID array, fire in the datacenter, evil script kiddies, sysadmin mistake–and see how your team copes. Some questions to ask when all is done:

- If you don't have another of these systems to fail over to, where are your users going while the system is down? What stopped working and for how long?
- If you have a failover system, how long did it take to fail over? What did your users experience in the mean time?
- How hard was it to replace the system? Were backups adequate? Did the available personnel know what to do and have the authority to do it?
- What data was lost? Are backups being made often enough?
- Were any other systems impacted by this system's death? For example, if your LDAP server died suddenly, did administrators still have access to other systems? Did anything fail open?

2 Naughty Ned

Choose a team member with elevated privileges (any member of your security or systems administration / ops team is usually a good choice, so might be a leadership team member or a developer). Pretend he or she has been fired, and revoke all of his or her privileges. Now he or she gets to cause whatever chaos he or she can with any privileges that remain. This is a great way to test your offboarding checklist.

3 Wolf in Sheep's Clothing

Most of the Red Team plays the part of ordinary users here. One plays a malicious user. Can the blue team terminate the malicious user's activity without negatively impacting any of the nice users?

4 Committer Should Be Committed

This is a great one for software development teams. A developer, working while sleep deprived (thank you red team), has committed something to the master branch of the repo that he or she

shouldn't have. It might have been login credentials for an internal system, or naked pictures of the boss's dog, the content doesn't matter. The important thing is that it has to go.

See how your team removes the offending data both from the working tree AND the repository history, without breaking everyone's workflow beyond recognition.

5 Operation!

If you run a devops environment, this one is for you. It's far too easy for deployment workflows to end up with very low bus factors (the number of people who must be hit by a bus before the project is doomed or at least in serious trouble). Watch a deployment or two and figure out who the 1-3 most critical people are in that path, then declare them unreachable for the purpose of the exercise.

Now, suppose that a critical security vulnerability has been found in your deployed product. Challenge your team to make a trivial code change (e.g. add a comment saying "We did it!" to the code at a specified point) then run your entire test suite and deploy the code with those critical people gone.

6 Finger in the Dam

Find a (hopefully fairly harmless) proof-of-concept for the most recent security vulnerability you applied patches for. Run it against everything and find out whether or not the hole was really plugged.

7 Negative Nancy

Have a red team member contact your primary customer support avenue, playing the part of a user who is absolutely certain that their private information entrusted to your service has been compromised. Bonus points if the character is a "difficult" personality. See how the team handles it.

8 Fell Off a Truck

Your primary authentication database has fallen off a truck (your choice whether this is your database of external user accounts or something for internal personnel only). Demonstrate how you would notify those affected and force password resets. Bonus points if you can detect and flag attempts to use compromised credentials.

9 Ewe Did It

Start an (otherwise innocuous) process on one of your system that occupies as much RAM as it can get its hands on. See how long it takes for anyone to notice, and how they respond.

10 Stowaway

Connect an unauthorized network device into your network and let it talk to something. See how your team tracks it down and removes it.

11 Exfiltration

One of your employees has decided that they would like your big, valuable, internal database. The Red Team tries to exfiltrate the target (any way they like) without being detected.

12 Nosy Nelly

One of your systems starts nmapping the network. Does anyone notice?

13 Blame the Mailman

A system that should never send mail starts sending (or trying to send) spam. What happens next?

14 Delivery

Someone not known to your staff tries to talk their way into some limited-access area of the building, such as your datacenter. It helps to appear pregnant, talk on your phone, tailgate someone, carry something heavy, or insist you are making a delivery or have an appointment. See if anyone stops you.

15 Pick-up Stix

Drop some USB sticks around the building: in the parking lot, the restroom, a conference room, a lobby. Place an autorun executable on the sticks that notifies you when they are inserted in a machine that autoruns USB devices, and place an interesting-looking file on there that also tries to call home when opened.

16 Phishing Expedition

Send a convincing phishing email (with at least one flaw that a reasonable person would pick up on) to your staff, directing them to a fake login page and see who gives up their credentials. Note: this one is likely to rankle some people who feel duped when you come out and tell them what happened, but is really good at driving home the importance of phishing awareness if you can afford the political fallout.

17 Compromising Positions

Suppose that a rootkit has been discovered on a critical piece of infrastructure on your internal network (e.g. your Satellite server or your LDAP server). Challenge your team to prove that none of your *other* systems have been compromised. Not assume: prove.

18 Failure is Always an Option

Single points of failure are frequently discussed in meetings. Pay attention, and document these, then break one. Does the scope of the outage match expectation? Does the recovery time/process match expectations?

19 Free For All

This is a big, high-investment exercise to run, but it's also the best:

Set up a dedicated environment for your exercise to run in that is not connected to your other internal networks or to the public internet. Provide a set of services that need to be kept running, and consider adding some data meant to be kept confidential. Don't set up that environment in the most secure way possible.

Set targets for the Red and Blue teams with various point values, e.g. 10 points to each team for each system they control at the end of the exercise, 20 points for the blue team for every half-hour that a particular service continues without interruption, 50 points for the red team if they find and decrypt such-and-such a file. Then set both teams loose with nothing but a time limit and see what happens.
