3.4. Deployer

The *deployer* role refers to the individual or organization responsible for the fielded systems that use or otherwise depend on products with vulnerabilities.

Deployers include the following:

- network and cloud infrastructure providers
- <anything>-as-a-service providers
- outsourced IT operations
- in-house IT operations
- individual users

Deployers typically must take some action in response to a vulnerability in a product they've deployed. Most often this means deploying a patch, but it can also involve the application of security controls, such as reconfiguring defensive systems, adding monitoring or detection rules, or applying mitigations.

Automation of the deployment process increases the efficiency of the deployer's response at the same time it decreases the duration of the risk posed by vulnerable systems.
Although the deployer role is primarily concerned with Vulnerability Management practices that sit downstream of CVD, it's worth spending a few moments to understand how it fits in with CVD.

Deployer Vulnerability Response Process

A deployer's vulnerability response process usually involves the following sequence of stages:

- Become aware of vulnerability, mitigation, and/or fix
- Prioritize the mitigation or fix into existing workload (triage)
- Test the mitigation or fix
- Confirm that the fix addresses the problem
- Avoid undesirable side effects
- Identify affected systems and plan the deployment:
    - staged or all-at-once
    - automated or manual
    - scheduled update window or out-of-band
- Deploy the mitigation or fix to affected systems

We cover each of these in more detail below.

Become Aware

In order to take action, a deployer must know about the vulnerability and have sufficient information to act on. Most often this information originates from the product vendor. However, since not all vulnerability reports are coordinated with the vendor for disclosure, vulnerability information can arrive from other sources as well.

Deployers should be on the lookout for and pay attention to:

- Vendor security notices
- Vendor customer support notices (not all vendors provide separate security notices, nor are all vulnerabilities always explicitly called out in update notes)
- Vulnerability and threat intelligence services
- Security discussions online including social media
- Mass media coverage of vulnerabilities

Prioritize Response

Deployers have many responsibilities beyond deploying patches. As a result, they need to prioritize their work and integrate patch deployment into their normal operations cycle. That might mean testing, scheduling out-of-band fixes, or planning for scheduled maintenance windows. Just as vendors need to triage reports in order to prioritize patch development appropriately, deployers must decide which patches and mitigations to deploy and when to deploy them. The deployer's workload often makes it difficult to patch all the things as quickly as they would like.

Test the Solution

Testing prior to deployment is important if either of the following conditions is true:

- The system's availability and performance are critical
- Reverting a patch deployment gone bad is difficult

In environments with efficient automated deployment and rollback capabilities, it may not be as necessary to test as heavily. But that's often an ideal scenario that few deployers find themselves in. Staged deployments or rollouts can be a significant help here—where some portion of the affected systems are updated to confirm the fix prior to wider rollout—allowing deployers to balance patch deployment with the risk of negative side effects.

Plan the Deployment

Deployers have many options when it comes to planning to deploy a patch or mitigation. Highly automated environments can dramatically shorten the time required to complete these stages, but the functions described here will usually still occur regardless of the deployer's automated patching capability.

Planning for a patch deployment requires two major steps:

1. Identify and enumerate system instances affected by the vulnerability. Vulnerability management tools can be used to scan for affected systems and prioritize patch deployment. Information about affected hosts helps to define the scale of the patching effort required.
2. Set the deployment schedule. If there are relatively few systems under management and vulnerabilities are fairly rare, a firstinfirst-out process might suffice. Larger enterprises often have scheduled maintenance windows during which they can deploy most patches. Alternatively, an organization might choose to push out a patch outside of a scheduled maintenance window, especially in cases where a vulnerability is being actively exploited or significant harm is expected should the vulnerability remain unpatched until the next maintenance window. Essentially the question boils down to deploy now or defer to later?

Execute the Plan

Obviously, it is important to actually carry out the deployment of the mitigation or fix. Automated patch deployment tools can make this process quite efficient. Regardless of the degree of automation of patch deployment, recurring or continuous monitoring for vulnerabilities can help measure the success of the deployment effort.

< 3.3. Vendor | 3.5. Coordinator >