RSAConference2019
San Francisco | March 4–8 | Moscone Center

SESSION ID: PDAC-F03

Tales from the PSIRT: 10 Years of Bugs, Vulnerabilities, and CVEs

Jon Green
VP & Chief Security Technologist
Aruba, a Hewlett Packard Enterprise company
@jgreen1024

Agenda

- Our clueless beginnings
- What's a PSIRT, exactly?
- Mismatched expectations
- Historical and not-so-historical examples
- Current challenges and future plans
- Q&A

Clueless Beginnings

- Mobility Controller Management Interface Buffer Overflow – 02/13/2007
- Aruba Mobility Controller Guest User Privilege Escalation – 02/13/2007
- VPN ISAKMP Message Processing Denial of Service – 11/13/2005
- SSH Tunneling (Port Forwarding) Through the Aruba Devices is Allowed – 06/14/2005
- IPsec Configurations May Be Vulnerable to Information Disclosure – 05/10/2005
- Risk of Multiple Denial of Service Attacks Using Modified ICMP Packets – 04/19/2005
- Aruba Switches are Vulnerable to a PPTP Exploit – 02/10/2005
- ISC DHCP Contains C Includes That Define "vsnprintf" to "vsprintf" Creating Potential Buffer Overflow Conditions – 06/15/2004
- ISC DHCPD Contains a Stack Buffer Overflow Vulnerability in Handling Log Lines Containing ASCII Characters Only – 06/14/2004
- IEEE 802.11 Wireless Network Protocol DSSS CCA Algorithm Vulnerable to Denial of service – 04/17/2004
- SSH Vulnerabilities – 11/18/2003

“Just trying to help…”

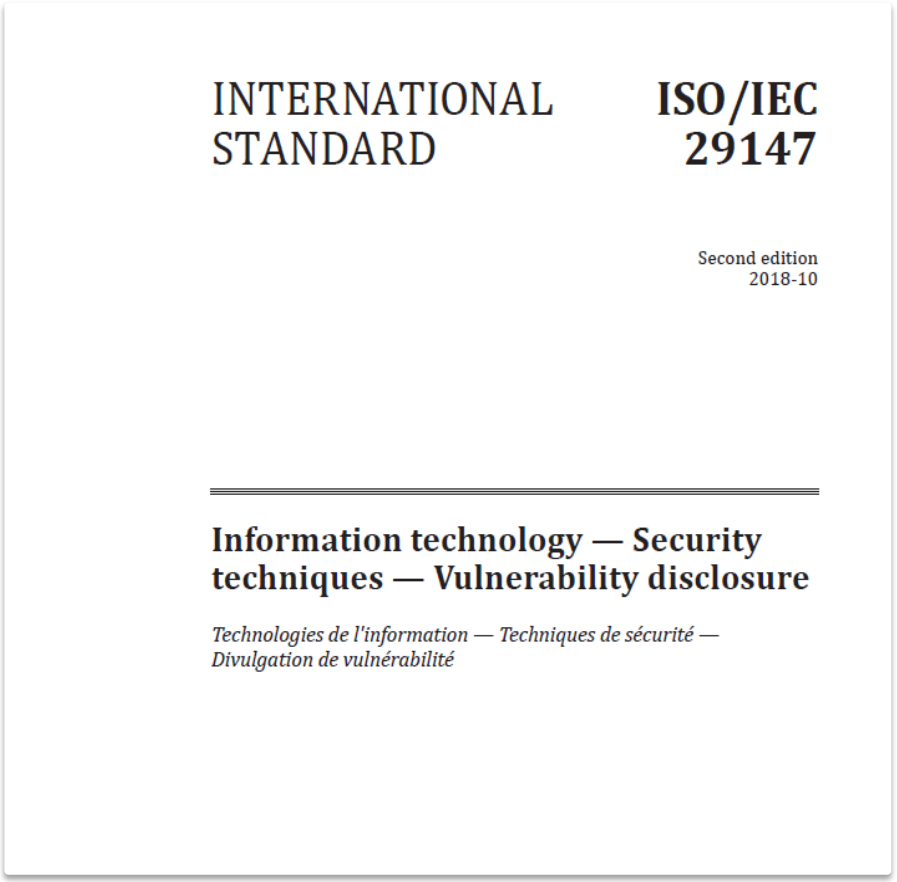
Forgotten developer test, or backdoor?

```
@@ -43,7 +43,6 @@
 #define SYCONF_SUCCESS_COOKIE   "SuccessCookie="

 #define AUTHSERVER_IP          "127.0.0.1"
-#define CP_GUEST_USER          "g73st7s3r"                                    /* name of guest user */
 #define CP_LOGIN_SCRIPT_PAGE          "/auth/login.html"        /* Java script with delay mechanism */
 #define CP_LOGIN_PAGE                  "/auth/index.html"
 #define CP_DEF_WELCOME                  "/auth/welcome.html"        /* page after authentication */
```

```
==== //depot/margot/sw/apache-cp/login-cgi/aruba-login.c#22 (text) ====

@@ -742,10 +742,6 @@
          * page forcebily
          */
     if (username && strlen(username) && password && strlen(password) && allow_user) {
-                if (!strcasecmp(username, CP_GUEST_USER) && !allow_guest) {
-                        MYSTRCOPY(reason, "Access denied");
-                        return -1;
-                }
                 if (show_fqdn && fqdn && strlen(fqdn)) {
                         snprintf(fqdn_name, sizeof(fqdn_name), "%s@%s",
                                 username, fqdn);
```

What is a PSIRT?

- *Product* Security Incident Response Team
- There's an ISO standard for that…
- Three primary tasks
  - Accepting incoming vulnerability reports
  - Communicating with vulnerability reporters
  - Publishing vulnerability advisories
- Other possible tasks:
  - Replicating vulnerabilities
  - Managing bug bounty programs
  - Ensuring vulnerabilities don't languish


INTERNATIONAL STANDARD

ISO/IEC 29147

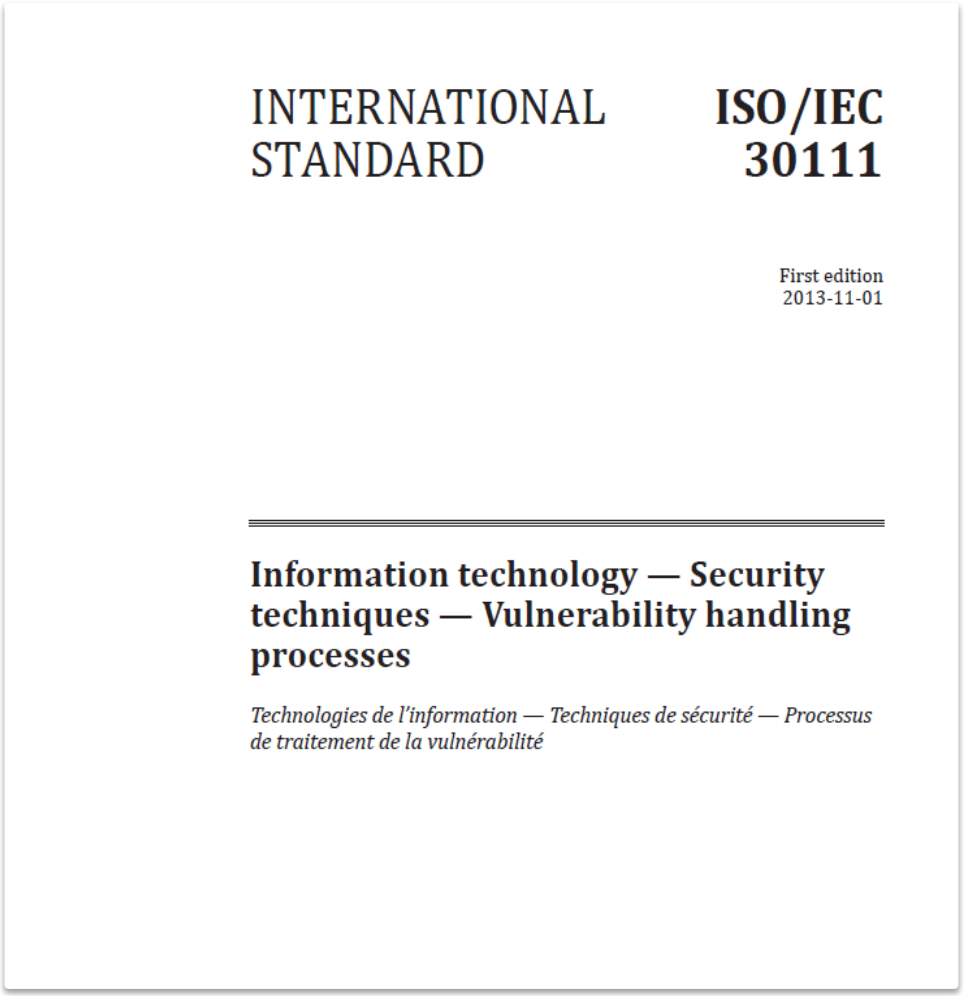
Second edition
2018-10

Information technology — Security techniques — Vulnerability disclosure

Technologies de l'information — Techniques de sécurité — Divulgation de vulnérabilité

The other half of vulnerability management

- What happens after the PSIRT takes a report?
- Developers typically respond with:
  (choose 3)
  - Silence…
  - “What does this mean?”
  - “I don’t think this is possible in the real world”
  - “OK we’ll take this in the next release (4 months away)”
  - “This bug is in CentOS. We’ll have to wait for Red Hat to fix it.”
  - “Oh crap we have to fix this right away!”


INTERNATIONAL STANDARD

ISO/IEC 30111

First edition
2013-11-01

Information technology — Security techniques — Vulnerability handling processes

Technologies de l'information — Techniques de sécurité — Processus de traitement de la vulnérabilité

A slightly different approach…

aruba
a Hewlett Packard Enterprise company

THREAT LABS

- Combines PSIRT with security research team
- Overcomes challenge of PSIRT incident responders without deep technical knowledge
- Overall mission: Drive security learnings back into product organization. Prevent vulnerabilities.

Mismatched Expectations

1. Selective disclosure
2. “Are any of your products vulnerable to CVE-2018-12345?”
3. “Here’s my vuln scan results… what does this mean?”
4. Outdated/vulnerable open-source software
5. The wannabe sysadmin
6. Banks and telcos…

RSAConference2019

Other Historical Examples

Aruba Controller Authentication Bypass (CVE-2014-7299)

```
Issue was seen in releases 6.4.2.1 and 6.3.1.11 builds

Issue : Give any random username and Password of more than 32 characters to login using SSH.
            will be able to login into the box with random username with role being the "usernanme"

steps to reproduce :
  $ ssh fdrsdr@10.4.156.11
    pswd : abcdefghijklmnopqrstuvwxyz123456789
```

```
scenarios tried

1. tried logging in with username same as the different mgmt-role we have

ssh read-only@10.4.156.11
ssh local-api-mgmt@10.4.156.11
ssh network-operations@10.4.156.11
ssh guest-provisioning@10.4.156.11

(7210-7019) #show loginsessions

Session Table
-------------
ID  User Name       User Role       Connection From  Idle Time  Session Time
--  ---------       ---------       ---------------  ---------  ------------
1   read-only       read-only       10.100.69.57     00:00:55   00:01:05
2   fdrsdr          fdrsdr          10.100.69.57     00:00:06   00:16:40
3   admin           root            10.100.69.57     00:00:00   00:27:28
4   local-api-mgmt  local-api-mgmt  10.100.69.57     00:00:15   00:00:25
```

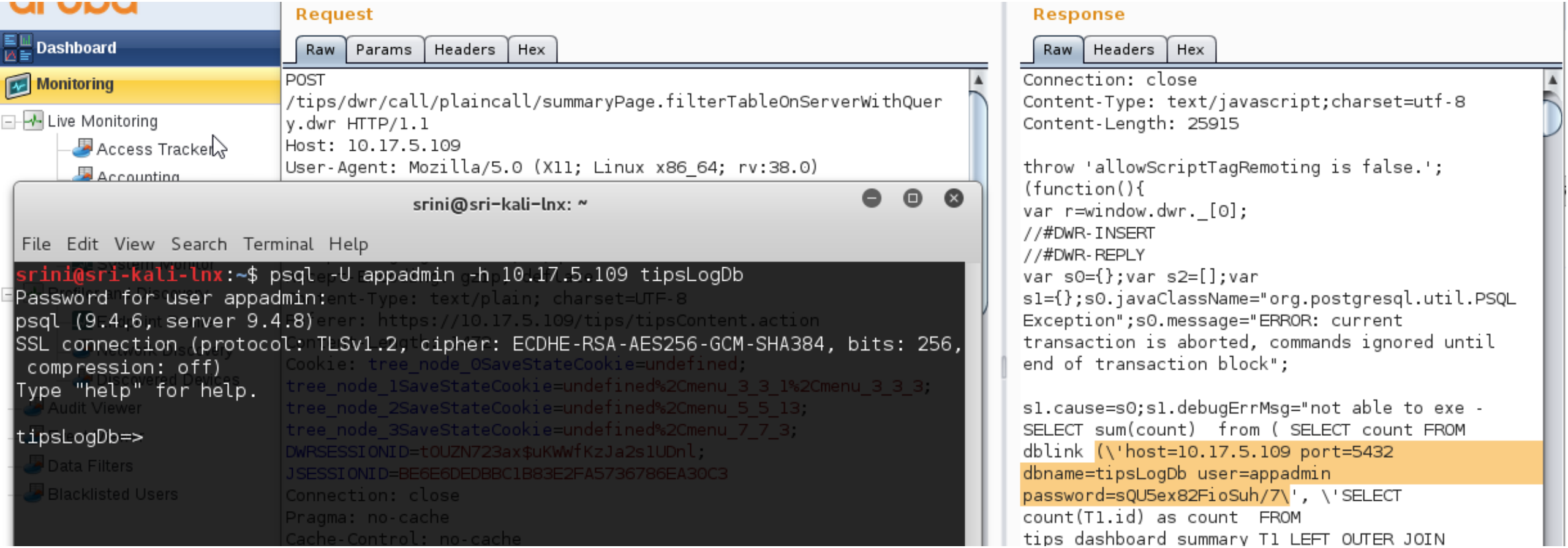
```
Root cause:
In aaaAuthenticateSSHUser(), AAA_ERROR_SENDING_DATA is returned when the password length is longer than 32.
However, the calling function, auth_password(), treats this return-code as 1 (which means successfully authenticated).

Regression:
Yes. It was caused by fix of ~~bug#105369~~.
```

```
==== //depot/margot/FCS6.4.X.0/openssh-5.8p1-X509/arubaSSHAuth.c#3 (text) ====

@@ -74,7 +74,7 @@
        if(!user || !passwd)
                return 0;
        if(strlen(user) > AAA_MGMT_USER_LEN  || strlen(passwd) > AAA_MGMT_PASSWD_LEN)
-               return AAA_ERROR_SENDING_DATA;
+               return 0;
```

aruba
a Hewlett Packard Enterprise company

Database Credential Compromise (CVE-2016-4401)

- Error messages sometimes reveal WAY too much.



A Small Vulnerability Report from Google…

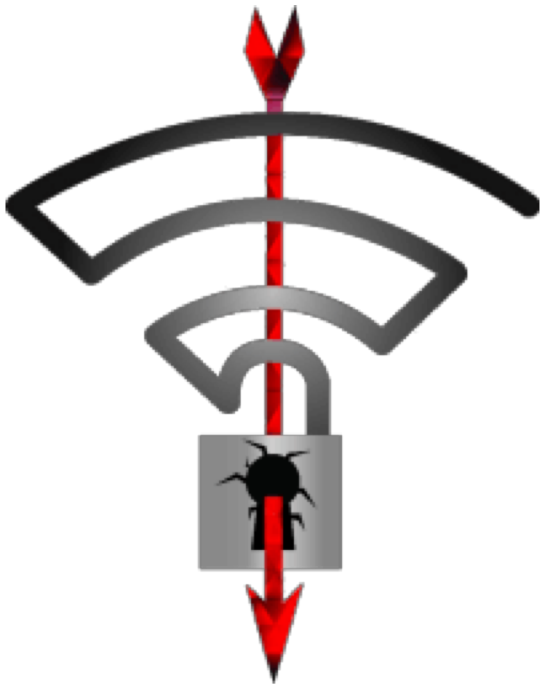
Vulnerabilities were identified in several Aruba Access Point models and Mobility Controllers used by Google. The Vulnerabilities were discovered during a black box security assessment and therefore the

This bug is subject to a 90 day disclosure deadline. If 90 days elapse without a broadly available patch, then the bug report will automatically become visible to the public. Please do let us know if you anticipate

Issue Details

- Aruba-01 - AMP: RabbitMQ Management interface exposed
- Aruba-02 - AMP: Perl eval{} used with XSRF token
- Aruba-03 - AMP: XSRF token uses weak calculation algorithm
- Aruba-04 - AMP: Arbitrary modification of /etc/ntp.conf
- Aruba-05 - AMP: PAPI uses static key for calculating validation checksum (auth bypass)
- Aruba-06 - AP: Insecure transmission of login credentials (GET)
- Aruba-07 - AP: Built in privileged "support" account
- Aruba-08 - AP: Static password hash for support account
- Aruba-09 - AP: Unusual account identified ("arubasecretadmin")
- Aruba-10 - AP: Privileged remote code execution
- Aruba-11 - AP: Radius passwords allow arbitrary raddb commands
- Aruba-12 - AP: Unauthenticated disclosure of environment variables
- Aruba-13 - AP: Information disclosure by firmware checking functionality
- Aruba-14 - AP: Unauthenticated automated firmware update requests
- Aruba-15 - AP: Firmware updater does not check certificates
- Aruba-16 - AP: Forceful downgrade of FW versions possible
- Aruba-17 - AP: Firmware update check discloses machine certificate
- Aruba-18 - AP: Firmware is downloaded via unencrypted connection
- Aruba-19 - AP: Firmware update Challenge/Response does not protect the Client
- Aruba-20 - AP: Unencrypted private keys and certs
- Aruba-21 - AP: Potential signature private key
- Aruba-22 - AP: PAPI Endpoints exposed to all interfaces
- Aruba-23 - AP: PAPI Endpoint does not validate MD5 signatures
- Aruba-24 - AP: PAPI protocol encrypted with weak encryption algorithm
- Aruba-25 - AP: PAPI protocol authentication bypass
- Aruba-26 - AP: Broadcast with detailed system information (LLDP)
- Aruba-27 - AP: User passwords are encrypted with a static key
- Aruba-28 - AP: The SSL library in use does not support TPM

KRACK (CVE-2017-13077 through 13088)



- Nightmare scenario for a Wi-Fi company: a flaw in the WPA2 *standard*
- KRACK was not that, but…
- Hundreds of vendors, millions of devices affected
- Coordinated disclosure required
- ICASI acted as coordinating body, with US-CERT
- ICASI prepared with NDAs for non-members
- Our main exposure: wpa_supplicant (open source)

Dealing with open source in supply chain

- Open-source software is widely used in enterprise products
- Vendors don't always understand which pieces they incorporate

  (did they type 'make' or 'apt-get'?)
- Ask for a manifest

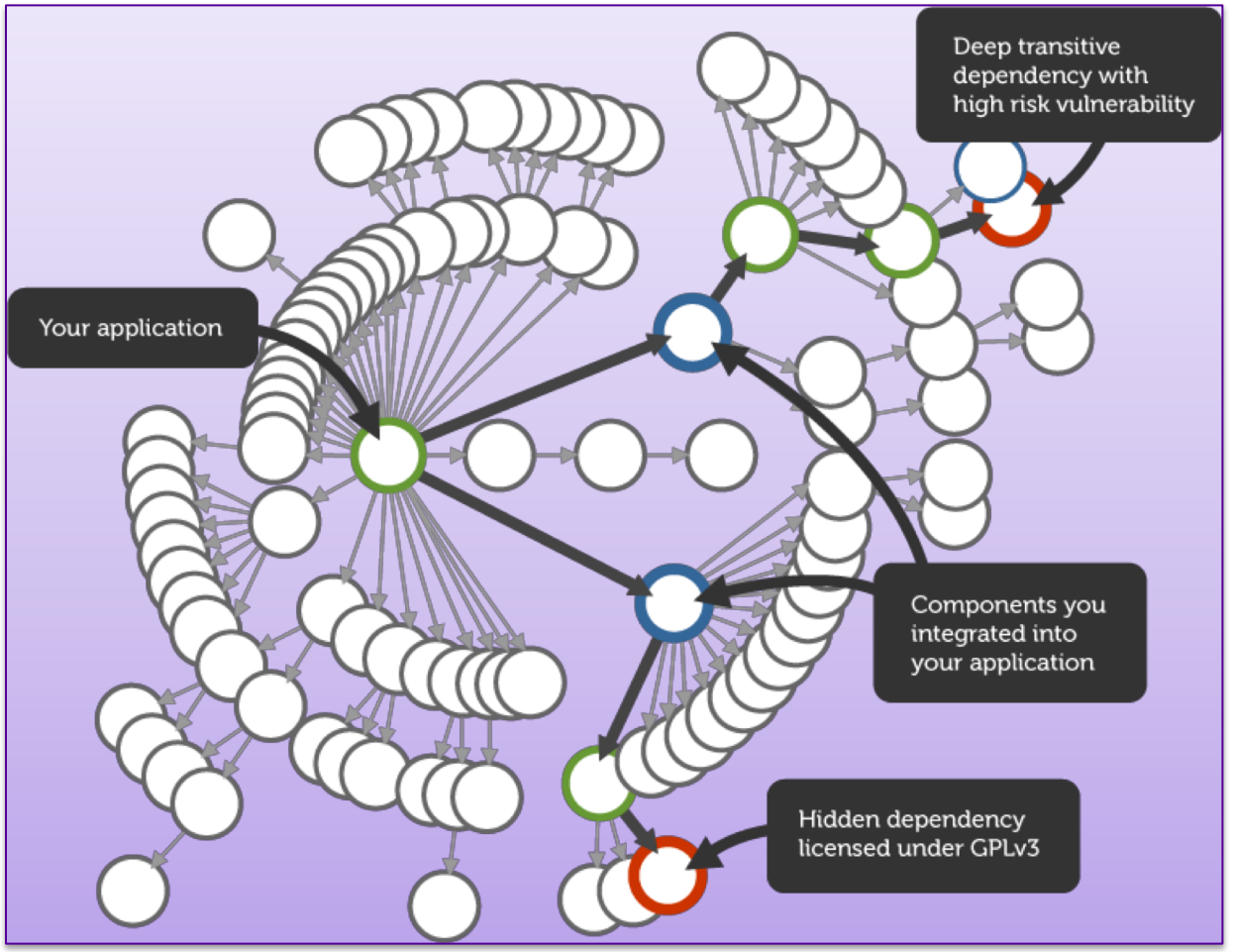


Image credit: Terry Bernstein/Sonatype
https://blog.sonatype.com/2011/10/evaluate-open-source-components-before-use-open-source-development-tip-5/

Current Challenges, Future Directions for Aruba

- Scaling
- Automating open-source vulnerability disclosure
- APIs
- Taking bug bounty program public
- Cloud, cloud, and more cloud
  - Disclosure
  - Bug bounty
- Knowing when/how to call for help

Summary

- Every IT product *will* have vulnerabilities
- Vendors are at different maturity levels. Help them understand your needs and expectations.
- If you have the skills and time – join a bug bounty program
- Resist the urge to email the PSIRT every time a new public vulnerability emerges.

Apply this information

- Survey your critical systems. Does every vendor have a PSIRT?
- Are you subscribed to one of their disclosure channels?
- Does the vendor have a reasonable and responsible disclosure policy? Does it align with your requirements?
- Review past advisories. Learn about how the product works, its components, and how vulnerabilities apply.

RSAConference2019

Thank you. AMA.

Jon Green
jon.green <at> hpe.com