All.Net Analyst Report and Newsletter

Welcome to our Analyst Report and Newsletter

Cybersecurity From Scratch – Part 10: Testing

Every once in a while we get to create a cyber-security program from scratch. …

Incident

At 0435 Sunday morning, I was sent an email from another client indicating an outage in a Management Analytics system that also supports the System of Records for Company. I am not a fire fighter, so I don't wake up at any hour on an emergency basis to fix things. That is reflected in the decisions about response times and service level agreements I have with clients. So when I got to my desk at about 0545, I went through over night emails, and noticed this one. I did a quick test and verified the system was down.

I recently spend 12 hours developing and recording a new class on Ransomware and Cyber-extortion, and as a result it was on my mind, especially since I am releasing these articles and just released videos yesterday covering these articles in a light way to get readers to read the details. Naturally, my own systems follow the same regimens I tell others to follow according to the standards of practice. But reality happens.

I was able to login to the non-functional (from an end user perspective) system, and found it had run out of disk space (again – see my latest article on the AWS disk space hole and related problems). This happens to systems, and so I did a reboot to recover the 25% of disk space AWS was using for no identifiable reason, and the system came back up and was operating, except of course with the same problem it had before. User services appeared to be substantially affected.

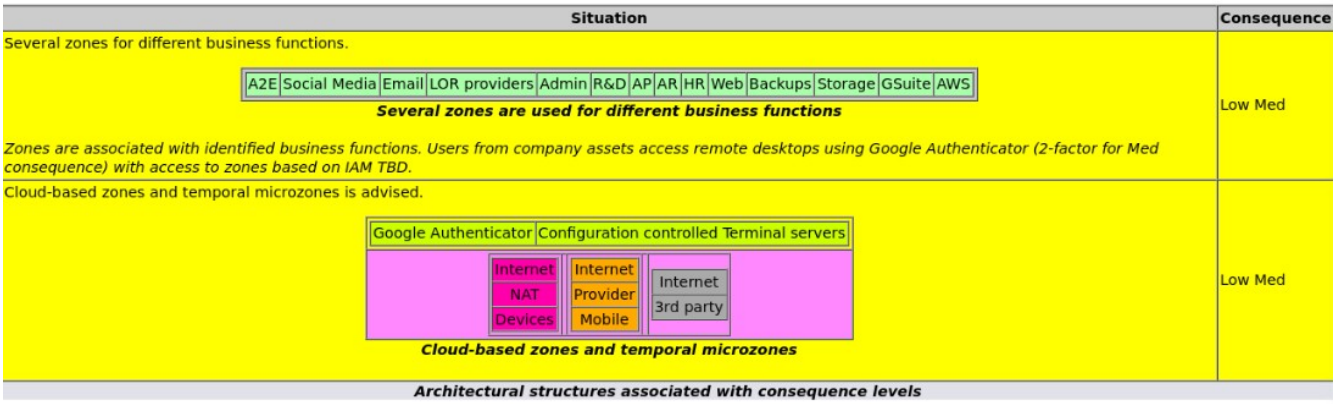
Next, I identified the potential sources of such a failure and focused on the storage area for passwords, machine states, etc. Lo and behold, the disk space failure happened in the middle of a 2-step critical code section, and thus the failure that should almost never happen happened. Still, I have backups, so I restored from the most handy backup (the one from 8 hours earlier), and the system worked again. A quick check on user activity indicated that no records showed anything that should have been adversely effected in the intervening time frame, and we were off to the races. I didn't need to go to the slightly less handy every 15 minute change tracking backup in the controlled access room… Total time to repair: <5 min.

But of course that is not the end of the incident. The fact that a critical section failed due to an out of disk condition had to be addressed. So I added code to do a backup of every change to the critical section files after the end of each critical section, each copy time and date stamped. While this will take additional disk space (~5K each time), it also means we will always be able to restore to the last good copy immediately, and never have to check for intervening changes.

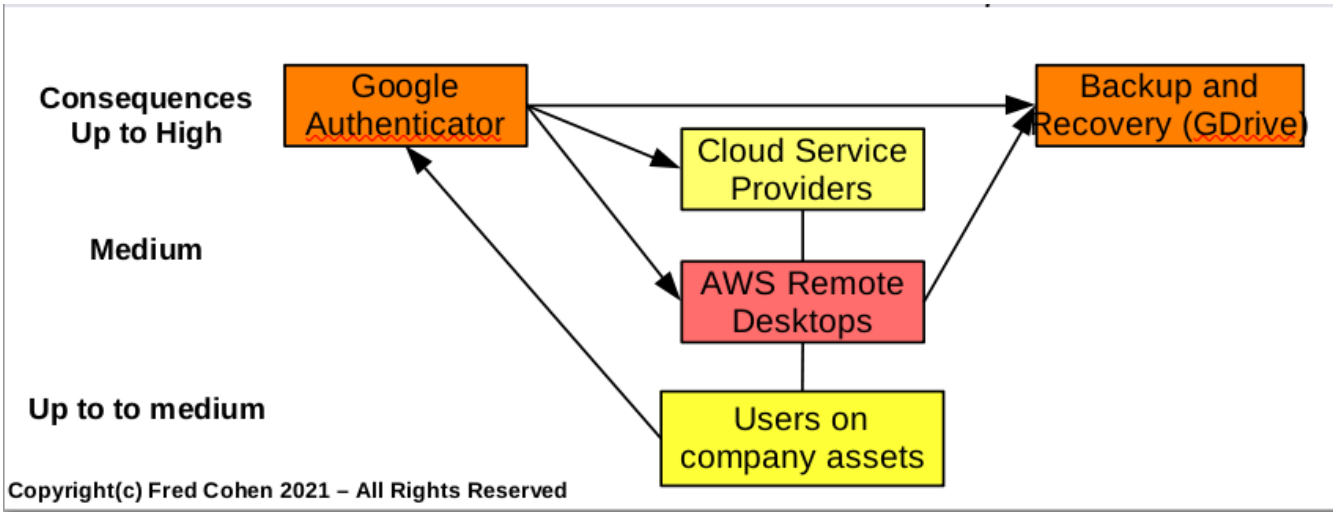
The longer term fix involves (1) getting AWS to fix their disk memory hole problem – not likely, and (2) adding more disk space. But this is slightly more complex because we are also planning a change-over to redundant shared disk storage and systems to access that storage in case of outages. So we will live with a monthly reboot to fix the disk problem for another 6 months (we won't run out of storage but will have a 2-minute planned outage on Sundays.

Testing

When we left off, we just started to move toward a test implementation of terminal servers using AWS WorkSpaces remote desktop (RDT) for what is effectively temporal microzones.

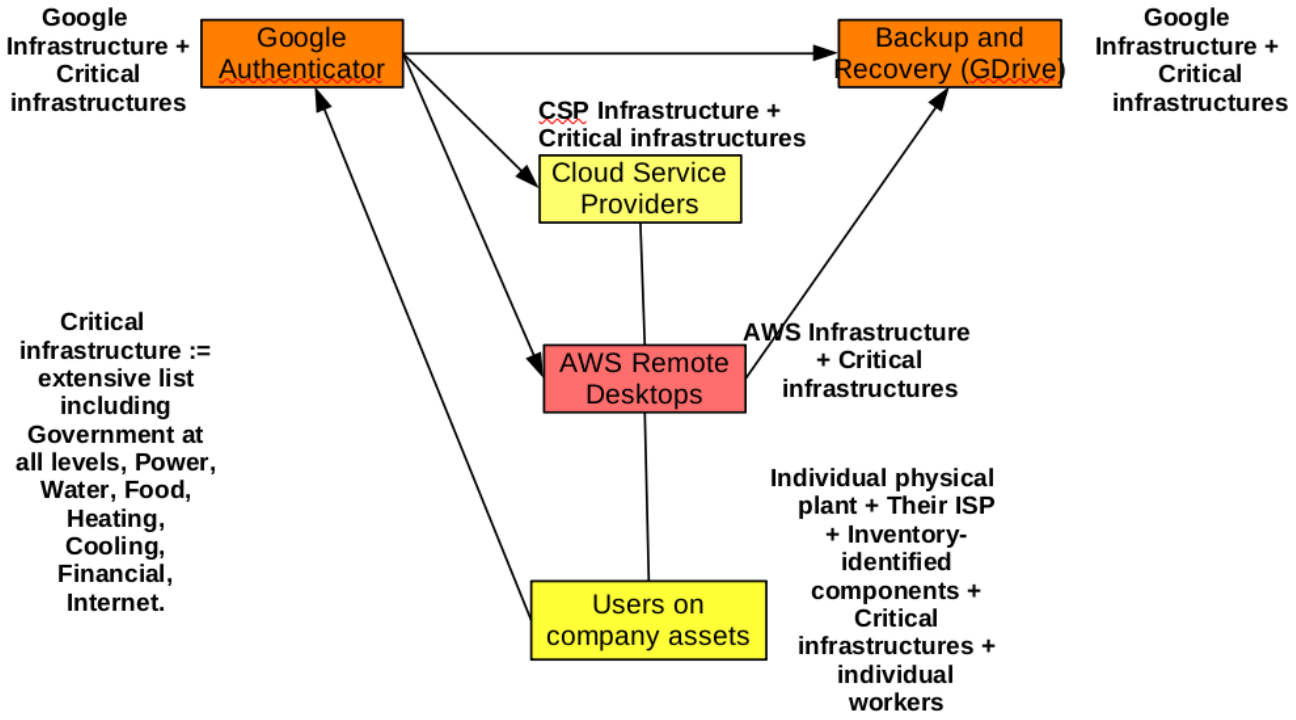


Testing the new cloud infrastructure became a priority for the week. Some examples of how this process works include initial deployment (getting new users onboard), basics for system content (it took me less than 2 minutes to install openoffice from scratch and start using it, and it ran faster on the RDT than it did on my local system), PDF files were visible through Firefox, but no PDF viewer was running native on the system, the file browser decided to make the extensions of files disappear, so I had to turn that feature back on so I could tell what was what (I hate the use of iconography when not really up to the task), etc. The plan changed from AWS as the separate and different backup repository to Google, because AWS was (temporarily) chosen for RDT. Note that the consequence levels of content and usage leads to low enough aggregated risk in any independent system for the surety needed.



The first step in real inventory and system dependencies started to reveal itself with the first steps in testing. We are starting to make the list of the software packages we want to install for different user bases. Note that we are not, for now, going to identify common deployed software other than the base operating system. This is because, for example, adding the Microsoft Office suite to a system not intended to use those mechanisms only increases the

risk and requires additional licenses. Thus we lower risk and cost at the same time by not providing capabilities not required for the specific tasks of the desktops. A system intended to run social media campaigns will need a very different set of tools than one for maintaining HR records, and those are quite different from a system used only to access legal and regulatory libraries and specific databases. From the original conceptual dependency diagram we are starting to fill in the details.



As it turns out, multiple RDT instances can co-exist on the same computer at the same time. Thus a user doing both HR and Social media work over the same time period can be logged into the separate RDTs with a window for each. They don't cut and paste at the text level, but of course you can take partial screen shots and do it all with graphics, but you can also do that with your cell phone camera, so there is no perfect way to stop it at that level.

- The ability to resize desktops also support a wide range of application situations, and the use of relatively low cost wide screen televisions ($500 for a 60 inch 4K display and getting better over time) for displays allow much more flexibility for users in their home environment. Similarly, projection displays ($170 for a palm-of-the-hand 1080p) can be used on the road for 2 sq. meter or larger display areas, allowing road warriors to have large complex display environments in hotel rooms and at client sites with controlled access from their local laptop.

The WorkDocs file sharing (FS) mechanism also allows limited movement of files and directories between RDT machines, and once we get it working right, we can likely use its approval mechanism to control movement of content from area to area, so change control can be used, for example, when moving new content, like new hire announcements, from HR through change control (review and optional approval), to the social media area for release.

Use of file encryption for the entire computer is fine in the sense that it limits access by someone getting control over the raw storage, but it doesn't help at all for someone planting a Trojan horse or gaining credentials for the VDT. So this approach can be augmented by local file encryption of areas thought to require an additional layer of protection.

- However, this also produces potentials for denial of access, and backups only restore the encrypted content. It's a great technique for ransomware to use. It's really only good for confidentiality and works against availability and other similar things. For now, we have chosen to encrypt disks at the OS level, but not anything where a user needs to provide an explicit password to access files within their file system. This eliminates unnecessary complex key management.

It's also noteworthy that the least expensive RDT solution only runs systems when they are in use. When you disconnect from the RDT, after an hour of non use (by default), the systems hibernate. When you reconnect they come out of hibernation in less than 60 seconds. Assuming remote storage is in use for shared content, this limits exposure of the systems to exploitation while retaining full operation for use of the content.

- For example, if someone gets a Trojan horse into a RDT, it will only run as long as the machine is running. That means that instead of getting 24x7 access, they only get access for limited periods of use. There are lots of potentials for harm in those time frames of course, but on the other hand, there are also lots of times when the system is not available to try to get in or exploit it if you are in. The RDT systems also automatically do a restart once a week (by default). Again, in this case, for user access to content stored in the file storage solution, reboots are no problem, and this also clears memory resident bad things as well as all the content remaining from memory holes in applications and operating environments.

This particular technology set has been a long time coming and has improved in the last several years to where it is not a viable enterprise solution that is less expensive, more effective, and easier to manage for most low and medium surety situations with good Internet access. It affords options for very high availability, allows selection of locations of resources so that multiple instances can be available on different continents, places within continents, or set in different countries to meet the combined requirements for facility distance and local storage and access within a country or region.

- Note particularly that legal requirements in different jurisdictions require local storage and use, so this solution can meet EU non-dissemination requirements while also meeting the 250 minimum distance requirement commonly associated with redundant data centers for larger than regional companies. For more details on these requirements, see the standards of practice at all.net[1]

A few problems were identified that need to be resolved for feasibility and ease of use. These include:

- There is an apparent delay between changes in WorkDocs and their appearance on virtual disks of VDTs. This means waiting or using old versions longer than necessary.
- The remote desktop client from AWS doesn't support remote video on Linux (so far).

1 http://all.net/ → Protection

New decisions

As we are at the end of the control architecture and now doing initial technology selection, we have paused the adoption of new decisions that reach into the technical security architecture while we test out the current technology approach. This is important because, while we can claim all sorts of things about the technical security architecture, we have to implement it in things that exist today.

- For a large enterprise going after a long term solution, R&D and high cost custom systems may be feasible, but for a small non-security company trying to grow, using commercially available inexpensive solutions is the only real option.

While the Standards of Practice take company size and surety into account and thus likely have feasible reasonable cost solutions identified, by spending some time on the technology and testing it out in a sample implementation provides far higher certainty that the technical solution will actually be suitable to the need.

- Within a few weeks, sample users will be familiar enough with the technology and understand its suitability to allow us to finalize longer-term decisions on the technical security architecture.
- We also don't yet have defined policies, in part because we don't yet have duties to protect, but also in part because it takes time to write them. We could copy from some policy library somewhere, but then we would have policies not reflecting what we actually do, and that would lead to liability as well as incompatibility between the policies and the realities.
- We are likely meeting all of the requirements of standards likely to be chosen (e.g., ISO 27001 and 27002 and others) because the SoP tends to meet or exceed those standards, we haven't checked against them explicitly yet because we cannot really fo so without an actual implementation in place and operating.
- Procedures are just being developed and documented, and this takes time to get started as well as time to do each specific thing for the first time, then repeat it, and generate the resulting documentation of how it was done and will be done. From there we get to checklists so we can have people do what they were supposed to do and confirm that they have done it without resorting to hand written notes and emails.

The list of things we cannot do quite yet is extensive, so we are putting further formalization of decisions on hold till we get through the initial technology in good enough shape to run with it.

Working through the details

Some solutions that arise from the RDT approach include mitigation of all sorts of risks associated with 3rd party applications on user computers along with augmented apparent capabilities and resources. Some examples will help get a sense of this.

- One example of this is Zoom, an extremely popular and widely used live online remote (LOR) application with widely published and known major security problems, all likely to continue over time. An obvious solution with RDT is to run Zoom on a dedicated RDT with read-only access to storage required for zoom sessions. The video from my camera didn't work when I tried this for the first time, but the audio worked, including recording and playback, which means things like screen sharing with playback of

videos worked well. Since this system is in a major provider's cloud infrastructure, the bandwidth and delay problems are far less than when they ran from my local connection, and thus the whole user experience is better for meetings. There is also no worry about leakage across meeting types if different Zoom desktops are used for different work zones.

  - I tested running the zoom meeting from the cloud and simultaneously running zoom from the browser on my desktop machine, and it worked without a problem. So you can run the meeting itself from the cloud (in the RDT window) and join the meeting in another (e.g., Web browser) window for video and related meeting activities and for controlling complex sessions, while using the local browser (or virtual machines locally) to limit the extent of consequences from Zoom. You can also run Zoom from a Chromebook or similar platform to separate it completely from your higher consequence systems. I tested recorded videos, sharing multiple windows, and so forth, and it works better than the local versions, likely because of the higher performance of the cloud solution and network.
  - However, Zoom is still a major security issue. So… we (not the company at this point – just Management Analytics and some of its ecosystem partners) use a private server version of Jitsi running on AWS. This can also be used from the RDT, as well as from local systems. This server is running on a platform we operate at AWS, costs about $30/mo for more users than we have hit yet, and the communications are limited to the endpoints talking to the server via encrypted SIP protocols. The net effect is that when running from RDTs, we have a private channel running without external connections (other than the RDT) through encrypted tunnels and it runs faster than if we ran it from our individual systems.
  - Another really interesting point to make here is that by running multiple applications that do screen sharing and so forth, I can have a Zoom meeting and also share the audio and session with 3rd parties through a Jitsi meeting. Nobody in Zoom will be aware of it, and of course recording can be done covertly using the 3rd party application. For example insiders can have relatively secure access to the meeting from their Jitsi platforms while Zoom users take the higher risks associated with that platform. Because Jitsi supports multiple sessions on the same computer, such a setup could even be used for allowing multiple differently controlled subgroups to have different access, share information covertly, and augment the capabilities of the local representative to the Zoom (or other LOR meetings) to make them appear to be far more of an expert than they really are, and to reflect the decisions and views of others in a negotiation as their representative, all in real time. Again, this is facilitated with the cloud environment because you can get the bandwidth, low delay times, and performance desired without having to deploy that to multiple endpoints. Of course beware that others may do this to you as well...

- File sharing (FS) in the cloud infrastructure also has its pluses and minuses. The key tradeoffs in terms of protection objectives are (no surprises here) integrity, availability, confidentiality, use control, accountability, transparency, and custody. And of course cost has to come into play for any real application.

- **Integrity:** Intentional or accidental by the provider, insiders, or 3rd parties must always be considered. We will assume for the moment that the provider is not intentionally corrupting content and rather seeking to protect it. If that's true, we can download content to our own independent repository, verify it's cryptographic checksum is changed or unchanged, and for changes, test on an independent system, not otherwise connected to the Internet, to verify it works there. As we come to see this working repeatedly, we may come to trust the process more and more and move toward statistical testing with the statistics related to the consequences of failure (for extreme consequences the testing is 100%, and lower consequences testing requirements depend on the trade-off in testing costs vs. consequences of lost integrity).
- **Availability:** Availability in the cloud environment using major providers is clearly way better than it is on local systems in almost all cases. The reason for this is that they can afford to and run in redundant high quality (Tier 4[2]) data centers all over the World with good physical security and maintenance processes that most smaller entities cannot afford, with redundant connections and infrastructure, and near the core of the Internet in multiple regions. The exceptions are:
  - **Local availability when offline:** This problem is largely solved by the local cache approach described elsewhere in this write-up.
  - **Availability when the cloud provider is offline or loses/screws up your data:** This is solved by the separate and different downloaded and verified backup and recovery approach described above, combined with the use of commodity capabilities available for changing providers as/if needed.
- **Confidentiality:** It is unclear that small or medium entities have any hope of maintaining confidentiality any more than the larger providers do. As such, the challenge is one-time access grants unlimited exfiltration (given enough time).
  - The access control mechanisms and temporal microzone approach restricts access of content to systems associated with users and purposes. This includes limiting read vs. write vs. delete operations and allows for an approval process.
  - At the end of the day, anything a user sees can be copied and most user systems can be compromised over time unless severely restricted as to use.
  - Ignoring the leaks by insiders (accidental and intentional) that bypass the control mechanisms because the access is authorized, and assuming the control mechanisms operate as intended, any added protection is essentially against the service provider (e.g., AWS, Google, etc.) and malicious actors who bypass the controls.
    - Bypasses are limited by application limitations on systems, and of course no direct access to any of these systems is permitted.
    - But that's really all we can do without adding complex, expensive, hard to manage, high false-positive and false-negative protection mechanisms… Which we have not gotten to decisions regarding yet...

2 https://uptimeinstitute.com/tiers for details.

- U**se control:** Control over use of mechanisms in this approach consists of two areas of coverage:
  - **Authentication of users and systems:** Multi-factor authentication is selected to the desired surety level, and implementation will follow the initial testing period for access to all VDTs and remote systems under company control. No additional controls are required for the surety level desired.
  - **Limitations on what can be done from where and what software is available from there to do it:** This is controlled by limiting the software on RDTs and the fact that these systems are the only authorized means to perform the functions.
    - Remote functions from there (e.g., twitter feed) is controlled by access to the feed using passwords in those systems. To the extent these systems allow for Google authenticator authorization for access, this can be controlled in the same way as the VDT system access, and as such, the access has the same effective technical controls as if it were running from VDT.
    - However, since the VDT controls are different from the user endpoint controls, user endpoint system owned by the enterprise will be required in order to increase use control for external cloud providers, unless they have additional mechanisms (such as federated identity through the AWS Active Directory instance). This is a risk management and management issue.
- **Accountability:** Accounting for actions is largely a matter of having adequate logging and attribution. Software Bill of Materials (SBOM[3]) and Digital Bill of Materials (DBOM[4]) may be used to augment existing logging controls to allow provenance to be established in the attestation channels they provide, but that will wait till the existing systems are operating under their current control scheme.
  - Audits (i.e., logging) will be turned on to the extent feasible, and the storage solution should be configured to keep backups and attribute all versions to the writer(s) and reader(s) of those versions, which transports across from VDTs to the storage solution and back automatically.
  - Augmenting this for 3rd party cloud-based applications depends on those application and what they provide, and as such, functionality will be almost always favored over audit capabilities.
  - Auditing to the virtual keystroke and mouse movement is of course possible, but seems too extreme for the present situation. It also comes at a cost, and of course all of this auditing only have actual effect when analyzed. So it's main purpose is forensic for detected events in investigative and legal frameworks, none of which has been identified from the duties to protect yet.
- **Transparency:** Transparency applies particularly to specific content in this particular company. Without going into details, the real requirement is in the financial and operational aspects of the company where independent 3rd party

3 https://en.wikipedia.org/wiki/Software_bill_of_materials
4 https://www.linkedin.com/pulse/supply-chain-attestation-grid-chris-blask/

providers are used for the very purpose of supporting transparency. In addition, these weekly write-ups constitute more transparency than almost any company ever has with regards to the protection program. The only thing lacking is the company information, which would become transparent as/if required by external forces or the duties to protect. Of course all the documentation that is being prepared and the systems of record provide the availability of transparency to authorized parties as and if that becomes an issue.

- **Custody:** Clearly, the custody of the cloud-based content is that of the solution provider. The question of whether and to what extent this meets the needs of the company is a matter of the duties to protect, and since Company doesn't have these defined yet, no determination can be made. Custody copies of content will be available through the backup mechanisms, of course, but the addition of DBOM and SBOM as and if higher surety requirements appear will likely be the path to added custody controls as/if needed. Ultimately an archival quality records management solution would be the desired approach, but the limitations of commercial systems limit the feasibility vs. the cost at this point.

As should be apparent by now, these decisions are all rational, but are also a matter of trust. More on that to come.

- Testing before deployment is critical to change control as well as saving a lot of time. An example of this is codec compatibility in Windows. The AAC codec commonly used in the Internet is not native to Windows Media Player, but it works perfectly well if you download some software from the Internet as a plug-in. You have to reboot, and the download attempts to get you to download and run lots of other software (presumably part of their advertising model). So I downloaded, tested, and found that it worked (after reboot), but only if you open with Windows Media Player classic.
  - While that's fine for me, how do I determine the trustworthiness of this process for the rest of the users? In come the Trojans. All I can really do is confine its extent to the VDTs I am willing to risk it on. Which comes to the idea of non-standard setups… each setup only downloading and allowing the additional software required for necessary functionality. So setups are tested from a from-scratch system, with procedures documented for reuse when desired, downloaded software preserved (in case it's no longer available later), and a set of libraries with procedures for use are built up over time.
  - One of the big problems with this approach is that as systems and software are upgraded, these older methods stop working at some point (at least most of them do), so when do you fix them? I think the solution is to control changes so that you roll them out slowly as and if needed. When a new version of something comes out, you test it to make sure the key functions still work, retain a copy of the old VDT environment (cloud services commonly let you snapshot), and move forward while retaining the ability to go back (temporarily) as you fix the next problem or adapt your processes to it. The big win here is that you only have to deal with it when the change is actually required and the existing solution doesn't work.
  - So your rolling updates retain older versions not yet susceptible to new attacks, but still susceptible to old attacks. Of course the other restrictions on the operating

environments are part of the defenses against these older (and newer) attacks, and also reduce the problem of risk aggregation through monoculture (all the same means that they all fall at once). Redundancy, at its heart must be separate and different to with stand the environment it exists in. The question of how separate and how different is a matter of risk management. And that trades convenience, functionality, cost, and so forth. It's not all about "security" (whatever that is).

- Risk aggregation and trust are at the heart of the risk-related decisions being inherently and implicitly made, and making them explicit is important to a security program.
  - **Risk aggregation:** As more and more content, capability, etc. is placed in the cloud infrastructure, more trust is placed in that infrastructure because more potential for loss is placed there. This aggregation of risk puts more weight (potential for loss) on the provider by taking weight off of other mechanisms. So while there was previously weight on each of a set of independent resources (i.e., the individual computers) and each could only have so much total effect on consequences, there is now less weight on those systems and more weight on the provider.
  - **Trust:** The central issue of trust is that losses result from trust given not realized in execution. Trust is the other side of the weight issue. The surety level of the provider, their Tier 4 data centers, and their capabilities and resources to deal with security issues far exceeds that of the Company.

  The approach taken here is to disaggregate the risk by distributing resources both within and between service providers. Backups at and an ability to recover to a different service provider reduce the potential for loss associated with long-term outages. Because the nature of Company is such that hours to days of outage are not devastating, there is no need to go to higher surety than normal commercial protection. Thus the disaster recovery plan (essentially a massive service provider outage) is to recover to another provider. Company will not survive the global devastation from a high meteor collision with Earth, and that's pretty much the level of disaster required to wipe out all available redundant mechanisms in today's information infrastructure.

The cost of AWS

Cost prediction is not a trivial matter in these sort of leased cyberspace approaches. To get a sense of this, a minimal Windows Desktop running only when in use on AWS costs something like $0.30/h (30 cents/hour). That's $216/mo if you run it 24x7, or $48 if you run it for 160 hours. Add directory services ($16), bandwidth usage, application leasing (Microsoft Office 2013 Professional for example is $15/mo, add-on security packages are in the $3-$4/mo, and so forth), and a few other charges and you can easily get to $100/mo/user for 40 h/week user. If users keep their instances open all the time, this can run into the range of $300+/user/mo. Considering that the cost of a PC is only a few months rent, if it were not for the advantages in terms of control, availability, performance, and other capabilities, this would not be a very cost efficient approach. But the main benefit is really in security. This is where they run the servers in a far more secure physical facility, they keep things operating 24x7 and support global redundancy (for a fee), they provide the mechanisms for backups and controlled file sharing, they handle automatic update integration, they manage the authentication schemes, they run the active directory instance, and so forth. It's a tradeoff.

Testing with Azure

As a quick comparison, I tried to create a Microsoft Azure account using their “free trial”. But as soon as I created my ‘free’ account, after entering my name, address, birthday details, credit card details, and getting two independent verifications (at least 3-factor authentication), they told me I already had an account and was not eligible for the free account. Except of course that’s just not true, at least as far as I am aware. Now I have had previous experiences with Microsoft automatically trying to bill me for things I never bought, and I didn’t want to get into their exploitative recurring revenue model again, so I decided to get out while I could.

But then a miracle happened… somehow I ended up on a page that allowed me to create a VM ($70/mo minimum charge). A 12-character minimum length password auto-generated by my browser, and Microsoft asks me to confirm I have an eligible Windows 10 license with multi-tenant hosting rights… but I am paying them for their machine… how exactly do I get the license other than by the purchase I am making? I have no clue. Then I have to decide on my disk type and encryption type, create and attach a new disk (they try to charge me for the high end, I go for the low end), etc. So finally I manage to get to where I can push the “Create” button, and … Validation failed… We are getting into the Google complexity level for the so-called instant creation of a virtual machine. Lo and behold, I had to lie about having a license for Windows to get them to start. They have repeatedly told me I need to have RDP open so I can access the machine, and yet they also keep warning me it is a bad idea and insecure. But how else can I do remote desktop? They don’t tell me.

So I have to find my own RDP client, login to a virtual machine experience, look at lots of BS things that Microsoft apparently puts there to enhance my user experience, and no audio output device, virtual assistants popping up all over the place, the windows doesn’t resize with the desktop, I get things that pop up asking me to confirm (no other apparent option), and I start to push buttons to try to get it usable… It is fast – apparently far faster than what I get locally. A good sign. The Edge browser doesn’t properly handle some tables (no surprise there), so some of my output doesn’t look right, and the font is not very good, but it appears to work, at least 3 orders of magnitude better than Explorer… again no surprise. There is a built-in virus scanner, a good thing I think, but essentially no other software provided.

I downloaded and installed open office in about 1 minute, and things seem to run reasonably well. Time will tell, but it appears that for about twice the price, I can run Azure, although operationally, it will be more time consuming and expensive to manage, because it operates more like Google than AWS in terms of ease of use and convenience for the systems administrator. Similarly, I know there are file sharing and similar mechanisms in the Microsoft world, and I have confidence that Azure will be a bastion for recovery and reconstitution of the company if some day AWS is unable to perform as needed. There is an automated backup and recovery mechanism, change tracking, inventory mechanism, policy control system, and logging, capability manageable from the control center, and other similar mechanisms are present. There is also an automated “advisor” for usage of the platform, and interestingly, it claimed I was doing everything right, including “security” where I went against their advice to allow remote desktop and did nothing to configure security at all. I guess they think they are secure out of the box… Of course no decision has been made with regard to the choice yet, but it’s good to know there are alternatives available.

Making the decision

A decision such as the one to go with Azure or AWS is not a trivial one. As such, while a satisficing solution may do, it's worth the effort to do a real comparison if the platform will be in use for the next year or three, and possibly a lot longer. So the next step will be to go to a Decider[5] process. Hopefully we will decide in a week or three and then start to implement after the user base has done initial testing.

Transition to RDT (with security changes)

The transition to RDT is pending acceptance of the costs, performance, and usability of the platform, or selection of a different platform (e.g., Azure). This process is planned to go something like this:

- Security related steps during the roll-out:
  - Use the built-in inventory mechanisms for technical inventory of software and configurations.
  - Use the built-in identity management infrastructure, configured as needed.
  - Use the built-in file sharing mechanisms in accordance with the ABAC approach.
  - Use enhanced logging/auditing capabilities, likely enhanced with 3rd party analytics.
  - Identify other mechanisms that meet requirements as we build through the SoP to Defined maturity and implement them in the context of the systems in use.
- Testing is now underway with the CEO. More users wil try the test environment for feasibility for an hour or so to assure that performance and related matters work, and we will try to start having company LOR meetings on the RDT platform as part of that testing.
  - Until we get YES from this process, we will keep adapting.
- As soon as we get the Yes, we will transition LOR meetings (they currently use Zoom which is endangering their local system) to the RDT environment and all Zoom meetings will be sourced there, even if some remote users use the Browser interface on their local systems for access.
  - Since LOR meetings require infrastructure in any case, they cannot be held without Internet access and a working user platform and cloud service. The addition of AWS RDT will not significantly hinder availability, and it will likely increase performance of the meetings, while forcing relevant content to be made available and synchronized through the WorkDocs or equivalent file sharing mechanisms.
- As more functional components are tested in the test environment, they will be transitioned to live use in a similar manner. This will also transition the content into the remote storage solution, and allow for automated backups and recovery, version control and reversion, shared access according to the ABAC policies, and so forth.
  - This process will also support retention and disposition, the association of people, content, systems, and businesses, help to resolve legal holds, backup and restoration processes, and record keeping overall.

5 http://all.net/Decider.html

- Users may or may not transition to purely company-owned resources, and as the remote desktop solution looks more and more usable, the CEO is leaning toward the possibility of using a user solution like a Chromebook or similar mechanism at lower cost and with different built-in protective mechanisms and limitations.

More information from the group

As we are transitioning to working with the group of companies instead of the individual company, we are starting to have meetings with top management for the group and the subsidiaries. One example is the need or desire to support a key team member in China.

- The subsidiary believes it needs this worker for now, at least until such time as the subsidiary grows large enough to have redundancy in key people.
  - The worker is in China and not likely to immigrate to somewhere else any time soon. Among other things, this is also handy for watching over any China-based operations, management, manufacturing, sales, etc.
- The subsidiary identifies concerns that the Chinese government might gain access to content and use it to violate pre-market pre-patent intellectual property confidentiality, perhaps even filing first and causing provenance problems with respect to ownership.
  - I identified that we will not be able to build systems capable of stopping China from surveilling people working from China. Rather, what we can do is limit the access from the worker(s) in China to things needed to do the job, and with the RDT solution, this would be readily controlled and audited.
  - The subsidiary identified that the available mechanisms for access for this worker were currently only known to be Skype. I responded that we could put Skype on a VDT for that access, and from the VDT access other VDTs based on need and authorization. I also identified that, the last time I looked, TeamViewer also worked from China.

This point also brings out the identified requirement to be able to divest efficiently. Mergers essentially always involve a painful process of reconciling and transitioning from the seller's way of doing things to the buyer's way, hopefully with both learning and taking value from the other. But an integrated system of companies often has far more problems in splitting apart, because of the shared… everything.

- The companies in the group will be following pretty much the same general scheme of things, but as each grows, it will be separately resourced and operated. This will support risk disaggregation for the group as a whole, while also easing divestiture and allowing for customization where needed. The arms length requirement is also important here, and in particular, the overall scheme of ownership of companies, intellectual property, and so forth has to be kept in proper control from a legal and financial point of view regardless. The structure of the solution will also gain the advantages of economies of scale (although it might not get all the same discounts).

Conclusions

Momentum is starting to grow, and solutions are starting to look like they will work, the program is maturing (IT as well as CyberSecurity), and we will start again to look at the next steps in SoP future state review in the coming week.

